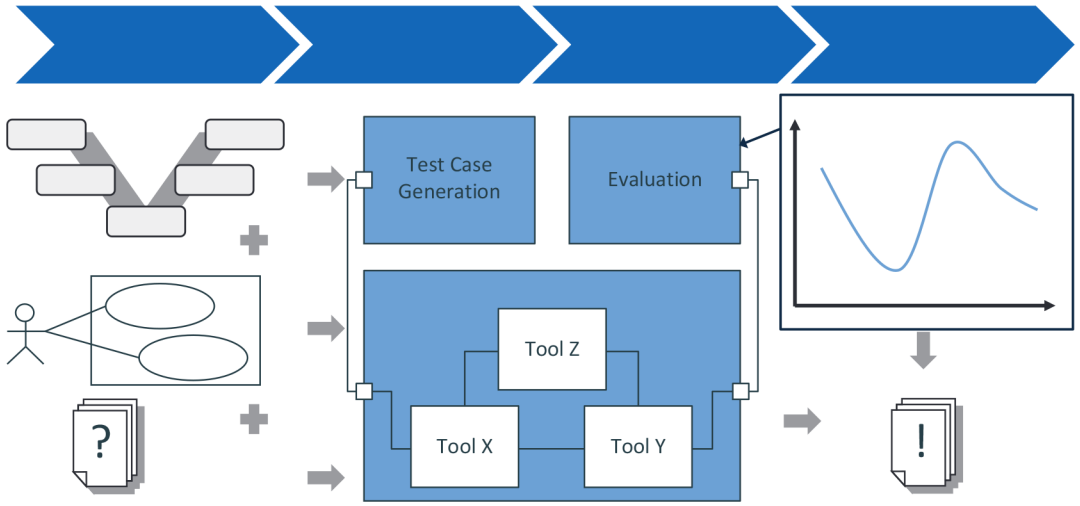
SE | RWTH AACHEN UNIVERSITY

Stefan Kühnel

Eine agile Methode zur simulativen Qualitätssicherung von Aktiven Sicherheitssystemen



SimAQuAss4ConTest

Aachener Informatik-Berichte,
Software Engineering

Band 51

Hrsg: Prof. Dr. rer. nat. Bernhard Rumpe

Eine agile Methode zur simulativen Qualitätssicherung von Aktiven Sicherheitssystemen

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der
RWTH Aachen University zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt

von

Dipl. Wirt.-Inf.
Stefan Kühnel
aus

Minden in Westfalen

Berichter: Univ. Prof. Dr. rer. nat. Bernhard Rumpe
Assoc. Prof. Dr. rer. nat. Christian Berger

Tag der mündlichen Prüfung: 13. Dezember 2021

[Kue22] S. Kühnel:
Eine agile Methode zur simulativen Qualitätssicherung von Aktiven Sicherheitssystemen.
Aachener Informatik-Berichte, Software Engineering, Band 51, ISBN 978-3-8440-8427-6, Shaker Verlag, März 2022.
www.se-rwth.de/publications/

Aachener Informatik-Berichte, Software Engineering

herausgegeben von
Prof. Dr. rer. nat. Bernhard Rumpe
Software Engineering
RWTH Aachen University

Band 51

Stefan Kühnel
RWTH Aachen University

Eine agile Methode zur simulativen Qualitätssicherung von Aktiven Sicherheitssystemen

Shaker Verlag
Düren 2022

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Zugl.: D 82 (Diss. RWTH Aachen University, 2021)

Printed in Germany.

ISBN 978-3-8440-8427-6
ISSN 1869-9170

Shaker Verlag GmbH • Am Langen Graben 15a • 52353 Düren
Telefon: 02421 / 99 0 11 - 0 • Telefax: 02421 / 99 0 11 - 9
Internet: www.shaker.de • E-Mail: info@shaker.de

Verzichtserklärung

Ergebnisse, Meinungen und Schlüsse dieser Dissertation sind nicht notwendigerweise die der Volkswagen Aktiengesellschaft.

Disclaimer

The results, opinions and conclusions expressed in this PhD thesis are not necessarily those of Volkswagen Aktiengesellschaft.

Wolfsburg, den 03. Juni 2021

Kurzfassung

Kontext: Neben dem aktuellen Bestreben die Elektrifizierung des Antriebs von Automobilen durch Innovationen voranzutreiben kommt der Integration von Fahrerassistenzsystemen im Rahmen der Automobilentwicklung eine besondere Bedeutung sowohl zur Steigerung des Fahrkomforts als auch zur Verbesserung der Sicherheit zu. Eine gewichtige Rolle spielen dabei Consumer Tests wie z.B. von Organisationen wie EuroNCAP, welche sowohl als Treiber aber auch zur Beurteilung der Leistungsfähigkeit der getesteten Sicherheitssysteme dient. Um die Qualität der dafür entwickelten Software besser beurteilen und steigern zu können, bilden Tests sowohl in ausgewiesenen Prüfarealen unter realen Bedingungen als auch simulationsbasierte Tests in synthetischen Umgebungen geeignete Ansätze, den Herausforderungen der kontinuierlichen Qualitätsverbesserung agil zu begegnen, wenngleich beide Ansätze über unterschiedliche Hürden und Grenzen verfügen. Speziell der hier betrachtete simulative Ansatz mündet nicht selten in dem Dilemma, dass die Entwicklung geeigneter Umgebungen ebenfalls ein hohes Maß an Ressourcen wie die eigentliche Systementwicklung benötigt. Dies hat zur Folge, dass diese weiter parallel voranschreitet und die Simulationsumgebung aufgrund des hohen Aufwandes nicht rechtzeitig einsetzbar ist. Solch eine Möglichkeit kann bspw. dann eintreten, wenn dem Aspekt der Modellbildung mit dem Güteziel einer möglichst engen Realitätsnähe ohne Rückkopplung auf die eingangs zu definierende Fragestellung: “Was soll durch die Simulation beantwortet werden?” erfolgt.

Ziel: Das Ziel dieser Arbeit besteht darin, den komplexen Entwicklungsprozess von Aktiven Sicherheitssystemen im Rahmen der Consumer Tests durch einen simulativen Ansatz zur Verbesserung der Softwarequalität zu unterstützen. Weiterhin soll durch die Verhaltensanalyse von Algorithmen die Ressourcenallokation während der Entwicklung und der notwendigen realen Tests verbessert und damit effektiver gestaltet werden.

Methode: Nach der Durchführung eines Systematic Literature Reviews (SLR) zu Prüfung evtl. vorhandener Ansätze und Methoden zur Entwicklung solcher Simulationsumgebungen wird eine eigene Methode entwickelt, vorgestellt und unter Berücksichtigung der vorliegenden Projektbedingungen im Rahmen einer Fallstudie angewendet.

Ergebnisse: Die Analyse des Projektkontextes kommt zu dem Ergebnis, dass es durchaus Simulationsaktivitäten gibt, jedoch eine strukturierte Herangehensweise zu ihrer Entwicklung fehlt. Das Systematic Literature Review bestätigt dieses Ergebnis, sodass der Bedarf der Entwicklung einer agilen Methode zur simulativen Qualitätssicherung von Aktiven Sicherheitssystemen insbesondere mit Blick auf Consumer Tests aufgezeigt wird. Die vorgestellte Methode umfasst

vier Bausteine: (a) die Analyse und Modellierung des Untersuchungsraumes, (b) die Entwicklung einer adäquaten Simulationsinfrastruktur, (c) die Entwicklung von Bewertungsverfahren und (d) die Durchführung von Simulationsläufen sowie ihrer Auswertung. Zum Schluss wird die Methode mit Hilfe einer Fallstudie für ein Proof of Concept angewendet.

Schlussfolgerung: Es wird aufgezeigt, dass die Methode während der qualitativen Bewertung von Softwarekomponenten auf simulativer Basis einen positiven Beitrag leistet, speziell dort, wo Äquivalenzklassentests nicht ausreichend sind, Consumer Test Szenarien hinreichend zu testen. Einschränkungen und Erweiterungsbedarf der Methode werden vorrangig in der Übertragung auf andere Kontexte im Fahrerassistenzumfeld und die Erweiterung um zusätzliche Consumer Test Szenarien wie den Fußgängerschutz gesehen.

Abstract

Context: In addition to the current efforts to advance the electrification of the powertrains through innovations, the integration of driver assistance systems in the context of automotive development is of particular importance, both to increase driving comfort and to improve vehicle safety. Consumer tests, e.g. from organizations such as EuroNCAP, play an important role, which serves both as a key driver for safety improvement and to assess the performance of the underlying safety systems. In order to be able to better assess and increase the quality of the software developed for this purpose, tests both in designated test areas under real conditions and simulation-based tests in synthetic environments are suitable approaches to meet the challenges of continuous quality improvement in an agile way, although both approaches have different hurdles and limits. In particular, the simulative approach considered here often leads to the dilemma that the development of suitable simulation environments also requires a high amount of resources like the actual system development. As a result, this continues to progress in parallel and the simulation environment cannot be used in adequate time due to the high effort. Such a possibility can occur, for example, if the aspect of modeling with the quality goal of being as realistic as possible is defined without feedback to the question at the beginning: “What should be answered by the simulation?”.

Goal: The main goal of this thesis is to support the complex development process of active safety systems in the context of consumer tests with a simulative approach to improve software quality. Furthermore, the allocation of resources during development and the necessary real tests should be improved and thus made more effective through the behavioral analysis of algorithms.

Method: After carrying out a Systematic Literature Review (SLR) to examine any existing approaches and methods for the development of such simulation environments, a separate method is developed, presented and applied in a case study, taking into account the existing project conditions.

Results: The analysis of the project context comes to the conclusion that there are simulation activities, but a structured approach to their development is missing. The Systematic Literature Review confirms this result, so that the need to develop an agile method for simulative quality assurance of active safety systems, especially with regard to consumer tests, is shown. The method presented comprises four building blocks: (a) the analysis and modeling of the investigation area, (b) the development of an adequate simulation infrastructure, (c) the development of evaluation methods and (d) the implementation of simulation runs and their evaluation. Finally, the method is applied with the help of a case study for a proof of concept.

Conclusion: It is shown that the method makes a positive contribution during the qualitative assessment of software components on a simulative basis, especially where equivalence class tests are not sufficient to adequately test consumer test scenarios. Limitations and the need for expansion of the method are primarily seen in the transfer to other contexts in the driver assistance environment and the expansion to additional consumer test scenarios such as pedestrian protection.

Danksagung

Eine Danksagung ist ein wichtiges und zugleich herausforderndes, weil emotionales Dokument: Es dokumentiert die Würdigung all derer, die Teil jenen Erfolgs einer Sache oder Person sind, welcher ohne ihr Wort und ihre Tat nicht erreichbar gewesen wäre. Die Danksagung ist Ausdruck der tiefen Verbundenheit zwischen dem Verfasser und denjenigen, denen er seine Dankbarkeit entgegenbringt. Gleichwohl lastet die Bürde der vollständigen und fehlerfreien Erwähnung aller Beteiligten am Erfolg schwer auf den Schultern des Verfassers. Dennoch sei an dieser Stelle die Gelegenheit ergriffen, Danke zu sagen, wenngleich die Gefahr der fehlenden Erwähnung wichtiger Weggefährten allseits besteht. Möge sie mir nicht widerfahren!

Ich bedanke mich bei all den Menschen, die mich auf dem Wege dieser Arbeit begleitet und unterstützt haben. Im Besonderen möchte ich mich bei meinem Doktorvater Prof. Dr. Bernhard Rumpe bedanken, der mir die Möglichkeit und den Rückhalt gegeben hat, diese Arbeit als externer Doktorand bei der Volkswagen AG in Wolfsburg zu erarbeiten und fertig zu stellen. Speziell danke ich ihm für seine Betreuung meiner Arbeit, sein Feedback und sein mir entgegen gebrachtes Vertrauen, obwohl ich nur begrenzt am Lehrstuhlleben teilnehmen konnte. Herzlich bedanken möchte ich mich bei Prof. Dr. Christian Berger von der Chalmers University of Technology, der als Zweitgutachter und als enger Wegbegleiter diese Arbeit und mich in den ersten Jahren durch intensive Diskussionen entscheidend geprägt hat. Bei Herrn Prof. Dr. Bastian Leibe bedanke ich mich für die Übernahme des Vorsitzes meiner Prüfungskommission. Frau Prof. Dr. Erika Abraham danke ich für die Bereitschaft, als viertes Mitglied der Prüfungskommission teilzunehmen und mich zur Theorie der hybriden Systeme zu prüfen.

Einen großen Dank möchte ich den Kolleginnen und Kollegen am Lehrstuhl für Software Engineering aussprechen, die mich bei den Aufenthalten am Lehrstuhl und bei den Workshops sehr entgegenkommend und freundlich aufgenommen und durch ihr Feedback zum Erfolg dieser Arbeit beigetragen haben. Besonders bedanken möchte ich mich bei Dr. Arne Haber und Dr. Dimitri Plotnikov, die mich bei dem Industrieprojekt entscheidend unterstützt haben. Weiterer Dank gilt Sylvia Gunder, Sonja Müßigbrodt und Marita Breuer, die mir bei den organisatorischen Aspekten und bei der Veröffentlichung dieser Arbeit zuvorkommend und tatkräftig halfen. Dr. Oliver Kautz und Dr. Evgeny Kusmenko danke ich für die Antworten auf meine vielen Fragen zur Einreichung, zur mündlichen Prüfung und der Vorbereitung zur Veröffentlichung dieser Dissertation. Sebastian Stüber danke ich für die Organisation des obligatorischen Doktorhutes, womit ich bis zu seiner Nachricht nicht gerechnet hatte.

Bei der Volkswagen AG danke ich Prof. Dr. André Leschke für die Möglichkeit, als Dokto-

rand in der EKSE-5 tätig sein und diese Arbeit anfertigen zu dürfen. Auch danke ich ihm für die Unterstützung bei der Freigabe zur Veröffentlichung meiner Dissertation. Bei Delf Block und Christian Hons bedanke ich mich für die fachliche Unterstützung zu meiner Arbeit. Weiterhin möchte ich mich bei all den Kollegen der EKSE-5 bedanken, die mich in der Zeit bei meiner Arbeit fachlich sowie moralisch begleitet und gelegentlich abseits des Büros davon abgelenkt haben. Ich bin sehr dankbar dafür, dass ich Euch kennenlernen durfte. Sönke Heeren und Vladislavs Serebro danke ich ebenfalls für ihre Unterstützung bei der Umsetzung und den Veröffentlichungen. Danken möchte ich auch Dietmar Spring, der mich in den vergangenen Jahren stets ermutigt und moralisch unterstützt hat, diese Arbeit fertigzustellen.

Zum Schluss möchte ich meiner gesamten Familie für ihre Unterstützung und den nicht nachgebenden Rückhalt während all der Zeit danken. Meinen Eltern danke ich für die Möglichkeit, diesen Weg gehen zu können. Ein ganz besonderer Dank gilt meiner Frau Vivien, die viele Entbehrungen gerade in der letzten Zeit hinnehmen musste und mir die Freiheit gab, diese Arbeit fertigzustellen. Darüber hinaus danke ich ihr für das Korrekturlesen dieser Arbeit. Danken möchte ich meinen beiden Söhnen, die einige Zeit auf ihren Vater verzichten mussten. Stefan Kiel danke ich an dieser Stelle für die zweite Korrekturlesung. Ich danke Euch!

Minden, Januar 2022

Stefan Kühnel

Inhaltsverzeichnis

1 Einleitung

Die Entwicklung neuer Automobile ist mittlerweile nicht mehr nur auf die Perfektion und Weiterentwicklung mechanischer Bauteile fokussiert, sondern es wird sich seit der Jahrtausendwende zunehmend auf die Integration von software-intensiven Systemen und auf die Möglichkeiten der Umfelderfassung durch Computertechnologien konzentriert, um damit assistierende Systeme für den Fahrer zu konzipieren. Dies hat in den letzten Jahren dazu geführt, dass eine Vielzahl an innovativen Fahrerassistenzsystemen integriert wurden, welche auf Grundlage von Umfelddaten wichtige Informationen für den Fahrer erzeugen und ihn gegebenenfalls durch systemseitig initiierte Aktionen unterstützen. Diese Assistenzsysteme im Fahrzeug beginnen beispielsweise bei erweiterten Head-Up-Displays oder Infotainmentsystemen, über welche die Fahrzeugcharakteristik wie das Motor- oder auch Fahrverhalten durch den Fahrer verändert werden kann und reichen bis zu solchen Systemen, die dauerhaft die Umgebung über Sensoriken erfassen, bewerten und in kritischen Situationen aktiv in die Fahrzeugführung eingreifen. [Vol19b] [Vol19a]

Die Abbildung 1.1 verdeutlicht diese zeitliche Entwicklung und die Integration von Systemen ins Fahrzeug, die ausgewiesener Maßen ein Schutzpotential für den Fahrer und weitere Insassen bieten. Sie gibt auch einen Ausblick darauf, welche zukünftigen Umgebungsdaten für Fahrerassistenzsysteme genutzt werden können.

Zum Teil werden Fahrerassistenzsysteme auch als Sicherheitssysteme bezeichnet, da sie in Notsituationen im Straßenverkehr oder bei kritischen Fahrmanövern in die Führung des Fahrzeugs eingreifen. Zu solchen Systemen zählen beispielsweise Notbremssysteme und Pre-Crash-Systeme mit dem Ziel, Unfallrisiken zu reduzieren, potentielle Unfälle zu vermeiden oder die Folgen von Unfällen zu minimieren. [Vol19c] Dabei hat der Anteil von Software stetig im Fahrzeug zugenommen und beträgt mittlerweile mehrere Millionen Lines of Code, wobei aufgrund der Weiterentwicklung und der Funktionserweiterungen der Systeme dieser Anteil weiter zunehmen wird. [Sie19] Speziell durch diverse politische Initiativen in einzelnen Ländern und auch auf europäischer Ebene wird die Integration von solchen innovativen Fahrerassistenzsystemen weiter zunehmen und insbesondere ihre Entwicklung vor neue Herausforderungen stellen. [Com19] Gleichzeitig kommt es aber mit Blick auf das autonome Fahren zu einem Engpass: So

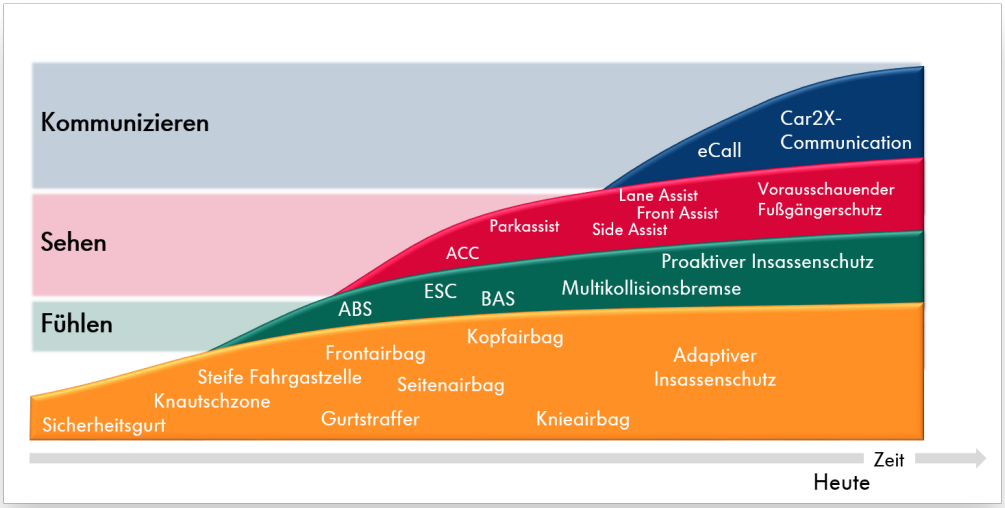


Abbildung 1.1: Zeitliche Integration von Assistenzsystemen ins Fahrzeug (Volkswagen AG).

sind die technischen Voraussetzungen für eine automatisierte Steuerung mit Level 3 bzw. Level 4 bereits technisch realisiert, jedoch gestaltet sich die Gesetzgebung und Freigabe für den Straßenverkehr weiterhin herausfordernd. [May19] Eine der Gründe für die fehlende Umsetzung kann u.a. auch in der schwierigen Beantwortung der Frage gesehen werden: ”Ab wann ist eine Funktion tatsächlich sicher?´´

1.1 Motivation

Diese wachsende Bedeutung der Fahrerassistenzsysteme als Haupttreiber für Innovationen im Fahrzeug und ihre zunehmende Integration machen es erforderlich, sich insbesondere mit ihrem Entwicklungsprozess intensiver auseinanderzusetzen, um die Qualität des jeweiligen Assistenzsystems weiter auf hohem Niveau zu gewährleisten. Hierbei kommt dem Testprozess als fundamentalem Teil des gesamten Entwicklungsprozess eine Schlüsselrolle zu, weil eine mögliche Fehlerwirkung nicht nur beim Kunden einen negativen Effekt, sondern auch negative Konsequenzen für Mensch und Material zur Folge haben kann.

Ein Fahrerassistenzsystem besteht demnach aus mehreren Hardware- und Softwarekomponenten. In diesem Zusammenhang wird häufig von einem “eingebetteten System” gesprochen (*engl.: embedded systems*). Edward Lee [Lee07], [Lee08]:

> *Embedded Systems are information processing systems embedded into enclosing products. [Lee08, S. 1]*

Das Produkt "Fahrzeug" verfügt also über ein integriertes System, welches Daten zu Informationen aggregiert, bewertet und weiter prozessiert. Damit Informationen jedoch verarbeitet werden können, ist auch entsprechende Software notwendig. Lee fügt dem hinzu:

> *Embedded software is software integrated with physical processes. The technical problem is managing time and concurrency in computational systems. [Lee08, S. 1]*

Somit umfasst eine erste Charakterisierung von eingebetteten Systemen die Merkmale **Zeit**, **Informationsverarbeitung** durch Software, **Gleichzeitigkeit/ Parallelität** von berechnenden Systemen, Einbindung in **gekapselten Produkten** und **Einbindung in physikalische Prozesse**. Zur Verarbeitung von physikalischen Größen werden verschiedene Technologien im Bereich der Sensorik und Aktuatorik verwendet, welche analoge in digitale Signale bzw. vice versa umwandeln. Hierdurch sind sie einerseits in der Lage, ihre physikalische Umwelt zu erfassen, andererseits aber auch physikalisch Einfluss auf die Umgebung ausüben zu können.

Eine Vielzahl an Beispielen für eingebettete Systeme ist heute in diversen Branchen und Domänen zu finden und reichen von der Luftfahrt- und Automobilindustrie über die Produktionswirtschaft und die Telekommunikationsbranche. [Sie19] Für die Automotive-Domäne wurden bereits eingangs mehrere prägnante Beispiele angeführt; weitere wichtige Systeme sind ACC, Spurwechsel-Assistenten oder auch den Spurhalte-Assistenten LKA (siehe Abbildung 1.1). Darüber hinaus existieren aber auch Beispiele, die für den Fahrer weniger offensichtlich sind, wie der HBA, ESC, Airbag-Systeme, (ir-)reversible Gurtstraffer, usw. Dies resultiert daraus, dass diese Systeme meist erst bei nicht alltäglichen und insbesondere kritischen Situationen wie bspw. bei Unfällen zum Einsatz kommen. Allein diese Auswahl macht deutlich, dass durch die Anzahl der Systeme schon ein gewisser Grad an Komplexität vorliegt, den es für jedes Fahrzeug zu beherrschen gilt.

Parallel dazu wird in der Forschungswelt zunehmend der Begriff "Cyber-Physical System" (CPS) im Kontext von Hard- und Softwaresystemen bzw. eingebetteten Systemen verwendet, die unmittelbar eine Schnittstelle zur physikalischen Umwelt aufweisen. Teilweise werden beide Begriffe synonym verwendet, obwohl die wissenschaftliche Diskussion und insbesondere ihre inhaltliche Abgrenzung dazu noch nicht abgeschlossen ist. Letztlich wird den CPS jedoch eine große Bedeutung für zukünftige Systeme zugerechnet. Eine zugehörige Definition von Lee lautet:

> *Cyber-Physical Systems (CPS) are integrations of computation and physical processes. [Lee08, S. 1]*

Marwedel spezifiziert ein CPS insbesondere dadurch, dass mehrere eingebettete Systeme durch ihr Zusammenspiel über Nutzungsschnittstellen eine zusätzliche Dimension in Form lokaler oder globaler Vernetzung erhalten. Auch die Multifunktionalität durch Integration verschiedener Funktionen wird als Charakteristikum für CPS identifiziert. Im Übrigen gleichen sich die Merkmale jedoch stark zwischen den eingebetteten und den Cyber-Physical-Systemen, welches durch die Abbildung 1.2 zusätzlich unterstrichen wird. [Mar11, vgl. S. 21]

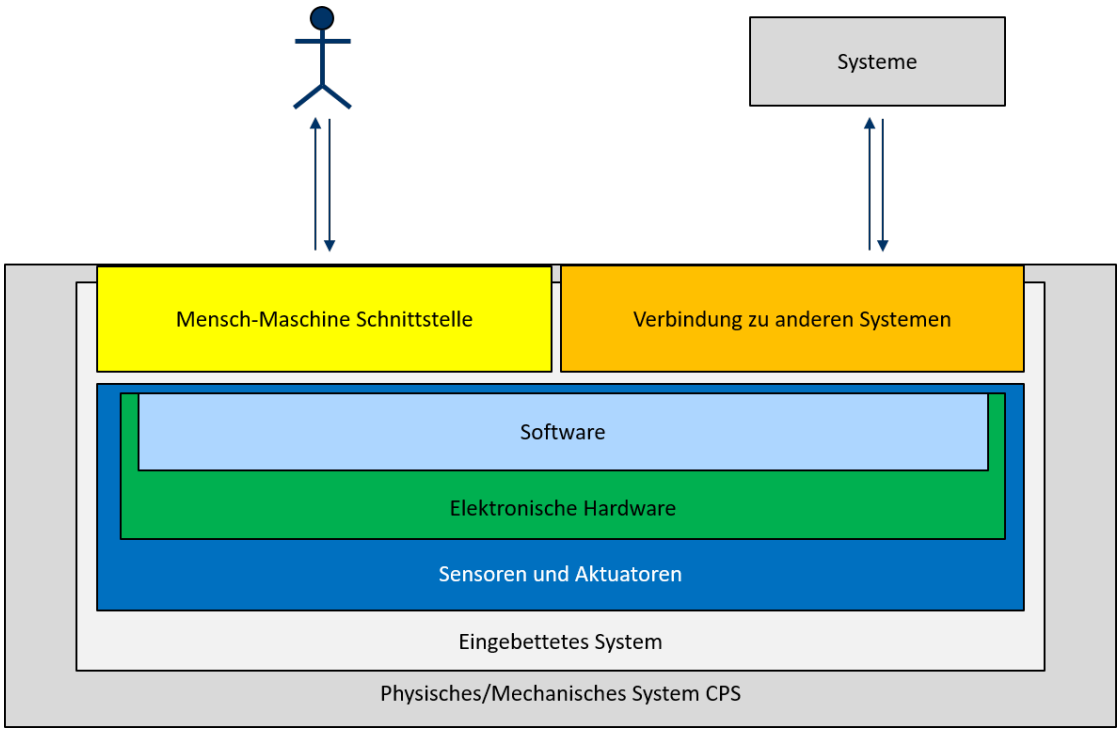


Abbildung 1.2: Zwiebelschalenstruktur der CPS in Anlehnung an [Bro10, S. 24]

Im Zuge der Entwicklung solcher Systeme in der Automotive-Domäne und dort speziell in der Fahrzeugsicherheit ergeben sich neue Herausforderungen und Problemstellungen speziell in der Sicherung und weiteren Verbesserung der Qualität der Systeme. Die schnell wachsende Vernetzung von eingebetteten Systemen bzw. CPS erhöhen deren Komplexitätsgrad und stellt gängige Methoden zum Test sowie der Absicherung auf den Prüfstand. Ein prinzipielles Überdenken ist aufgrund der steigenden Komplexität der jeweiligen Wirkfelder der Funktionen notwendig, da die Vielzahl der Szenarien rein reale Tests und Absicherungen nur mit einem vergleichsweise hohen Aufwand möglich macht und die Wirtschaftlichkeit der Entwicklung in Frage stellt.

Daher ist eine Etablierung von Simulationstechniken im Bereich der aktiven Sicherheit zur virtuellen Absicherung weiter voranzutreiben, um weitere Fortschritte hinsichtlich der Qualität erzielen zu können. Die passive Sicherheit nimmt hier eine Vorreiterrolle ein, die bereits diverse Simulationsmethoden wie Finite-Elemente- oder auch Mehrkörpersimulation in den Entwicklungsprozess integriert hat. Die Merkmale und Eigenschaften eines CPS erfordern jedoch eine andere Art der Simulation als es bisher in der passiven Sicherheit der Fall gewesen ist. So werden durch die Systeme überwiegend analoge und digitale Signale verarbeitet, die wiederum Informationen von einem (Teil-)System zum nächsten transportieren. Dabei kann es sich um z.T. komplexe Informationen handeln, die kontinuierlich über die Zeit übermittelt werden und einer hohen Änderungsrate über die Zeit verfügen. Dies kann in Bezug auf CPS auch neue Abstraktionskonzepte erfordern, wie bereits Lee in einem Artikel aufzeigte, in welchem er einen integralen Ansatz durch Zusammenführen der unterschiedlichen Modellierungstechniken für notwendig erklärte. [Lee08, vgl. S. 3] Somit müssen auch für solche Methoden und Techniken neue Test- und Absicherungsverfahren entwickelt bzw. angepasst werden.

Die Kombination von Komfort- und Sicherheitsaspekten in Fahrerassistenzsystemen sorgt dafür, dass die Wirkfelderweiterung mit einer deutlichen Zunahme von Szenarien und Parametern verbunden ist, die durch das System verarbeitet werden müssen. In der Vergangenheit hat sich der Entwicklungsprozess bei OEMs zunehmend standardisiert, um dem Qualitätsanspruch von Entwicklern und späteren Kunden an die Fahrerassistenzsysteme Rechnung zu tragen. Dazu wurden eine Reihe von Methoden und Best-Practices zur Verifikation und Validierung etabliert, um die jeweilige Funktion als Teil eines Systems serienreif in das entsprechende Fahrzeug zu bringen. Neben den üblichen Test- und Erprobungsfahrten mit realen Fahrzeugen wurden auch eine Reihe von Prüfständen realisiert, mit denen die Systeme außerhalb des Fahrzeugs im Verbund für besondere Fälle getestet werden.

1.2 Zusammenfassung der Problemstellung im Projekt

Die Entwicklung von Fahrerassistenzsystemen mit einem Schwerpunkt auf Sicherheitsaspekte (im Folgenden “Aktive Sicherheitssysteme”) ist momentan durch zwei wesentliche Entwicklungsleitlinien geprägt:

1. ein möglichst logisches und “fehlerfreies” Verhalten des Systems vor Kunde zu realisieren und
2. das Verhalten des Systems im Hinblick auf ausgewählte und standardisierte Consumer-Test-Szenarien zu optimieren.

Es kann hier gewiss als Argument angeführt werden, dass die Szenarien bei den Assessments im Rahmen von Consumer Tests wie EuroNCAP aus dem eigentlichen Einsatzfeld abgeleitet sind und einen Repräsentanten darstellen. Allerdings führt die Standardisierung und genaue Spezifikation zur Sicherung der Vergleichbarkeit zwischen den Fahrzeugen verschiedener Hersteller zu einem klinischen Testaufbau mit Laborcharakter. Die Einsatzszenarien im späteren Feld werden zwar sehr ähnlich sein, jedoch kann sich die konkrete Situation im Feld deutlich komplexer als bei den Consumer Tests darstellen, welches allein durch die Anzahl der umgebenden Objekte schon resultiert. Zudem kommt dem Kundenwunsch eine ganz wesentliche Bedeutung zu, da jeder Kunde eine bestimmte Erwartung an das System speziell im Vergleich zu seiner persönlichen Fahrweise hat und nur bedingt eine Übersteuerung durch das Fahrzeug akzeptiert. Speziell diese Gratwanderung ist die eigentliche Herausforderung zur Etablierung von aktiven Sicherheitssystemen, da das jeweilige System einerseits eine hohe Punktzahl in den Consumer Tests erreichen soll, gleichzeitig aber robust und möglichst spät im Feld eingreift. Dies gilt es somit während der Entwicklung in Einklang zu bringen.

Aus organisatorischen und organisationalen Gründen ist die Konzentration im vorliegenden Projektrahmen ausschließlich auf die zweite Entwicklungsleitlinie gerichtet. Hierzu werden die Anforderungen an das System speziell aus der Sicht des betreffenden Consumer Tests und die Testprozeduren definiert, während der Systementwicklung umgesetzt und anschließend durch momentan noch reale Versuche und Testläufe verifiziert und validiert.

Die Anforderungen an das jeweilige System sind abhängig vom jeweiligen Sicherheitsziel des Automobilherstellers. So ist für die Volkswagen AG grundsätzlich eine Bestbewertung seiner Fahrzeuge entscheidend. Speziell für EuroNCAP bedeutet dies eine Gesamtbewertung von fünf Sternen. Erwähnenswert hierbei ist jedoch, dass sogenannte Imbalance-Grenzen existieren, d.h., dass ein Fahrzeug in jeder Sicherheitssäule eine bestimmte Mindestpunktzahl erreichen muss, um insgesamt die Bestnote zu erhalten. Andernfalls wird nur die nächst schlechtere Stufe erreicht, selbst wenn die Gesamtpunktzahl oberhalb der fünf Sterne Grenze liegt. Damit ist es nicht ausreichend, z.B. in einer Säule die maximale Punktzahl zu erreichen und im Gegensatz dazu in einer anderen Säule ein deutlich schlechteres Ergebnis in Kauf zu nehmen.

Für diese Arbeit ist wichtig, die seit Beginn 2014 eingeführten Bewertungsverfahren in Bezug auf Aktive Sicherheitssysteme genauer zu betrachten. Die Bestnote erhält man dadurch, dass man mindestens ein modernes, aktives Sicherheitssystem wie Notbremssysteme, Spurhalteassistenten und Geschwindigkeitsbegrenzer mit einer entsprechenden Einbaurate im Fahrzeug integriert. Die Gewichtung der einzelnen Tests ist vergleichsweise komplex und wird auch in Kapitel 3 näher erläutert.

Während der Speedlimiter und der Spurhalteassistent aufgrund ihrer Testspezifikation bei EuroNCAP noch vergleichsweise einfach real getestet werden können, sind die Anforderungen an die Notbremssysteme deutlich komplexer, so dass sich die Nutzung eines simulativen Ansatzes eine mögliche Methode darstellt, um neben den realen Versuchs- und Testfahrten zusätzliche Erkenntnisse über die Qualität zu erhalten. Auch die Testspezifikation ist im Vergleich zu den anderen beiden Sicherheitssystemen umfangreicher, so dass trotz der genauen Randbedingungen die gleichen Testläufe aufgrund der Toleranzbereiche von Testparametern der Art voneinander unterscheiden und zu unterschiedlichen Bewertungen kommen.

Daher liegt die Konzentration auf einem Notbremsassistenten, der die Umgebung auf mögliche Kollisionen mit anderen Objekten hin bewertet und im Falle einer drohenden Kollision eine Notbremsung einleitet. Die jeweiligen Consumer Test Szenarien stammen dabei von den CTOs EuroNCAP und dem USNCAP . Eine genaue Darstellung der Testspezifikation für die Warn- und Notbremssysteme im Rahmen von AEB City und AEB Interurban sowie die Szenarien beim USNCAP erfolgen in Kapitel 3.

Das Ziel dieser Arbeit besteht darin, den ohnehin komplexen Entwicklungsprozess von aktiven Sicherheitssystemen im Rahmen Consumer Tests durch simulative Ansätze zu unterstützen, um einerseits die Qualität der eingebetteten Systeme zu verbessern, andererseits jedoch auch die Entwicklung effektiver zu gestalten, wie z.B. durch gezielte Verhaltensuntersuchungen von Algorithmen zur verbesserten Planung von Ressourcen während der realen Tests.

1.3 Forschungsfragen

Aufgrund dieses Projektkontextes ergeben sich folgende Forschungsfragen, die im Rahmen dieser Arbeit bearbeitet werden:

- RQ1: Welche Methoden zur systematischen Entwicklung einer Simulationsinfrastruktur für Aktive Sicherheitssysteme in Consumer Tests existieren gegenwärtig?
- RQ2: Wie kann die Entwicklung einer Simulationsumgebung für Aktive Sicherheitssysteme systematisiert und dadurch effizienter in Bezug auf die Ausarbeitung von Werkzeugen gestaltet werden?
- RQ3: Welche Erkenntnisse lassen sich simulativ bei der Bewertung von Aktiven Sicherheitssystemen im Rahmen von Consumer Tests erzielen?

Die erste Frage zielt darauf ab, grundsätzlich festzustellen, welche Methoden zur Entwicklung von Simulationsumgebungen im Kontext von Aktiven Sicherheitssystemen und Consumer Tests

beschrieben wurden. Sofern sie existieren, gilt es die wesentlichen Grundzüge der Ansätze darzustellen und auf ihre Tauglichkeit im Projektrahmen einzugehen. Im Falle, dass zur Zeit noch keine konkreten Methoden existieren bzw. systematisch festgehalten worden sind, wird ein eigener Ansatz hier entwickelt und beschrieben, um auf die zweite Frage näher einzugehen. Mit der dritten Frage wird die Praxistauglichkeit der beschriebenen Methode näher untersucht, um den Nutzen für die Serienentwicklung aktiver Sicherheitssysteme zu dokumentieren.

1.4 Ergebnisse

Die Ergebnisse dieser umfassen zunächst die Analyse der Projektsituation im Rahmen eines industriellen Kontextes und die Unterschiede zwischen reinen Forschungsprojekten und Projekten aus der Serienentwicklung. Dabei konnte aufgezeigt werden, dass es eine Reihe von Simulationsaktivitäten gibt, jedoch eine strukturierte Herangehensweise zu ihrer Entwicklung fehlt. Mit dieser Erkenntnis reifte die Überlegung, dass eine Methodik zur Entwicklung von Simulationsumgebung und eine Systematisierung der Arbeiten einen Ansatz bilden, der unter den gegebenen Randbedingungen so noch nicht ausgearbeitet wurde. Zur Absicherung dieser Erkenntnis wurde eine systematische Literatur Recherche (*engl.: Systematic Literature Review, kurz: SLR)*[KC07] in Kapitel 4 durchgeführt und evtl. Methodiken daraufhin untersucht. Als Ergebnis wurde festgehalten, dass es keine konkrete Methodik weder im Kontext von Consumer Tests noch in einem erweiterten Kontext publiziert wurde. Zwar werden Simulationsumgebungen im Fahrerassistenzumfeld beschrieben und ihr Nutzen herausgearbeitet. Es fehlt jedoch an einer systematischen Entwicklungsmethodik, die sich mit der zentralen Frage bei Simulationsumgebungen befasst: Wie kann eine Simulationsumgebung zielgerichtet entwickelt werden, ohne sich im Detail zu verlieren? Daher wurde eine eigene Methodik zur Entwicklung von Simulationsumgebungen ausgearbeitet und vorgestellt. Diese umfasst vier Bausteine:

1. Analyse des Untersuchungsraums in Kapitel 6
2. Entwicklung einer adäquaten Simulationsumgebung in Kapitel 7
3. die Entwicklung von Bewertungsverfahren in Kapitel 8
4. Durchführung von Simulationsläufen in Kapitel 9

Anhand einer Fallstudie wird hierfür ein Proof of Concept zur Methodik gezeigt.

1.5 Aufbau

In Kapitel 2 werden die für das Verständnis notwendigen Grundlagen beschrieben. Hierzu gehören unter anderem die Kunst des (Software-)Testens, Modellbildung und Simulation im Allgemeinen, diverse Grundlagen der Fahrzeugtechnik sowie die Rolle von Hard- und Software im Fahrzeug. Weiterhin werden einige Grundlagen zum Automotive Software Engineering und zu Integralen Sicherheitssystemen vermittelt. In Kapitel 3 werden aktuelle Ansätze im Projektrahmen erläutert und die Besonderheit der CTOs herausgestellt. Aktuelle Ansätze dazu bzw. im näheren Umfeld werden im Kapitel 4 im Rahmen eines Systematic Literature Reviews (SLR) identifiziert und analysiert. Ausgehend von diesem Review und dem Ergebnis wird dann eine Methode zur agilen Entwicklung von Simulationsumgebungen im Kapitel 5 zusammenfassend beschrieben, die aus vier Bausteinen besteht. Kapitel 6 geht detaillierter auf den ersten Baustein zur Analyse des Untersuchungsraumes ein, während Kapitel 7 aufzeigt, wie die technischen Aspekte zur Schaffung einer Infrastruktur realisiert werden. In Kapitel 8 werden mögliche Bewertungsverfahren herausgearbeitet und mit Kapitel 9 abgeschlossen, in welchem Hinweise zur Durchführung von Simulationsstudien und ihrer Auswertung gegeben werden. Diese Arbeit wird anschließend in Kapitel 10 kritisch betrachtet. Kapitel 11 fasst die wesentlichen Aussagen dieser Arbeit durch Beantwortung der Forschungsfragen zusammen und gibt einen Ausblick auf weitere Themen.

1.6 Vorveröffentlichungen

Im Zuge dieser Arbeit sind in einigen Vorveröffentlichungen und Forschungsbeiträge entstanden, die einige Aspekte dieser Arbeit zusammenfassen. Nachfolgend ist eine Liste der Beiträge aufgeführt:

AAET'2013 In diesem Beitrag skizzieren die Autoren die Methode der Meta-Metriken als Teil der heutigen Herausforderungen im Software Engineering für CPS. Sie sollen vornehmlich dazu dienen, eine Methodik zur Überwachung des Entwicklungsprozesses von Aktiven Sicherheitssystemen zu überwachen. Dabei wird insbesondere die zeitliche Entwicklung von Metriken betrachtet, die während des Simulationslaufes von Testfällen zur Absicherung von Aktiven Sicherheitssystemen und -funktionen erhoben wurden. Mit Hilfe der Entwicklung geeigneter (Meta-)Metriken können mögliche Schwachstellen eines Systems oder eine Komponente identifiziert bzw. die Konzentration weiterer Testläufe auf bestimmte Testszenarien gelenkt werden. [BBH+13]

es4cps'2014: Dieser Beitrag zeigt die Herausforderungen für die Serienentwicklung auf, die im Zuge der Berücksichtigung der neuen Anforderungen an die Consumer Tests wie z.B. von EuroNCAP entstanden sind, da ein einfacher Äquivalenzklassentest für solche Systeme nur bedingt ausreichend sind. Außerdem zeigt das Papier auf, welche Möglichkeiten virtuelle Testansätze während der Entwicklung von Aktiven Sicherheitssystemen bieten können. Zudem wird ein Ansatz bzw. die Notwendigkeit einer Methodik erläutert, wie eine Simulationsumgebung effektiv und zielgerichtet entwickelt werden kann, damit sie die Entwicklung auch unterstützt. Darauf aufbauend wird eine erste technische Umsetzung der Simulationsumgebung präsentiert. [BHK+14]

ITSC'2014: Dieses Papier beschäftigt sich aufbauend auf dem vorherig genannten mit zwei Forschungsfragen:

1. Wie können bestimmte Parameter innerhalb eines Toleranzbereiches modelliert und variiert werden, um verschiedene Trajektorien eines VUT auf ein stehendes Hindernis systematisch zu simulieren?
2. Welchen Einfluss haben verschiedene Fahrzeugpositionen und Gierwinkel auf einen AEB/FCW Algorithmus und damit auf die Restgeschwindigkeit eines VUT außerhalb der EuroNCAP Bedingungen?

Es konnte gezeigt werden, dass die Position des Fahrzeugs einen Einfluss auf die Restgeschwindigkeit haben kann und das der präsentierte Simulationsansatz die Entwicklung solcher aktiven Sicherheitssysteme insbesondere durch die effektive Ressourcenallokation unterstützen kann. [BBH+14a]

VDI'2014: In diesem Beitrag wurde das Prinzip der modellbasierten Generierung von Simulationsszenarien näher erläutert. Dabei wurde sich auf die Veränderung und die Variation des Geschwindigkeitsparameters konzentriert. Die folgenden zwei Forschungsfragen wurden dabei untersucht:

1. Wie kann der Geschwindigkeitsparameter systematisch innerhalb des Toleranzbandes beim Test auf ein stehendes Hindernis modelliert werden und welche Auswirkungen hat dies auf den Auslösezeitpunkt?
2. Wie wirkt es sich im Vergleich zu einer konstanten Fahrt auf die Restgeschwindigkeit aus, wenn um einen möglichen Auslösezeitpunkt die Geschwindigkeit erhöht wird?

Dabei konnte beispielhaft anhand eines Black-Box Algorithmus verschiedene Anomalien im Verhalten aufgezeigt werden, so dass der Testprozess in der Realität effizienter

geplant werden kann, da die Konzentration dann auf solche Anomalien gelenkt werden kann. [BBH+14b]

ITSC'2015: Dieser Beitrag befasst sich mit den folgenden zwei Forschungsfragen:

1. Welche Varianten innerhalb einer Spanne von 1000 Simulationsläufen können für Consumer Test Szenarios modelliert, ausgeführt und ausgewertet werden?
2. Welchen Einfluss auf haben diverse Variationen von Consumer Test Randbedingungen auf das Auslöseverhalten eines Aktiven Sicherheitsalgorithmus?

Der Beitrag konnte die strukturierte Analyse von mehr als 3000 modellbasierten Consumer Test Szenarien aus dem EuroNCAP und USNCAP Protokoll aufzeigen. Die Ergebnisse umfassten mehr als 7,7 GB an Ergebnisdaten und war zum damaligen Zeitpunkt die erste in einem industriellen Kontext. [BBH+15b]

ITSM'2015: Dieser Artikel ist auf Einladung der Redaktion des IEEE Intelligent Transportation Systems Magazine entstanden und stellt eine Erweiterung zum Beitrag von der ITSC 2014 dar. Darin wurde der Artikel noch einmal detaillierter ausgearbeitet. [BBH+15a]

2 Grundlagen

In diesem Kapitel werden di e Grundlagen zur Bearbeitung der Forschungsfragen dargestellt. Dies umfasst zum einen den Begriff des Testens im Allgemeinen, aber auch speziell im Bereich der Fahrzeugentwicklung. Jener wird der Begriff der Simulation gegenüber gestellt, der aus Sicht des Autors allzu leichtfertig synonym zum Begriff des Testens verwendet wird. Anschließend werden allgemeine Grundlagen der Fahrzeugtechnik behandelt. Danach wird auf die grundsätzlichen Rahmenbedingungen des Projektes eingegangen, anhand derer eine Reihe von Anforderungen an das zu entwickelnde System gestellt werden. Sie werden in einem abschließenden Unterkapitel behandelt.

2.1 Die Fundamente des Testens

Die Entwicklung von Produkten in der heutigen Zeit ist zu einem hochkomplexen Prozess herangereift, da die stetige Innovationskraft durch die eingesetzte und zu entwickelnde Technologie die Komplexität der Produkte auf immer höhere Ebenen befördert. In Verbindung mit der Verkürzung von Lebenszyklen der Produkte wird die Qualitätssicherung dieser ebenfalls vor neue Herausforderungen gestellt und muss in der gleichen Intensität hinsichtlich Reifegrad und Komplexität mit der eigentlichen Funktionalität und Technologie des jeweiligen Produktes Schritt halten; andernfalls kann ein hinreichendes Maß an Zuverlässigkeit des Produktes im Sinne des Kunden nicht gewährleistet werden. Das gilt im gleichen Maße für Toaster wie für moderne (Premium-)Fahrzeuge, wobei die Fehlertoleranz von Kunden allgemein mit steigendem Preis abnimmt.

Welche Konsequenzen Fehler insbesondere bei sicherheitskritischen Systemen verursachen können, zeigen unter anderen die Beispiele des Bestrahlungsgerätes “Therac-25” im Jahr 1985/86 oder der Jungfernflug der “Ariane 5” im Jahr 1996, in denen Softwarefehler schwerwiegende Auswirkungen hatten: Während bei der Ariane 5 aufgrund eines Überlaufs einer Zahlkonvertierung zur Lageregelung die Sprengung herbeigeführt werden musste und somit nur Kapital vernichtet wurde, kamen wegen diverser Softwarefehler u.a. in der Benutzerschnittstelle mehrere

Menschen um ihr Leben, weil diese Fehler falsche Einstellungen zur Bestrahlung verursachten und die Patienten eine Überdosis erlitten. [Sch06]

Softwarefehler müssen jedoch nicht zwangsläufig in solchen dramatischen Konsequenzen enden, dennoch beeinträchtigen Softwarefehler bzw. Entwicklungsfehler nicht selten unerheblich die Qualität eines Produktes. Die Sicherung eines gewissen Niveaus an Qualität sollen standardisierte Prozess- und Vorgehensmodelle sicherstellen, die beschreiben, zu welchem Zeitpunkt welche Aufgabe von welcher Person oder System auf welche Weise wahrgenommen werden soll. Die Feinheit der Aufgliederung solcher Vorgehensmodelle hängt dabei von der Konkretisierung und der Komplexität des tatsächlichen Produktes ab. Unter dem Begriff des Testens ist zunächst folgendes zu verstehen:

> *Unter dem Testen von Software [selbst] wird jede (im Allgemeinen stichprobenartige) Ausführung eines Testobjektes verstanden, die der Überprüfung des Testobjektes dient. [SL07, S. 8]*

Dabei verfolgt der Prozess des Testens mehrere allgemeine Ziele, die letztlich von der einzelnen Teststufe betrachtet werden können:

- Fehlerwirkungen nachzuweisen,
- die Qualität zu bestimmen,
- Vertrauen in das Produkt zu erhöhen,
- durch Analyse Fehlerwirkungen vorzubeugen. [SL07]

Insgesamt nimmt der Testprozess selbst einen deutlichen Anteil an den Gesamtentwicklungsumfängen eines Produktes ein, wie anhand von Entwicklungsmodellen deutlich wird. Im Folgenden soll daher das allgemeine Entwicklungsprozess-Modell im Fahrzeugbau beschrieben werden.

2.1.1 Das V-Modell als etablierter Entwicklungsprozess im Automobilbau

Prägend für die technische Entwicklung sowohl einzelner Fahrzeugfunktionen als auch von Gesamtfahrzeugprojekten ist im Automobilbau der Ansatz einer funktionsorientierten Entwicklung, der jedoch nicht immer konsequent durchgehalten wird. Konkret gliedert sich dieser Ansatz in den Entwicklungsprozess zum Produkt “Fahrzeug” ein, der über einen festen Zeitraum die zur vollständigen Realisierung eines Fahrzeugprojektes erforderlichen Meilensteine bis zum SOP beschreibt. Dabei können drei Abstraktionsebenen unterschieden werden:

- Die technisch unabhängige Entwicklung der Funktion,

- die Umsetzung der Funktion in ein System für ein bestimmtes Fahrzeug und
- die Entwicklung der einzelnen Komponenten eines Fahrzeuges.

Eine Erläuterung dieser Ebenen wird nachfolgend in Kapitel 2.3 fortgesetzt. [SZ10, GSSZ13, S. 141]

Das allgemeine V-Modell adressiert diese Abstraktionsebenen durch unterschiedliche Stufen und mit jeder weiteren Stufe erfolgt eine Detaillierung der Entwicklungsaktivitäten. Ein grundlegendes Konzept des V-Modells ist dabei, dass Entwicklungs- und Testaktivitäten gleichberechtigt und korrespondierend als Teil des Entwicklungsprozesses betrachtet werden. Aufgespaltet und entsprechend als "V" angeordnet ergibt sich die prägende Form für dieses Vorgehensmodell zur Produktentwicklung. [SL07] Durch die sukzessive Detaillierung erhält das Modell auch eine zeitliche Komponente, die im Automobilbau in drei aufeinander folgende Phasen gegliedert werden kann:

- Konzept- und Planungsphase
- Realisierungsphase
- Anlaufphase.

Die Konzept- und Planungsphase ist insbesondere durch die Definition von Anforderungen an das Produkt bzw. System und die Erstellung einer jeweiligen Spezifikation gekennzeichnet. Die einzelnen Abstraktionsebenen verfeinern zudem die Anforderungen auf die einzelnen Komponenten, so dass die Implementierung während der Realisierungsphase erfolgen kann. Grundsätzlich können folgende Aktivitäten auf der linken Seite des Modells nach Spillner et al. festgehalten werden:

- Anforderungsdefinition (aus Kundensicht)
- Funktionaler (logischer) Systementwurf
- Technischer Systementwurf
- Komponenten-Spezifikation
- Programmierung (oder auch Implementierung).

Während der Anforderungsdefinition werden die Wünsche und die Erwartungshaltung des Kunden oder auch des Auftragsgebers an das spätere Produkt festgehalten. Während des funktionalen Systementwurfs werden die Anforderungen in Funktionen zusammengefasst und als logisches System beschrieben. Der technische Systementwurf legt fest, welche Funktionen sich

auf welches (Teil-)system beziehen und wie das logische System technisch realisiert werden kann. Die Komponenten-Spezifikation bricht dieses System dann auf die einzelnen Systemkomponenten herunter und definiert die einzelnen Anforderungen und Schnittstellen. Im Rahmen der Implementierung werden die Anforderungen dann umgesetzt. [SL07]

Korrespondierend und gleichberechtigt zu den vorherigen Entwicklungsaktivitäten sind die Testaktivitäten auf der rechten Seite des V-Modells. Diese umfassen folgende Aktivitäten im Detail in zeitlicher Reihenfolge:

- Komponententest
- Integrationstest
- Systemtest
- Abnahmetest (oder: Akzeptanztest)

Die einzelnen Teststufen weisen ebenfalls unterschiedliche Aktivitäten auf. So wird im Komponententest eine einzelne Komponente oder ein Bauteil getestet, während beim Integrationstest weitere Komponenten hinzugenommen werden, um auch das Kommunikationsverhalten und Zusammenspiel näher zu untersuchen. Im Systemtest wird dann die Gesamtheit aller Komponenten überprüft, um schließlich im Abnahmetest die vollen vereinbarten Leistungsmerkmale nachzuweisen. [SL07, S. 40ff] Konkret auf die Automotive-Domäne schließt dies auch Freigaben mit ein, so dass Funktionen anschließend auch für Kunden in Serienfahrzeugen nutzbar sind. Insgesamt sorgen die Teststufen dafür, dass die spezifizierten Anforderungen sowohl korrekt umgesetzt sind als auch das geforderte Verhalten bzw. den gewünschten Effekt aufweisen. Das Schaubild 2.1 illustriert diesen Zusammenhang von Verifikation/Validierung und jeweiligen Tests auf den einzelnen Ebenen.

Die Anordnung der Teststufen zu den einzelnen Entwicklungsaktivitäten folgen dem Prinzip, dass Fehler idealerweise auch auf der Ebene entdeckt werden sollen, auf der sie letztlich auch entstanden sind. [SL07, S. 40ff]. Es wird zudem zwischen “Validierung” und “Verifikation” unterschieden:

> *Beim Validieren bewertet der Tester, ob ein (Teil-)Produkt eine festgelegte (spezifizierte) Aufgabe tatsächlich löst und deshalb für seinen Einsatz tauglich bzw. nützlich ist. Untersucht wird, ob das Produkt im Kontext der beabsichtigten Produktnutzung sinnvoll ist. [SL07, S. 41]*

Ergänzend hierzu die Definition zur Prüfung, ob die Anforderungen der nächsthöheren Abstraktionsstufe korrekt umgesetzt worden sind:

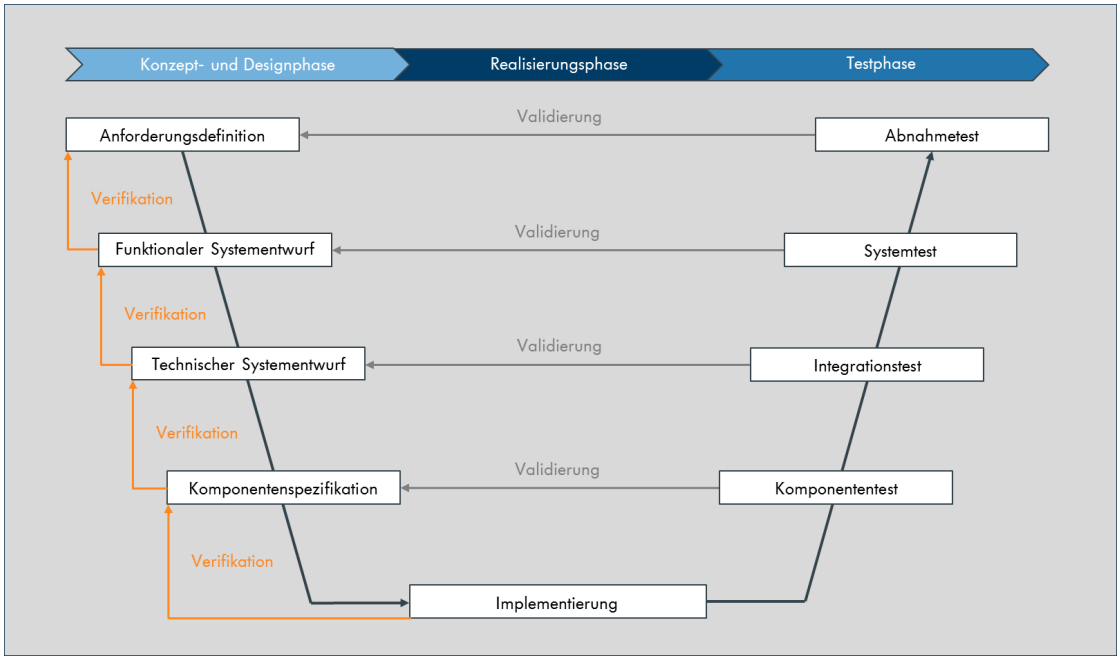


Abbildung 2.1: Verifikation und Validierung im V-Modell in Anlehnung an [SL07, S. 40]

> *Verifikation ist im Gegensatz zur Validierung auf eine einzelne Entwicklungsphase bezogen und soll die Korrektheit und Vollständigkeit eines Phasenergebnisses relativ zu seiner Spezifikation (Phaseneingangsdokumente) nachweisen. Untersucht wird, ob die Spezifikationen korrekt umgesetzt wurden, unabhängig von einem beabsichtigten Zweck oder Nutzen des Produktes. [SL07, S. 40]*

2.1.2 Der allgemeine Testprozess

Das International Software Testing Qualifications Board (ISTQB) hat im Rahmen seines Lehrplans einen fundamentalen Testprozess definiert, der aus mehreren Prozessschritten besteht. Die folgendes Abbildung illsturiert diesen Prozess:

Zu Beginn der Planung müssen die Zielsetzung und die Aufgaben definiert werden, so dass eine entsprechende Ressourcenplanung hinsichtlich Mitarbeiter und weiterer technischer Hilfsmittel, evtl. auch von Neuentwicklungen der Testinfrastruktur erfolgen kann. Die Steuerung des gesamten Testablaufes muss kontinuierlich über den gesamten Prozess sichergestellt werden, um angemessen auf Situationsveränderungen reagieren zu können. Während der Testanalyse und des Testdesigns werden die vorhandenen Eingangsdokumente auf Verwendbarkeit und De-

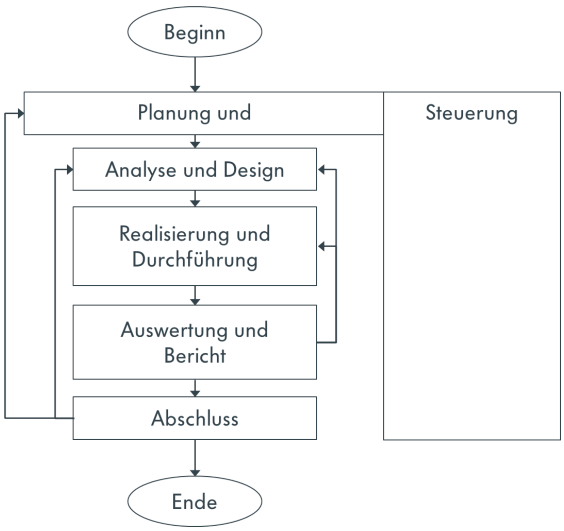


Abbildung 2.2: Der fundamentale Testprozess gemäß ISTQB in Anlehnung an [SL07, S. 20]

tailgrad untersucht, aus denen dann wiederum die logischen und konkreten Testfälle abgeleitet werden. Letztere werden dann während der Testrealisierung umgesetzt und anschließend auch durchgeführt. Hierbei ist es wichtig, dass die Ausführung der Testfälle protokolliert werden, um die Nachvollziehbarkeit des Testablaufes insbesondere für Außenstehende zu gewährleisten. Im Laufe der Testauswertung werden die vorliegenden Testergebnisse im Detail analysiert und entsprechende Fehlerwirkungen zur Behebung priorisiert. In diesem Zuge ist darauf zu achten, dass mögliche Fehlerwirkungen auch aufgrund einer falschen Testspezifikation oder der Testinfrastruktur basieren können. Daher ist eine genaue Analyse der Ergebnisse speziell auf diesen Aspekt zu untersuchen. Der Bericht fasst die wesentlichen Ergebnisse für weitere Entscheidungsträger zusammen. Zur weiteren Verbesserung des Testprozesses insgesamt sollten zum Abschluss der Testaktivitäten die gemachten Erfahrungen dokumentiert und evtl. Handlungsempfehlungen beschrieben werden, um bei einem weiteren Testablauf diese Erfahrungen einfließen zu lassen. [SL07, S. 20ff]

Die Abbildung 2.3 stellt die Einbindung des fundamentalen Testprozesses in das allgemeine V-Modell dar und orientiert sich dabei auch an einer V-Form. Während der Definition der Anforderungen auf der jeweiligen Abstraktionsebene werden parallel die nötigen Arbeitsschritte Testplanung, Testanalyse und -design sowie Testrealisierung vorgenommen. Ein fließender Übergang zur Durchführung, der Auswertung und Berichtsverfassung sowie des Testabschlusses erfolgt dann je nach Stand der Implementierung auf dem aufsteigenden, rechten Ast als Teil der zugehörigen Teststufe. Die Rückspiegelung zur Testanalyse während der Auswertung und analog zur Planung als Rückmeldung aus einem vorherigen Projektabschluss ist durch entspre-

chende Pfeile gekennzeichnet. Die Steuerungsaktivitäten werden hier implizit als übergeordnete Aufgabe angesehen.

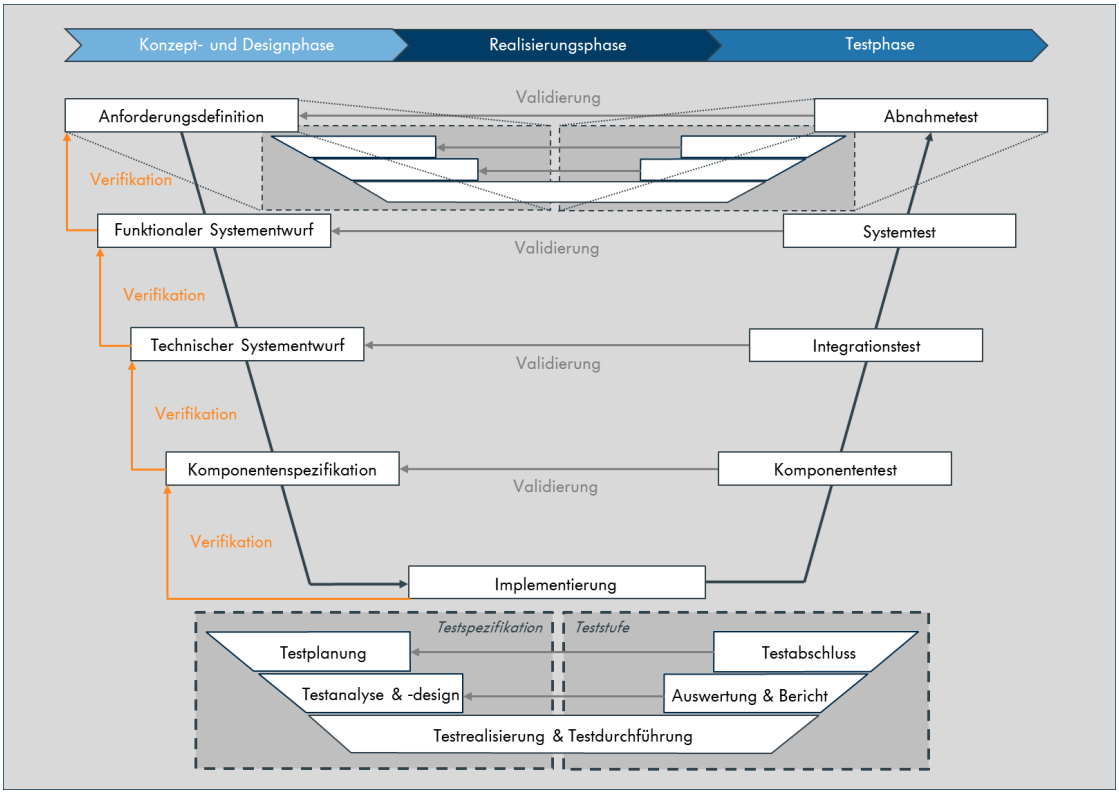


Abbildung 2.3: Das erweiterte V-Modell um den fundamentalen Testprozess (Eigene Darstellung)

2.2 Modellbildung und Simulation

Während der Entstehung dieser Arbeit ist häufig der Begriff “Test” und “Simulation” synonym verwendet worden. Daher wird zunächst die Bedeutung der jeweiligen Begriffe erläutert, um ein einheitliches Verständnis zu gewährleisten. Anschließend wird eine allgemeine kurze Einführung zur Modellbildung und Simulation gegeben, damit die wichtigsten Aspekte und Fragestellungen hierzu deutlich werden.

2.2.1 Der Begriff

In einem sehr weiten Sinne wird zunächst unter Simulation “ein möglichst realitätsnahes Nachbilden von Geschehen der Wirklichkeit” verstanden. Ziel dabei ist es, sich aus der Realität durch Modellbildung zu lösen und damit ein Problem auf eine höhere Abstraktionsebene zu heben. [Gab12, S. 1] Eine weitere Definition gemäß VDE-Richtlinie 3633 bezeichnet Simulation als “ein Verfahren zur Nachbildung - das bedeutet Modellbildung - eines realen oder gedachten Systems mit seinen internen dynamischen Prozessen in Form eines experimentierbaren Modells, um zu Erkenntnissen zu gelangen, die auf die Realität übertragbar sind.” [ITW12, S. 1] Wichtig dabei ist, dass die Wirkstrukturen und -mechanismen vollständig und korrekt modelliert werden müssen. Dies führt letztlich jedoch zu einer Abwägung zwischen der Vollständigkeit/Korrektheit der Wirkungen und der gewählten Abstraktion/Güte der Modelle.

Acker definiert den Begriff wie folgt: “Unter der Simulation eines Systems versteht man die Darstellung des Verhaltens eines physikalischen oder abstrakten Systems durch das Verhalten eines Ersatz-Systems (Modell), mit dessen Hilfe man Verhaltensstudien vereinfacht durchführen kann.” [Ack09, S. 16]

2.2.2 Der Prozess

Bungartz et al. sprechen im weiteren Sinne hinsichtlich Simulation nicht von einem integralen Akt, sondern von einem andauernden Prozess, bei dem mehrere Schritte ausgeführt werden müssen, die wiederum eine Rückkopplung zu vorherigen aufweisen, so dass der Prozess häufig mehrfach durchlaufen wird. [BZBP13, S. 2ff] Hier wird auch der Begriff der Simulationspipeline geprägt, wie die Abbildung 2.4 illustriert.

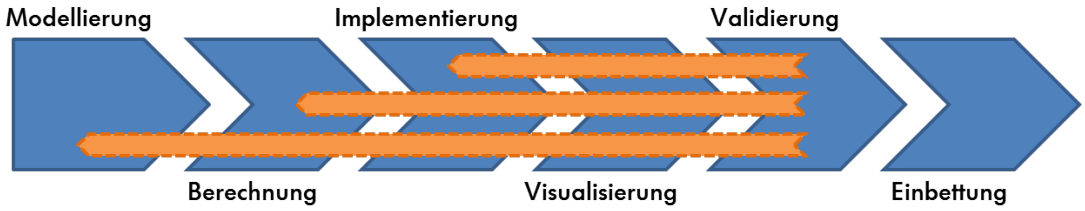


Abbildung 2.4: Die Simulationspipeline in Anlehnung an Bungartz et al., 2013 [BZBP13]

Die grundlegenden Bausteine sind wie folgt beschrieben:

- Modellierung: Der Ausschnitt des Betrachtungsgegenstandes gilt es in ein Modell zu überführen.

- Berechnung (Simulation im engeren Sinne): Das Modell wird so aufbereitet, dass es auf einem Rechner verarbeitet werden kann; ggf. müssen zur Lösung des Modells Algorithmen eingesetzt werden.
- Implementierung: Überführung des Modells und der Berechnung auf die vorgesehene Zielplattform. Heutzutage reicht nicht mehr nur die Implementierung als performantem Code aus, sondern es muss unter Verwendung der Methoden des Software Engineering komplexe Softwaresysteme entworfen und entwickelt werden.
- Visualisierung: Aus den gewonnenen Daten der Simulationsläufe müssen die relevanten Daten gewonnen werden.
- Validierung: Die Ergebnisse müssen auf ihre Verlässlichkeit untersucht werden. Daher müssen Vergleichsdaten herangezogen werden, um mögliche Fehler im Modell, der Infrastruktur oder in der Interpretation der Ergebnisse auszuschließen.
- Einbettung: Aufgrund eines Entwicklungsprozesses ist die Simulation in einen Projektkontext eingebunden, so dass die Simulation in Form eines geeigneten Werkzeugs in den bestehenden Prozess zu integrieren ist. [BZBP13]

Die sechs aufgeführten Schritte sind nicht losgelöst voneinander zu sehen. Insbesondere durch die Validierung und der Identifikation von möglichen falschen Annahmen bzw. anderen Arten von Fehlern im Modell ergeben sich Rückkopplungen, welche dann eine erneutes Durchlaufen der Simulationspipeline erfordert.

2.2.3 Die Modellierung

Als Modell wird allgemein ein Abbild einer Realität verstanden, das mehr oder weniger die Zusammenhänge und Effekte vereinfachend wiedergibt bzw. das einen größeren oder kleineren Ausschnitt der Wirklichkeit repräsentiert. Durch die formale Beschreibung wird dieses Abbild mit Methoden der Mathematik oder der Informatik abstrahiert. Bungartz stellt drei zentrale Fragen in den Mittelpunkt der Modellierung:

1. Auf welche Art gelangt man zu einem geeigneten Modell?
2. Welche Beschreibungsmöglichkeiten existieren dafür?
3. Wie ist nach der Herleitung des Modells anschließend zu beurteilen?

Die wichtigste Aufgabe zu Beginn des Prozesses ist zunächst, sich darüber bewusst zu werden, was genau modelliert und simuliert werden soll. Es ist entscheidend, möglichst detailliert

eine Vorstellung davon zu gewinnen, welches Ergebnis bzw. Erkenntnisgewinn am Ende der Simulation erzielt werden soll. Eng verknüpft damit ist die konkrete Formulierung der Aufgabenstellung: Sollen z.B. genau eine oder mehrere Lösungen gefunden werden? Oder eine bestimmte Lösung? Soll ein bestimmter Bereich untersucht werden? Dabei gilt es festzulegen, welche Größen qualitativ für den Sachverhalt von Bedeutung sind und wie quantitativ ihr Einfluss zu bewerten ist. Weiterhin sind die Beziehungen der einzelnen Größen untereinander qualitativ (logisch) zu identifizieren und ebenfalls quantitativ zu erfassen, bspw. durch entsprechende Faktoren. [BZBP13]

Zur Modellierung von Sachverhalten können zum Beispiel Konzepte, Methoden und Werkzeuge von Rumpe [Rum11, Rum12, Rum16, Rum17] genutzt werden. Weitere Möglichkeiten der Modellierung sind in [HRR12] und auch in [RRW14] zu finden. Eine weitere Erläuterung dieser Artefakte erfolgt u.a. in Abschnitt 2.5.2.

Bei der Analyse bzw. der Bewertung des hergeleiteten Modells gilt es drei Aspekte zu untersuchen: Die Lösbarkeit, die Eindeutigkeit und die Stetigkeit der Eingabedaten. Zunächst ist zu klären, wie sich die Lösung des Modells gestaltet. Existiert beispielsweise ein bestimmter Grenzwert, der zum Modell konvergiert oder existiert ein Maximum bzw. ein Minimum, wenn es sich um ein Optimierungsproblem handelt. Im Falle der Lösbarkeit schließt sich die Eindeutigkeit der Lösung an. Der dritte Aspekt betrachtet die Stetigkeit der Lösung: Wie sensitiv ist das Modell bei Änderung der Modellparameter? Oder verhält sich die Lösung unerwartet bei kleinen Änderungen der Eingabedaten?

Bei der Skalierbarkeit eines Problems bzw. eines Modells muss zwischen der Komplexität bzw. dem Aufwand und der Genauigkeit abgewogen werden. Je mehr Details und Effekte des Originals in das Modell einfließen, desto präziser können die jeweiligen Resultate ausfallen, allerdings auch zu Lasten eines höheren Simulationsaufwandes. Bungartz spricht in diesem Zusammenhang auch vom “passenden” Modell in Abgrenzung zum korrekten Modell. [BZBP13]

Im Rahmen der Validierung kann eine solche Bewertung erfolgen. Hier gibt es auch verschiedene Möglichkeiten, die Simulationsergebnisse einzuordnen: Der Abgleich mit experimentellen Untersuchungen, die A-posteriori Beobachtungen oder Plausibilitätstests. Bei Experimenten wird versucht, die Bedingungen der Simulation 1:1 oder durch Skalierung nachzubilden und die Ergebnisse zwischen Simulation und Versuch miteinander zu vergleichen. Aus Machbarkeits- oder Aufwandsgründen ist dies jedoch nicht immer zu gewährleisten; auch bei der Schaffung der gleichen Bedingungen können bereits Fehler entstehen, die einen Vergleich erschweren. Bei den A-posteriori Beobachtungen werden im Anschluss an die Simulation die real eingetretenen Ergebnisse der Vorhersage gegenübergestellt. Dabei muss dann bestimmt werden, inwiefern

die Resultate zufrieden stellen. Mit Hilfe von Plausibilitätstests werden die erzielten Simulationsergebnisse mit bisher als bestätigt kennzeichneten Theorien verglichen und auf mögliche Widersprüche überprüft. Grundsätzlich gilt es jedoch festzuhalten, dass Validierungsvergleiche hoch fehleranfällig sind und die Ergebnisse damit nur bedingt vergleichbar sind. [BZBP13]

2.3 Grundlagen der Fahrzeugtechnik

2.3.1 Die funktionsorientierte Entwicklung in der Automobilbranche

Zu einem früheren Zeitpunkt der Automobilentwicklung war der Prozess nach Komponenten bzw. Baugruppen und Bauteilen, allgemein auch als komponentenorientiert strukturiert. Dies bedeutet, dass typische Bereiche eines Fahrzeug wie Karosserie, Innenraum, Fahrwerk usw. sich jeweils auf ihre Kernkomponenten konzentrieren und diese in Baugruppen und Bauteile aufteilen, welches auch die jeweiligen Kompetenzen einschließt. Diese Art der Organisation wurde lange Zeit als optimal angesehen, da auch mit Einzug der Elektronik viele Funktionen, die vormals von mechanischen Bauteilen realisiert wurden, für sich abgegrenzt werden konnten. [KAB13]

Aufgrund der zunehmenden Vernetzung von Steuergeräten wurde jedoch diese Art der Organisation als nicht mehr ausreichend angesehen, da man durch leistungsfähigere Hard- und Software auch bereichsübergreifende Funktionen entwickeln konnte. Daher wurde das Konzept der funktionsorientierten Entwicklung bei dem Automobilherstellern für eine verbesserte, bereichsübergreifende Zusammenarbeit eingeführt, die eine andere Sichtweise auf die Funktionen erlaubt und damit Raum gibt für erweiterte und neue Methoden hinsichtlich Spezifikation, Modellierung, Verifikation und Validierung. [KAB13]

Es können bei der funktionsorientierten Entwicklung grundsätzlich drei Abstraktionsebenen unterschieden werden:

1. Die technisch unabhängige Entwicklung der Funktion,
2. die Umsetzung der Funktion in ein System für ein bestimmtes Fahrzeug und
3. die Entwicklung der Bauteile eines Fahrzeuges.

Dabei werden auch bestimmte Verantwortlichkeiten mit bestimmten Aufgaben adressiert. Die Abbildung 2.5 in Anlehnung von [GSSZ13] zeigt am Beispiel des V-Modells, auf welcher Ebene die jeweiligen Verantwortlichkeiten liegen. Exemplarisch ist eine abstrakte Sicherheitsfunktion angefügt.

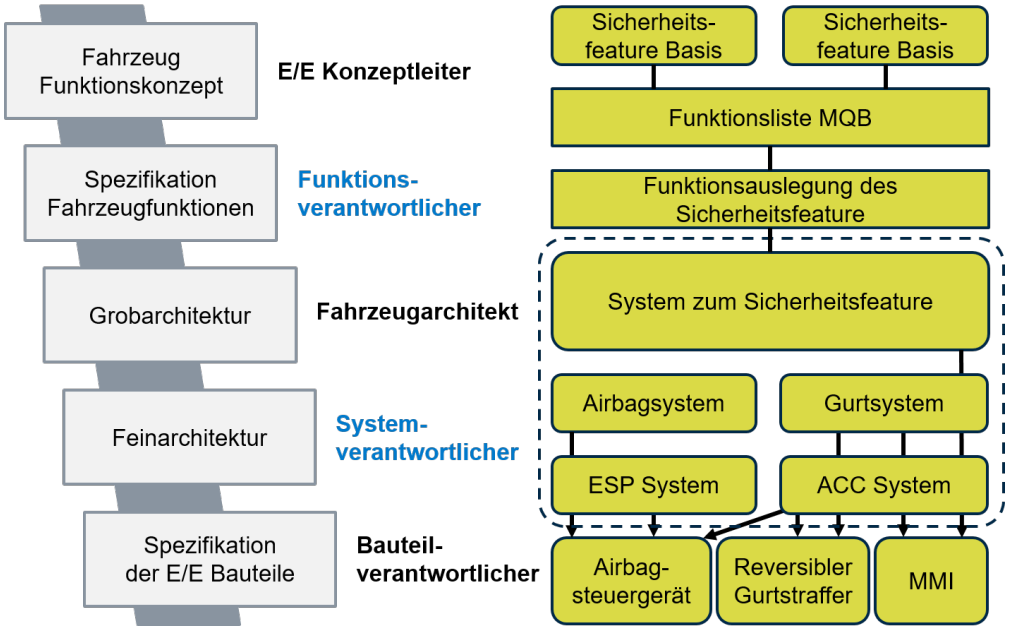


Abbildung 2.5: Verantwortlichkeiten und Beispiel einer Funktion in Anlehnung an [GSSZ13], Original ebd.

Auf der ersten Ebene wird die neue Funktion durch den E/E Konzeptleiter in das Fahrzeugfunktionskonzept aufgenommen. Auf der Ebene der Spezifikation wird die Funktion dann durch den Funktionsverantwortlichen in ihrem gewünschten Verhalten vor Kunde spezifiziert und damit technisch unabhängig beschrieben. Damit ist sie gleichzeitig auch vom Fahrzeug unabhängig formuliert und sie erlaubt eine Wiederverwendung der Beschreibung für unterschiedliche Fahrzeugkonzepte. Die technisch konkrete Umsetzung der Funktion erfolgt dann durch Überführung der Anforderungen auf die Systemebene, deren Ergebnis eine technische Lösung zur Realisierung der Funktion in einem Fahrzeug darstellt. Auf der Ebene der Grobarchitektur gliedert der Fahrzeugarchitekt die Funktion in das Gesamtfahrzeug ein und beschreibt dabei die Aufteilung der Funktion auf evtl. andere Systeme. Bei der Feinarchitektur übernimmt der Systemverantwortliche und realisiert die Funktion durch eine detaillierte Beschreibung als ein eigenständiges (Sub-)System. Dabei können mehrere Systeme eine Funktion realisieren bzw. ein System kann mehrere Funktionen abbilden. Im letzten Schritt werden die nötigen Bauteile vom Bauteilverantwortlichen entwickelt, welcher die Systemanforderungen auf die jeweiligen Systemkomponenten aufschlüsselt. Ein Bauteil kann dabei in mehreren Systemen enthalten sein und damit seinen Beitrag für unterschiedliche Funktionen leisten. [KAB13]

2.3.2 Vernetzung in Fahrzeugen - Bussysteme und ihre Kommunikation

Die Datenübertragung erfolgt im Fahrzeug grundsätzlich bitweise seriell. Dabei kann außerdem zwischen einer unidirektionalen und einer bidirektionalen Datenübertragung unterschieden werden. Die unidirektionale Übertragung erfolgt dabei paarweise; hier spricht man auch von einer Voll-Duplex Leitung. Im Gegensatz dazu handelt es sich bei der bidirektionalen Übertragung um eine gemeinsam genutzte oder auch eine Halb-Duplex-Leitung. Bei einer Halb-Duplex Anbindung kann das angebundene Steuergerät nur abwechselnd Senden oder Empfangen, was meistens im Fahrzeug dem Standard entspricht; mit einer Voll-Duplex-Anbindung ist jedoch ein gleichzeitiges Senden und Empfangen möglich. Weiterhin muss zwischen einer 1- bzw. 2-Draht-Leitung unterschieden werden, wobei bei der 1-Draht Realisierung die Signalrückführung über die Fahrzeugkarosserie erfolgt, wie bspw. bei LIN, so dass diese Art der Kommunikation das Datennetz auch für elektromagnetische Störungen von außen anfällig macht. Bei der Voll-Duplex Realisierung ist die Rückführung besser gegen diese Einflüsse geschützt. Daher kann mit einer 2-Draht-Leitung eine höhere Übertragungsrate erzielt werden als im Vergleich zu einer 1-Draht Variante. [ZS06]

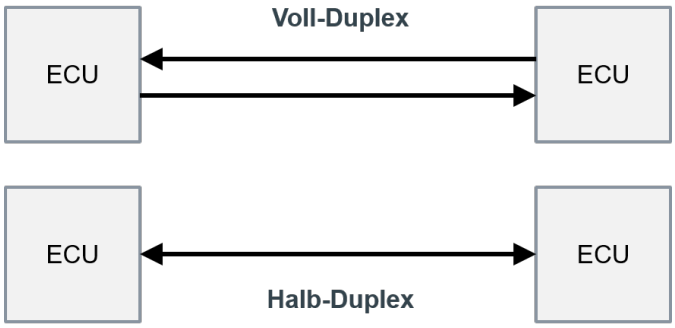


Abbildung 2.6: Punkt-zu-Punkt Verbindung von Steuergeräten als Halb- und Voll-Duplex. [ZS06]

Topologien und Kopplung

Im Vergleich zu einer Peer-2-Peer Verbindung sind insbesondere die Datennetze im Fahrzeug als (Linien-)Bussystem realisiert, so dass mehrere Steuergeräte über die gleiche Leitung miteinander kommunizieren. Hierbei sind die Steuergeräte über Stichleitungen an derselben Verbindungsleitung gekoppelt und über ein Buszugriffsverfahren muss wiederum geregelt werden, wie die Steuergeräte untereinander die zentrale Datenleitung nutzen. Das Bussystem kann dabei

aber auch in weiteren Ausbauformen, sogenannte Topologien, gestaltet sein, wie z.B. als Stern-, Ring-, Baum-, oder auch Maschennetze.

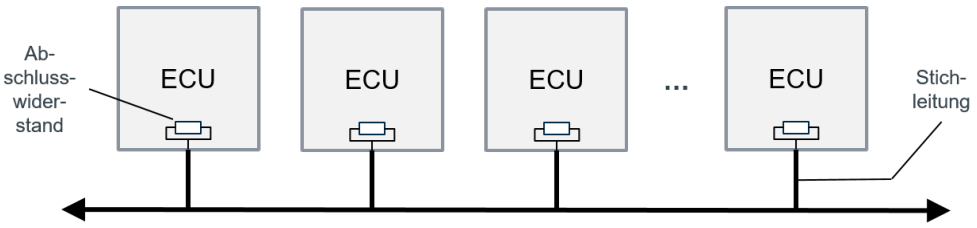


Abbildung 2.7: Datennetz realisiert als Linien-Bus-System. [ZS06]

Der Zugriff kann nach drei Prinzipien erfolgen:

1. als Broadcast, wo alle Steuergeräte die gesendeten Daten empfangen können,
2. als Unicast, wo nur ein bestimmtes Steuergerät die Daten empfängt und
3. als Multicast, wo eine ausgewählte Anzahl an Steuergeräten die gesendeten Daten erhalten.

Die Adressinformation für das zugehörige Steuergerät wird dabei am Anfang der Nachricht kodiert. Die Baumstruktur ist die wohl am häufigsten anzutreffende Topologie im Fahrzeug, da hier mehrere Datennetze miteinander gekoppelt werden, so dass aus mehreren Linien-Bussystemen eine Verzweigung mit Hilfe von Koppelstellen entsteht, welche gewisse Umsetzungsaufgaben erfüllen muss. Es lässt sich unterschieden:

1. der Transceiver, der die Umsetzung der Bussignale auf die ECU-internen Spannungspegel für Sende- und Empfangssignale übernimmt.
2. der Repeater, der einem Signalverstärker ohne Eigenintelligenz entspricht,
3. und das Gateway, welches mehrere Netze auch mit unterschiedlichen Protokollen oder Bitraten koppeln kann, indem es die Botschaften entsprechend übersetzt. [ZS06]

Botschaften, Protokollstapel und Dienste

Die Nutzdaten (Payload) werden zusätzlich noch mit weiteren Informationen, einem sogenannten Vorspann (Header) bzw. auch einem Nachspann (Trailer) versehen. Der Header enthält üblicherweise die Adressinformationen, weitere Informationen zu Menge der übertragenen Daten

und auch deren Art. Beim Trailer werden häufig Informationen zu Fehlerprüfungen und deren Korrektur kodiert. Da die Nutzdaten nicht notwendigerweise immer zusammenhängend sein müssen, werden zwischen unterschiedlichen Datenwerten Stuffing-Bits eingesetzt, um eine saubere Trennung der Daten zu gewährleisten. Handelt es sich bei den Nutzdaten um Messwerte aus der Sensorik, so spricht man auch von Signalen, welche durch ihre Länge und ihre Lage (Offset) inkl. einer möglichen Umrechnungsvorschrift definiert sind. [ZS06]

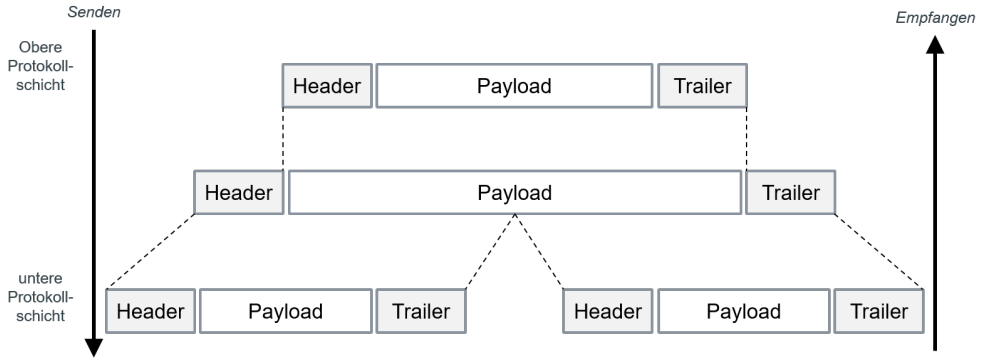


Abbildung 2.8: Aufbau einer Botschaft und das OSI Schichtenmodell. [ZS06]

Das OSI-Schichtenmodell beschreibt die Art und Weise, wie Daten von der Anwendung bis zur Übertragung mehrere Schichten mit eigenen Headern und Trailern passieren. Dabei wird die gesamte Botschaft beim Senden auf der nächsthöheren Ebene vollständig von der nächst niederen Ebene als Nutzlast betrachtet und wiederum mit eigenen Headern und Trailern versehen und umgekehrt beim Empfangen, so dass auf der jeweiligen Schicht nur Header- und Trailer-Informationen verarbeitet werden. So entsteht z.B. beim CAN ein Protokollstapel, bei dem die unterste Schicht meist auf eine Botschaftslänge auf eine max. Länge von 8 Byte bzw. 64bit begrenzt ist. Um nun längere Botschaften senden zu können, werden Botschaften auf einer höheren Ebene auf mehrere kleinere Botschaften aufgeteilt, welches man auch als Segmentierung bezeichnet. Für das Empfangen gilt analog die Desegmentierung. Für ein Kfz ist es jedoch unüblich, auf höheren Ebenen eine kleinere Botschaft zu einer größeren auf einer unteren Ebene zusammenzufassen. [ZS06]

Verfahren zur Datenübertragung

Zwei Verfahren werden zur Datenübertragung genutzt:

1. das zeichen-basierte Übertragungsverfahren und

2. das bitstrom-basierte Übertragungsverfahren

Erstere teilt die gesamte Botschaft inkl. den Header und den Trailer in 8bit Gruppen zu einem Zeichen auf und wird für niedrige Bitraten verwendet. Beim Bitstrom werden alle zugehörigen Teile einer Botschaft als ein gemeinsamer Block im Bittakt übertragen. Die Pausen zwischen den Paketen werden dann mit jeweiligen Minimal- oder Maximalwerten deklariert. [ZS06]

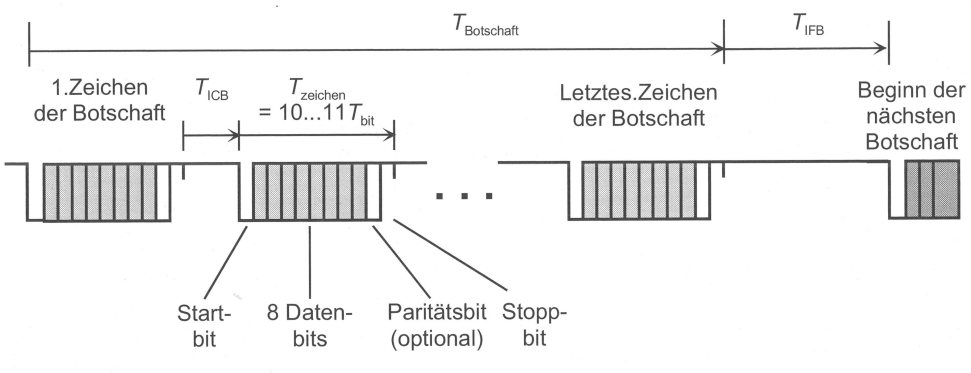


Abbildung 2.9: Schema zum zeichenbasierten Verfahren [ZS06]

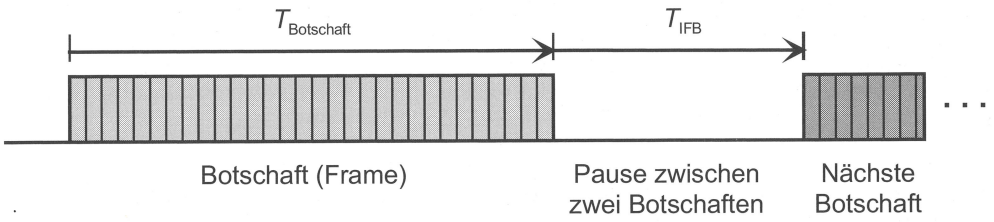


Abbildung 2.10: Schema zu bitweisen Verfahren [ZS06]

Controller Area Network (CAN)

Das Controller Area Network, kurz CAN, ist ein von BOSCH zusammen mit Intel in den 1980er Jahren entwickeltes Netzwerk der Klasse C für Fahrzeuge. Das zugrundeliegende Protokoll wurde durch die ISO 11898 und dem SAE J2284 für den Pkw und mit der SAE J1939 für Nutzfahrzeuge standardisiert. Die Spezifikation ist wie folgt aufgebaut:

- ISO 11898-1: Data Link Layer, der CAN 2.0A und CAN 2.0B entspricht.
- ISO 11898-2: Physical Layer für High Speed CAN.

- ISO 11898-3: Physical Layer für Low Speed CAN.
- ISO 11898-4: Erweiterung für zeitgesteuerte Kommunikation.

Da es sich bei beim High Speed CAN um eine verdrillte Zwei-Draht-Leitung zu einem echten Linien-Bus mit Stichleitungen von max. 30 cm handelt, müssen beide Enden mit üblichen 120 Ohm Wellenwiderständen ausgestattet sein. Bei Kurzschluss oder Aderbruch fällt der Bus aus. Anwendung findet diese Art häufig im Bereich des Antriebsstrangs, da hier Übertragungsraten von 500kbit/s nötig sind. Sind jedoch weniger als 125kbit/s nur notwendig wie z.B. bei Karosserieelektronik, greift man zum Low Speed CAN, der keine Abschlusswiderstände benötigt und daher auch über keine begrenzten Stichleitungen verfügen muss. Somit bleibt auch bei Kurzschluss oder Kabelbruch der Bus funktionstüchtig. [ZS06]

Die Übertragung erfolgt hier bitstrom-orientiert über die CAN Controller und die Botschaften sind folgendermaßen aufgebaut:

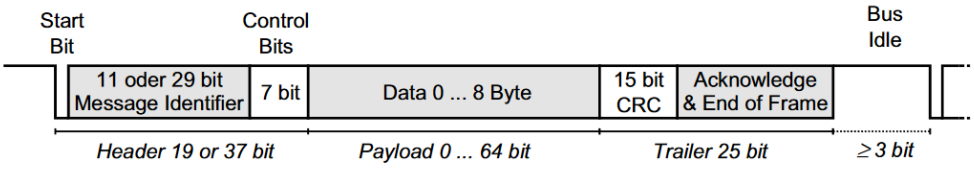


Abbildung 2.11: Botschaftsformat für CAN aus [ZS06]

Weil es sich um ein Broadcast-System handelt, bei welchem die Botschaften anstatt durch Quell- oder Zielinformationen durch Kennungen identifiziert werden, verfügt die Botschaft zu Anfang über einen Message Identifier, der je nach Generation über 11bit (CAN 2.0A) oder 29bit verfügt (CAN 2.0B). Weiterhin wird durch eine Zahl die Priorität der Botschaft definiert, die um so höher priorisiert wurde, je niedriger die Zahl ist. Beim Zugriff auf den Bus wird das sogenannte CSMA/CA Verfahren verwendet, so dass jedes Steuergerät dann senden kann, wenn der Bus für zumindest 3 Bit-Zeiten frei ist. Bei einer Kollision wird die Botschaft mit niedriger Priorität ignoriert und dann wieder vom Steuergerät gesendet, wenn der Bus wieder frei ist. Die Botschaft kann zudem zwischen 0 und 8 Bytes an Nutzdaten verschicken, deren Anzahl im Data Length Code (DLC) Feld innerhalb der Control Bits definiert ist. Am Ende der Nutzdaten wird eine 15bit lange Prüfsummer als Cyclic Redundancy Check mitgesendet. Die Synchronisation der Empfänger erfolgt über einen Bittakt-Generator, der mit dem Startbit des Senders abgeglichen wird und ggf. noch mit zusätzlichen Stuffing-Bits nachsynchronisiert wird. Dies erfolgt meist drei- bis viermal je Botschaft, kann aber im schlechtesten Fall bei jedem fünftem Bit stattfinden. Am Ende senden die Empfänger ein Acknowledge oder eine Fehlermeldung. Bei letzterem

ignorieren alle Empfänger diese Botschaft, um das Netzwerk konsistent zu halten und der Sender verschickt seine Botschaft erneut. Für das Senden sind etwa 225 bis 260 Mikrosekunden notwendig, abhängig von der CAN Generation. [ZS06]

2.4 Die Rolle von Hard- und Software im Fahrzeug

Die zugrundeliegende Hardware besteht überwiegend aus Steuergeräten, die als eigene Rechen- und Steuereinheiten in der Elektrik-Elektronik-Systemarchitektur konzipiert sind und alle wesentlichen Aufgaben innerhalb des Bordnetzes verarbeiten. Zu diesen gehören neben der Sensorauswertung auch die Algorithmen-Berechnung, die Diagnose bzw. der Selbsttest und die Ansteuerung von Aktuatoren. Durch die richtige Wahl dieser Grundaufgaben lassen sich komplexe Funktionen im Fahrzeug realisieren.

2.4.1 Die Ebene der Hardware im Fahrzeug

Im Gegensatz zu Personal-Computern sind bei einem Steuergerät die einzelnen Hardware-Komponenten wie Micro-Prozessor, Speicher sowie Aus- und Eingabebausteine auf einem Silizium-Chip verbaut, so dass häufig von Embedded Micro-Controllern in diesem Zusammenhang gesprochen wird. Da im Zuge der Automobilentwicklung die Vernetzung im Fahrzeug vom einzelnen, unabhängigen Steuergerät hin zu komplexen Steuergerät-Verbünden zunahm, ist es möglich geworden, Teilfunktionen miteinander zu kombinieren und so dem Kunden noch mehr Funktionen zur Verfügung zu stellen. Wallentowitz2006

Es können drei Grundtypen an Steuergeräten unterschieden werden:

1. Eingebettete Steuergeräte
2. Semi-eingebettete Steuergeräte
3. Rechnerknoten

Bei reinen eingebetteten Steuergerät werden alle Grundaufgaben innerhalb einer ECU konzentriert, woraus sich jedoch schnell Verfügbarkeits- und Sicherheitsrisiken aufgrund der Parallelität von Rechenzeit-intensiven Aufgaben und des Schaltens und Treibens mit hoher elektrischer Leistung ergeben. Die semi-eingebetteten Steuergeräte enthalten kaum Treiberbausteine, sondern nur noch diese, welche in direkter Verbindung zur realisierenden Funktion stehen, wie z.B. beim ACC-Steuergerät, bei dem die Sensorik über einen eigenen Bus an das SG gekoppelt ist. Die restliche Vernetzung mit anderen Steuergeräten erfolgt dann über den Antriebs-CAN. Von

Rechnerknoten wird dann gesprochen, wenn viele Funktionen nur mit mehreren Steuergeräten realisiert werden können und die Verfügbarkeits- und Sicherheitsanforderungen mit Hilfe interner oder externer Redundanzen umgesetzt werden können. Viele der neuen Fahrerassistenzsysteme basieren auf dem Prinzip des Rechnerknotens. [WR06, S. 182ff]

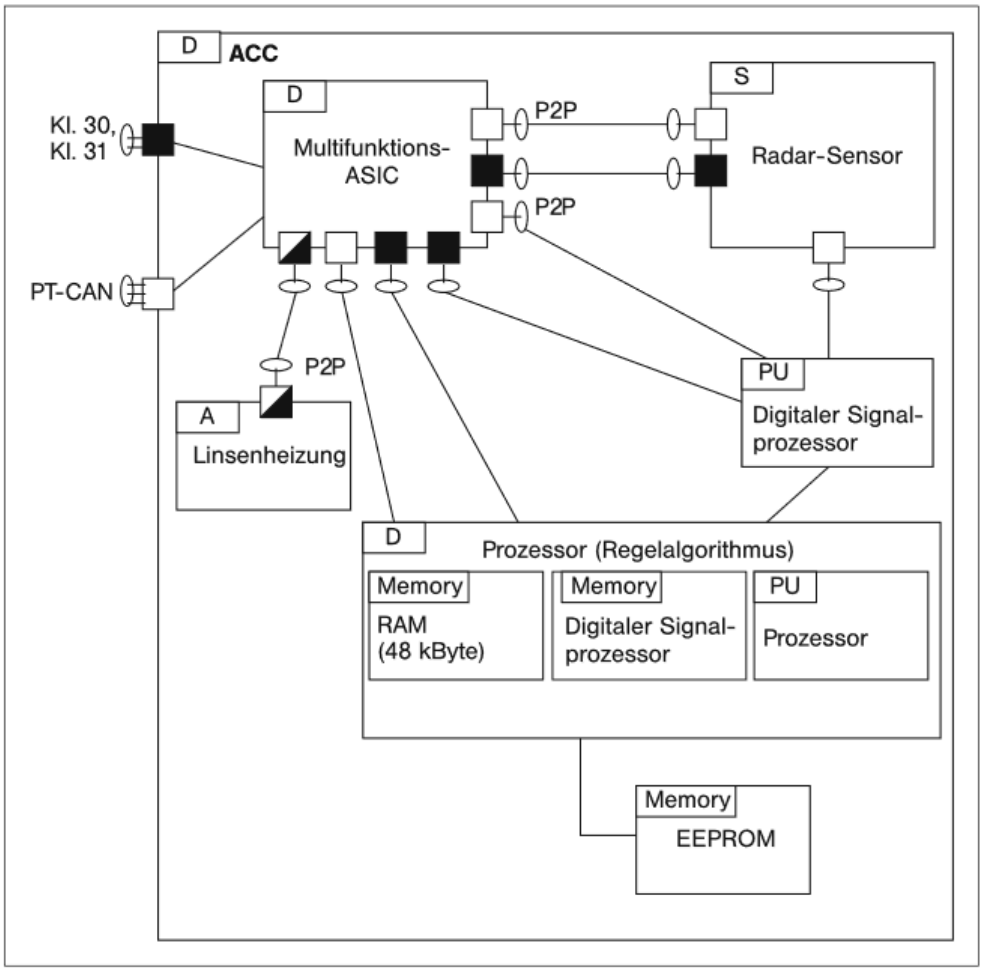


Abbildung 2.12: Beispiel für eine Architektur aus dem ACC Steuergerät. [WR06, S. 182ff]

2.4.2 Die Ebene der Software im Fahrzeug

Zunächst werden verschiedene Softwaremodule benötigt, welche die grundsätzliche Verwendung der zugrundeliegenden Hardware gewährleisten, ohne dass die eigentliche Fahrzeugfunktion umgesetzt wäre. Da es sich hierbei um Systemgrundfunktionen handelt, besteht ein großes Potential, die Grundfunktionen in einer Architektur zusammenfassen und zu standardisieren.

Diese können sich von OEM zu OEM und von Zulieferer zu Zulieferer mehr oder weniger stark unterschieden, so dass wegen der hohen Komplexität der Fahrzeugfunktionen ein weltweiter Standard entwickelt wurde: Die Automotive Open System Architecture (AUTOSAR). [WR06, AUT19, S. 185ff] Dieser erhält zunehmend Einzug in die Fahrzeugentwicklung, wird jedoch noch nicht von allen Branchengrößen konsequent umgesetzt. Durch die Standardisierung dieser Grundfunktionen erhofft man sich neue Impulse auch durch kleinere Zulieferer, die sich beispielsweise nur auf Anwendungssoftware konzentrieren und entsprechende Schnittstellen bedienen, ohne die eigentliche Hardware näher kennen zu müssen bzw. zu entwickeln.

Als Systemgrundfunktionen werden angesehen:

- Anwendung
- Betriebssystem
- Middleware
- Hardware-Abstraktion
- Basisdienste, wie Netzwerkmanagement, Diagnose oder Flash. [WR06]

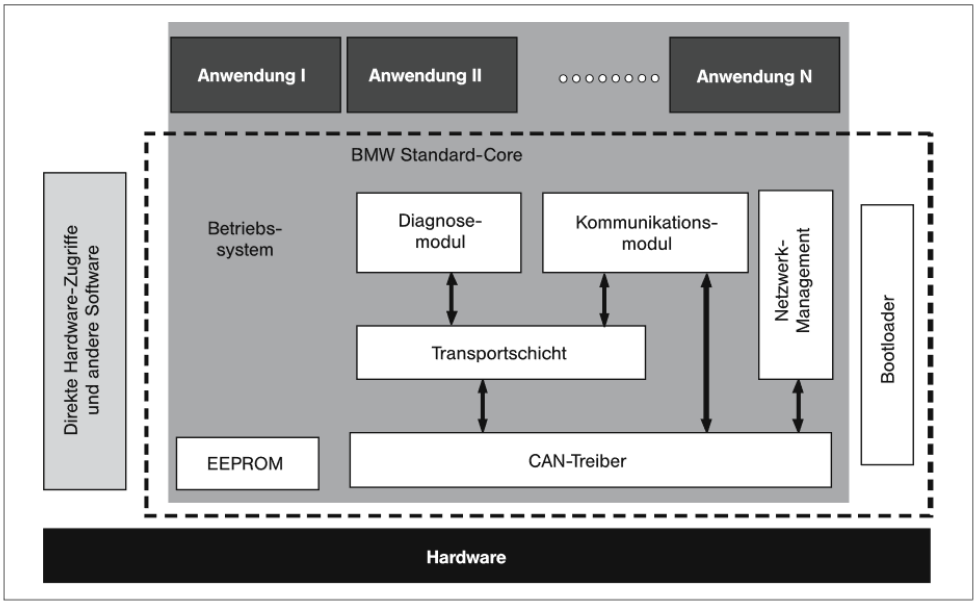


Abbildung 2.13: Beispiel für einen Standard-Core eines Steuergeräts. [WR06, S. 185ff]

Die Entwicklung der eigentlichen Anwendungssoftware-Module kann an sich auch noch als ein eigenständiger (Softwareentwicklungs-)Prozess angesehen werden, der eine Vielzahl von

Entwicklungsmethodiken und -verfahren umfasst und die im Folgenden erläutert werden sollen.

2.5 Automotive Software-Engineering

Die Entwicklung von Software für Fahrzeuge scheint einen vergleichbar kleinen Teil an der Gesamtentwicklung eines Fahrzeugs einzunehmen, sofern man Schäuffele und Zurwakas Standardwerk heranzieht [SZ10]. Dieses Einführungswerk nimmt eine ganzheitliche Sicht im Zuge der Entwicklung von Fahrzeugfunktionen und elektronischer Systeme ein. Hierzu wird mit Hilfe der Systemtheorie das Fahrzeug in mehrere Systemebenen aufgeteilt.

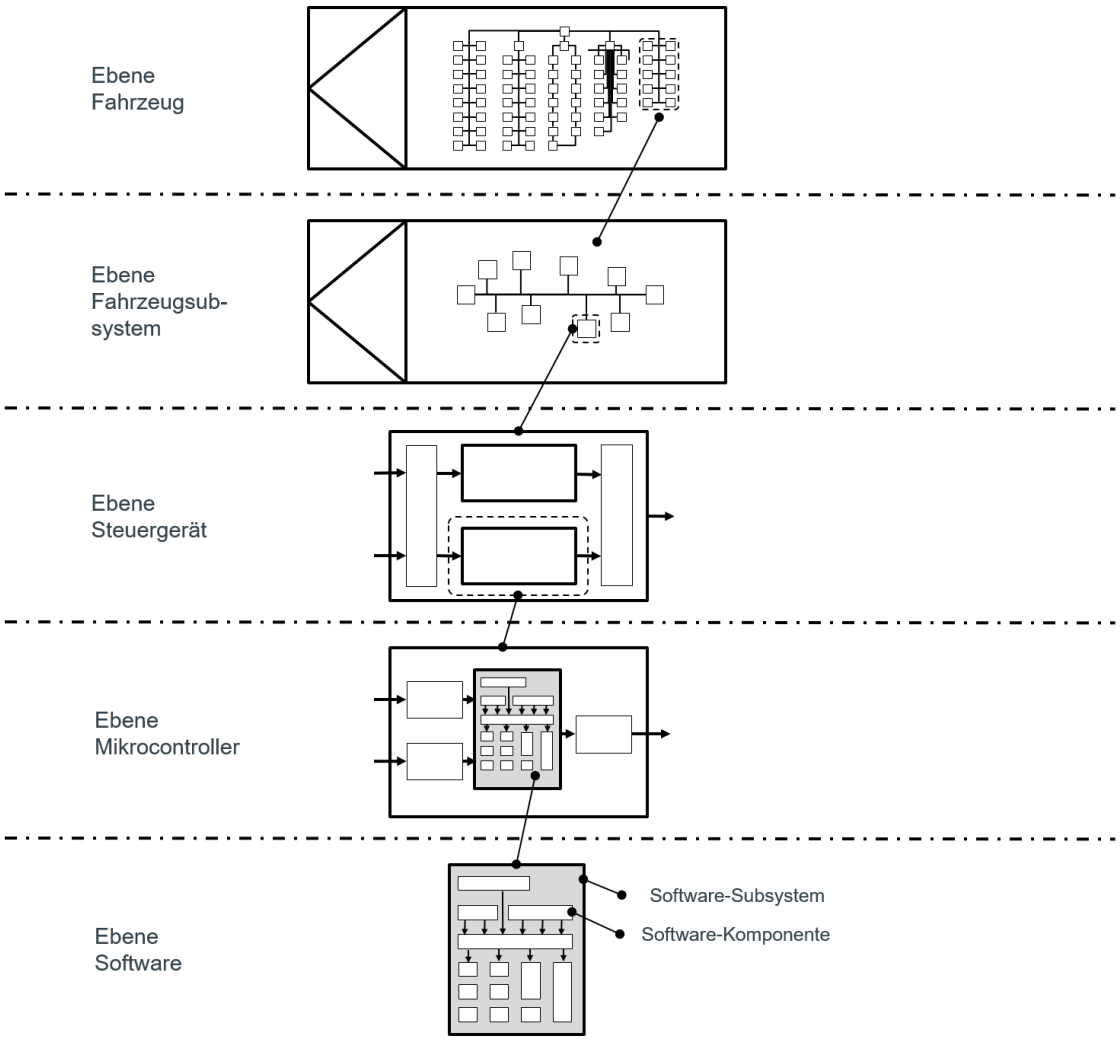


Abbildung 2.14: Die unterschiedlichen Systemebenen in Anlehnung an [SZ10, S. 116]

Die Ebene Fahrzeug betrachtet alle Steuergeräte, welche die geforderten Funktionen umsetzen und im Fahrzeug verbaut werden. Die Ebene Fahrzeugsubsystem teilt diese Sicht noch einmal in ihre Buszugehörigkeit auf, wie z.B. alle Steuergeräte, die zum Antriebsstrang gehören bzw. nötig sind. Daraus wird dann die Ebene Steuergerät herausgelöst, welche ein spezielles Steuergerät betrachtet, die wiederum in die Ebene Mikrocontroller unterteilt werden kann. Erst auf der letzten Ebene wird die konkrete Software bzw. das Software(sub-)system und seine Komponenten beschrieben. Diese Darstellung macht insofern zunächst Sinn, als dass sie hilft, die Komplexität eines vollständigen Fahrzeugs auf den für die Entwicklung wesentlichen Ausschnitt zu reduzieren. Dennoch darf daraus nicht folgen, dass der unteren Ebene Software weniger Bedeutung als einer höheren Ebene beigemessen wird. Zwar wird am Ende des Werks die Wichtigkeit von modellbasierter Entwicklung als Vorteil und zielführend hervorgehoben, jedoch wird der Software-Entwicklung und seiner Methoden selbst nicht der Stellenwert durch Aufführung geeigneter Vorgehensweisen eingeräumt, wie es in anderen Branchen wie z.B. bei der Telekommunikation bereits der Fall ist. Dies ist sicherlich durch die prägende Wissenschaftsdisziplin des Ingenieurswesens zurückzuführen. Doch wie bereits eingangs erwähnt, muss eine interdisziplinäre Sicht unter Einbeziehung der Informatik und speziell des Software-Engineering eingenommen werden, um weitere Fortschritte im Automobilbau zu erzielen. Daher werden im Folgenden entscheidende Methoden und Vorgehensweisen des Software-Engineering für diese Arbeit eingeführt. [SZ10]

Grundlegende Annahmen wie das V-Modell und der fundamentale Testprozess sind bereits beschrieben worden und finden hier ihre Anwendung.

2.5.1 Softwaretechnische Grundkonzepte - Agile Methoden, eXtreme Programming und Objektorientierung

Im Folgenden werden einige Grundkonzepte aus dem Software-Engineering erläutert, die als wesentliche Rahmenbedingungen für diese Arbeit Bestand haben. Dabei wird sich auf die grundlegenden Konzepte zunächst beschränkt, da davon ausgegangen wird, dass diese im Detail bekannt sind. Zur Vertiefung der jeweiligen Aspekte wird auf die einschlägigen Werke verwiesen.

Agile Methoden

Aufgrund der weiter zunehmenden Beschleunigung von Softwareentwicklungsprojekten und einer gewünschten Effizienzsteigerung mit einem höheren Qualitätsanspruch und einem geringeren personellen Ressourceneinsatz wurde bereits zu Anfang des 21. Jahrhunderts darüber dis-

kutiert, wie Software "agiler" abseits der üblichen Vorgehensmodelle wie RUP oder das bereits oben beschriebene V-Modell entwickelt werden kann. Denn häufig sind die Anforderungen an die Softwaresysteme einem solchen Wandel unterzogen, so dass flexibel auf sich ändernde Bedingungen reagiert werden muss, ohne dabei an einen unter Umständen schwerfälligen Entwicklungsprozess gebunden zu sein. Insbesondere wird der Trend zu immer leichteren Methoden verzeichnet, der es erlaubt, gerade in kleinen Softwareprojekten mit reduziertem, bürokratischen Aufwand qualitativ hochwertige Software zu entwickeln [Rum04, S. 10ff]

Als Methode mit Bezug auf das Software Engineering definiert Sommerville wie folgt:

> *[Methoden des Softwareengineering sind definiert als] strukturierte Ansätze für die Softwareentwicklung, darunter Systemmodelle, Notationen, Regeln, Ratschläge zum Entwurf und Anleitung zum Vorgehen. [Som01, S. 21]*

Rumpe definiert den Begriff "Agilität" (einer Methode) durch folgende Charakterisierung:

> *Eine Vorgehensmethode wird als "agil" bezeichnet, wenn sie verstärkt Wert auf folgende Kriterien legt:*
>
> - *Die Effizienz des Gesamtprozesses und der einzelnen Schritte ist möglichst optimal.*
> - *Die Reaktivität, also die Geschwindigkeit, mit der auf sich verändernde Anforderungen oder ein Projektumfeld reagiert wird, ist hoch. Die Planungen sind daher eher kurzfristig und adaptierbar.*
> - *Auch die Vorgehensmethode ist flexibel adaptierbar, um sich inneren und durch das Umfeld bestimmten äußeren und daher nur teilweise kontrollierbaren Projektgegebenheiten dynamisch anzupassen.*
> - *Einfachheit und pragmatische Umsetzung der Entwicklungsmethode und ihrer Techniken führen zu einfachem Entwurf und Implementierung.*
> - *Die Kundenorientierung der Methode erlaubt und erfordert die aktive Einbindung der Kunden während der Projektes.*
> - *Den Fähigkeiten, Kenntnissen und Bedürfnissen der Projektbeteiligten wird im Projekt Rechnung getragen. [Rum04, S. 27]*

Dabei wird Effizienz als das Weglassen unnötiger Arbeit wie z.B. der Dokumentation verstanden. Die Qualität des entwickelten Produktes steht zudem nicht zwingend an erster Stelle bei einer agilen Methode, sondern sie ist orthogonal hierzu. Vielmehr stehen andere Elemente im Vordergrund, wie z.B. ein einfacher Entwurf oder die Testautomatisierung, die aber bei Anwendung einen wesentlichen Beitrag zur Qualitätssicherung beitragen [Rum04, Rum11].

Extreme Programming (XP)

Extreme Programming umfasst eine Softwareentwicklungsmethode, die durch bewussten Verzicht von klassischen Softwareentwicklungselementen schnelleren und effizienteren Code produziert. Die daraus resultierenden Defizite bei der Qualitätssicherung können jedoch durch die Gewichtung angewandter Konzepte wie z.B. erweiterte Testverfahren neutralisiert werden. Insbesondere versucht man, bei XP auf jede organisatorischen Überladung zu verzichten und sich verstärkt auf das Wesentliche zu konzentrieren: den Programmcode. Durch den Verzicht auf aufwändige Dokumentation werden Fehler in deren Erstellung vermieden und es kann durch die Konzentration auf den Code schneller auf neue Anforderungen reagiert werden. Allerdings wird viel Wert auf eine gute Kommentierung des Codes und eine umfassende Testsammlung gelegt. [Rum04, Rum11]

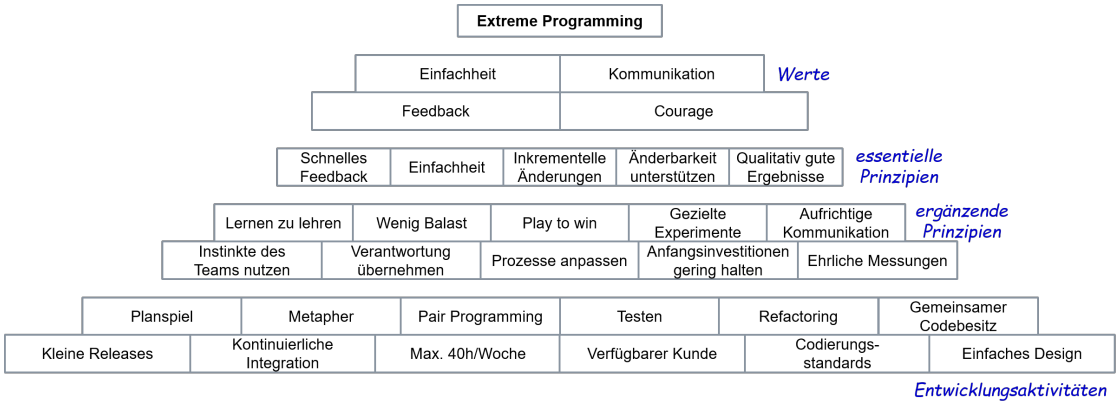


Abbildung 2.15: Aufbau des Extreme Programming in Anlehnung an [Rum04, Rum11]

Speziell werden vier Werte mit XP in Verbindung gebracht, die essentiell für den Erfolg der Methode sind:

- Einfachheit: es gilt einfache Lösungen zu finden, soweit es machbar ist. Denn einfache Lösungen lassen meist schneller und kostengünstiger realisieren, sind einfacher zu warten und weiterzuentwickeln. Dies gilt insbesondere auch für den eigentlichen Entwicklungsprozess.
- Kommunikation: Die Kommunikation zwischen den Projektpartnern sollte intensiv und persönlich erfolgen, so dass ein effektiver Austausch zwischen den Projektpartnern sichergestellt ist. Dies bietet im Übrigen auch Einsparpotential bei der Dokumentation in Abhängigkeit von der Tragweite der Softwarelösungen für den Auftragnehmer, wie z.B. für Banken und Versicherungen, welche die Dokumentation für interne Revisionen benötigen.

- Feedback: Die Qualitätssicherung der Software erfolgt insbesondere durch das direkte Feedback-Prinzip. Beispielsweise nimmt das Testen einen wichtigen Stellenwert bei dieser Methode ein, so dass der Entwickler direkt ein Feedback zu seinem Code bekommt. Aber auch das Feedback vom Auftraggeber bzw. vom Projektpartner ist wichtig, so dass in kurzen Iterationsschritten Software-Releases zur Verfügung gestellt werden.
- Mut: Mut ist wichtig für diese Methode, da unter Berücksichtigung der Einfachheit einer Lösung meist wichtige Aspekte von weniger wichtigen unterschieden werden müssen. Zudem muss auch entschieden werden, dass ein gewisser Aspekt nicht in die Lösung Einzug erhält, da er andernfalls die Lösung wesentlich komplexer werden ließe. Hier gilt das Prinzip "Mut zur Lücke". Aber es gehört auch im Umgang miteinander, da Feedback auch nicht selten als persönliche Kritik wahrgenommen oder auch auf fachlicher oder persönlicher Ebene missverstanden werden kann. Für die Entwickler gehört zudem Mut dazu, da sie sich einerseits auf die Domäne des Auftraggebers und andererseits auch auf seine Fachsprache einlassen müssen. [WRL05]

Für den Erfolg von XP-Projekten konnten bisher folgende Faktoren als entscheidend betrachtet werden:

- Motiviertes Team und der Arbeitsplatz ist für Pair Programming ausgelegt.
- Durch die räumliche Nähe des Kunden kann er selbst aktiv am Projekt partizipieren und wertvollen Input liefern.
- Die Wichtigkeit der Tests nimmt zu, wenn Änderungen ins System einfließen, neue Entwickler hinzukommen oder das System über eine gewisse Komplexität verfügt, so dass automatisierte Tests erforderlich sind.
- Zwang zur Einfachheit führt zu einer Reduktion von Aufwand und Arbeitslast durch zusätzliche Dokumentation.
- Dauerhafte und direkte Einbindung von Kunden in das Projekt fördert den Austausch von Vorstellungen und Anforderungen und führt letztlich bei Umsetzung auch zu einer höheren Akzeptanz beim Kunden durch soziale Übereinkunft.

Folgende Grenzen der Anwendbarkeit von XP konnten bisweilen aufgezeigt werden:

- Begrenzung auf Projekte bis zehn Personen
- geringe Fähigkeit der Skalierung für große Projekte

- Polarisierung der Meinungen hinsichtlich der Professionalität der Methodik, also mangelnde Akzeptanz und Projektteilnehmern.

Hinsichtlich der Skalierbarkeit konnten mittlerweile weitere Konzepte XP zugeordnet werden, um bürokratische Elemente für größere Projekte hinzuzufügen.

Für eine vertiefende Sicht sei auf die Werke von Beck [BA04] und [Rum04, Rum05, Rum11, Rum12, Rum16, Rum17] verwiesen.

2.5.2 Modellbasierte Software-Entwicklung - MDA, UML und Sprachprofile

Die grundsätzliche Idee der modellbasierten bzw. der modellgetriebenen Softwareentwicklung entstammt der eigentlichen Kernidee von Modellen:

> *Sie abstrahieren und fokussieren auf das Wesentliche und sie schlagen eine Brücke von der falschen Problemwelt in die technische Lösungswelt. [GPR06, S. 19]*

Bei einem modellbasierten Ansatz werden die Modelle zu einem Artefakt ersten Ranges erhoben, aus der letztlich auch ein konkretes Software-System generiert werden kann. Dies bedeutet dann für die Aussagekraft der Modelle einen Wandel, so dass diese von einem rein deskriptiven Charakter hin zu einem präskriptiven wechseln. Weiterhin begleitet einen modellbasierten Ansatz, dass es nicht **die** eine Entwicklungssprache geben kann, da nicht alle domänenspezifische Probleme universell darstellbar sind. Daher sollte hierfür eine nächsthöhere Abstraktionssprache gewählt werden, so dass selbstdefinierte, domänenspezifische Sprachen, versehen mit einer semantischen Eindeutigkeit, für die Problemstellung der Domäne entwickelt werden. [GPR06]

Aus dieser Betrachtungsweise der zwei Leitmotive ist die MDA entstanden, wobei der Begriff der Architektur hier nicht mit einer konkreten Architektur eines Software-Systems verwechselt werden sollte, sondern vielmehr als eine übergeordnete Infrastruktur zur Durchführung eines MDA-basierten Softwareentwicklungsprozesses betrachtet werden. [GPR06]

MDA folgt folgendem Grundsatz: Eine Spezifikation einer Software-Komponente sollte unabhängig von ihrer technischen Implementierung beschrieben werden. Daher werden auch mehrere Abstraktionslevel innerhalb des MDA Ansatzes unterschieden, denen die Modelle zugeordnet sind: Zum einen die plattformunabhängigen, welche die konzeptionelle Basis des Ansatzes darstellen, zum anderen die plattform-spezifischen, wobei der Begriff der Plattform hier relativ

aufgefasst werden muss. Denn die spezifischen Modelle auf der einen Ebene können die unabhängige Vorlage auf der anderen Ebene sein und vice versa. [GPR06]

Der Ansatz des MDA verfolgt mehrere Ziele:

- Konservierung der Fachlichkeit
- Portierbarkeit
- Systemintegration und Interoperabilität
- Effiziente SW-Entwicklung
- Domänenorientierung

Bei der Konservierung geht es vor allem um den Erhalt des impliziten Wissens beispielsweise eines Unternehmens oder auch des dortigen Fachpersonals. Durch die Portierbarkeit soll gewährleistet werden, dass die technischen Aspekte systematisch abstrahiert werden, damit eine Migration auf neue Versionen von Technologien oder auch die Portierung auf andere Zielumgebungen erfolgen kann. Bei der Systemintegration und der Interoperabilität soll besonders auf erleichterte Bildung von Schnittstellen geachtet werden, welches durch die Trennung von fachlichen Konzepten und der repräsentierenden Technologie gelingt. Die für die Durchführung nötigen Softwarewerkzeuge beschleunigen insgesamt den Entwicklungsprozess und gestalten ihn dadurch effizienter als bei anderen Softwareentwicklungsansätzen. Und durch die Konzentration auf die domänenspezifischen Modelle und ihre Erstellung wird ermöglicht, dass ein Wissensvorsprung erzielt werden kann, da eine üblicherweise hohe Technikzentrierung entfällt. [GPR06]

Die Unified Modelling Language (UML)

Die UML [OMG11, OMG12] ist eine Modellierungssprache, die das Ziel hat, die Effizienz der Softwareentwicklung und insbesondere eines jeden Entwicklers zu verbessern. Dies will man damit erreichen, dass sowohl die Modellerstellung als auch die Implementierung sowie der Test deutlich beschleunigt werden. Weiterhin erlauben dann diese Modelle des Software-Systems eine automatische Herleitung des zugrundeliegenden Codes, wozu jedoch die Modellierungssprache wesentlich kompakter sein muss als die Implementierung selbst.

Verschiedene Einsatzformen der Modelle sind also denkbar: Zum einen die Codegenerierung, worauf noch näher in einem nachgelagerten Abschnitt eingegangen wird; zum anderen aber auch zur Kommunikation speziell zwischen Entwicklern, da durch Wahl von relativ abstrakten und

informellen Modellen oder -teilen der zu erörternde Aspekt bzw. der Diskussionspunkt besprochen werden kann, ohne penibel die Korrektheit der Notation einhalten zu müssen. Letzteres wird entscheidend für die Codegenerierung sein.

Insgesamt haben graphische Notationen eine Reihe von Vor- und Nachteilen; bspw. erleichtern sie die Kommunikation, so dass schneller ein Überblick über Systemteile und deren Beziehungen untereinander erstellt werden kann. Jedoch benötigen solche Notationen deutlich mehr Platz, so dass deren Informationsdichte je nach Größe des Modells deutlich geringer ist als im Vergleich dazu gewöhnlicher Text.

Somit vereint die UML eine Menge von objektorientierten Modellierungsansätzen mit verschiedenen Notationen zu einer standardisierten Modellierungssprache. Sie hat mittlerweile einen hohen Verbreitungsgrad, der insbesondere darin begründet liegt, dass die Trennung von Vorgehensweise und zugrundeliegender Notation vollzogen wurde.

In ihrer Ausprägung ist die UML aus Sicht von Rumpe jedoch überladen, teilweise aber auch nicht ausreichend, um ein System vollständig beschreiben zu können. Sie erlaubt allerdings diverse Variationsmöglichkeiten, die als Sprachprofile bezeichnet werden. Damit fungiert die UML als Sprachrahmen, die durch die Anpassung im "Vokabular" einen konkreten Dialekt bzw. je nach Projektkontext zur Fachsprache mutiert. Für eine weitere Erörterung hinsichtlich der UML und Sprachprofilen sei auf Grönninger et al., [GKR+08], Schindler [GKRS06] sowie auf [Rum04, Rum05, Rum11, Rum12] verwiesen.

Prinzip der Sprachprofile und die UML/P

Das Sprachprofil UML/P ist aus der Maßgabe entstanden eine Modellierungssprache zu entwickeln, die gleichzeitig auch als vollwertige Programmiersprache eingesetzt werden kann, somit folglich auch als Implementierungs- und Testsprache dient. Dies wird dadurch erreicht, dass die Kernnotation auf sechs Teile reduziert wird und gleichzeitig die Integration von Java-Codestücken bzw. Text-Anpassungen in Java-Syntax vorgenommen werden. Dabei steht das "/p" für programmier-geeignet. Dennoch bleibt dieses Sprachprofil hinsichtlich Domänen- oder Anwendungsgebiet bzw. auch in Bezug auf die verwendete Technologie in jedem Falle erweiterbar. Kombiniert mit einer agilen Vorgehensweise wie oben beschrieben, wird damit die UML durch die Reduzierung auf eine Kernnotation aus sechs Teilen zu einem leichtgewichtigen Ansatz zur Entwicklung von Softwaresystemen entwickelt, die insbesondere auch die Generierung von Produktions- und Testcode unterstützt [Rum05, Rum12, Sch12].

Schindler hat in seiner Arbeit darauf aufbauend ein Werkzeug zur agilen, modularen und mo-

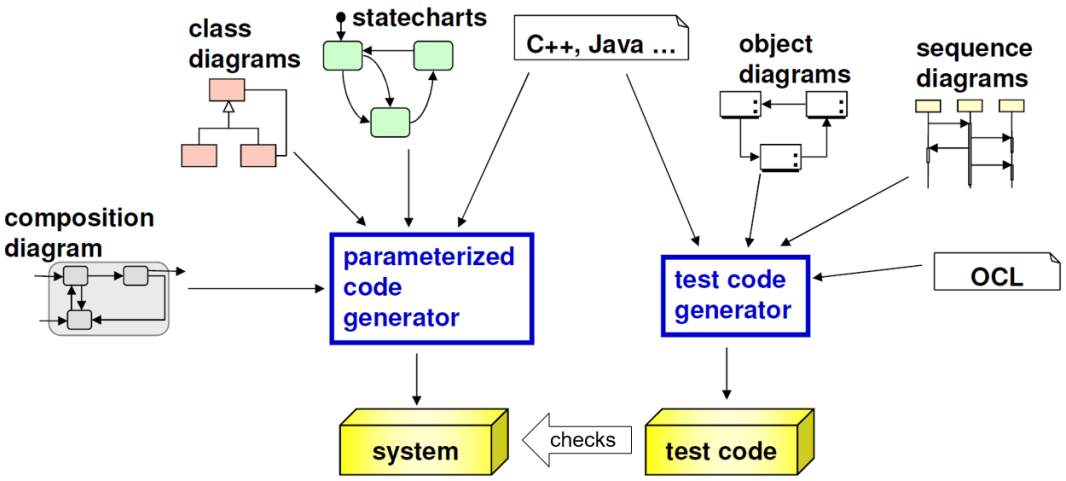


Abbildung 2.16: Übersicht der Diagramme in der UML/P [Sch12], adaptiert nach [Rum04]

dellgetriebenen Softwareentwicklung entworfen und implementiert, dass den zunächst theoretischen Ansatz von Rumpe in eine praktische Anwendung gegossen hat. Wichtige Aspekte dabei waren die Erstellung, die Verwaltung, die Verarbeitung, die Analyse und die Transformationen von Modellen sowie Eigenschaften einzubinden, welche die UML/P von einer General Purpose Language abgrenzen, so dass Unterspezifikationen, die syntaktische Erweiterbarkeit und die semantische Variabilität berücksichtigt werden. Dabei wurde insbesondere eine textuelle Notation gewählt und einer graphischen Variante vorgezogen. Für eine vertiefende Sicht sei auf die Dissertation von Schindler verwiesen. [Sch12] Eine wesentliche Rolle spielt dabei auch die generative Software-Entwicklung, die im folgenden näher erläutert wird.

2.5.3 Generative Software-Entwicklung

Die Vorteile der generativen Software-Entwicklung umfassen sowohl die Verständlichkeit, die Effizienz und die Wiederverwendbarkeit. Letztere kann auf mehreren Ebenen erfolgen, wie z.B. durch ein angepasstes Modell, das auch in anderen Projekten eingesetzt wird. Auch die Nutzung eines geeigneten Frameworks für gleichartige Projekte ist denkbar, bei welchem angepasste Codegeneratoren genutzt werden, um den projektspezifischen Anforderungen gerecht zu werden. Weiterhin kann aber auch technisches Wissen zur Erzeugung von Schnittstellen und sicheren Verbindungen in den Codegeneratoren hinterlegt werden, so dass dieses Wissen für eine Reihe von Projekten verfügbar wird, sofern es gewünscht ist. Die Wiederverwendbarkeit kann aber auch innerhalb des Projektes möglich sein, so dass Modelle sowohl zur Erzeugung von

Produktions- als von Testcode herangezogen werden können [Rum05, RSSW10].

Grundbegriffe

Es wird sich bei der Einführung der Grundbegriffe bzgl. der Codegenerierung an Rumpe orientiert.

Konstruktives Modell: bezeichnet die Spezifikation eines Systems, welches durch einen Codegenerator den Produktivcode generiert. Die eingesetzte Spezifikationssprache ist dabei grundsätzlich ausführbar.

Deskriptives Modell: ist eine Spezifikation, die primär zur Beschreibung des Systems dient, da gewisse Aspekte abstrahiert bzw. bewusst oder auch unbewusst unvollständig gehalten werden und somit nicht konstruktiv eingesetzt werden können.

Testmodell: Hier handelt es sich um eine Spezifikation, die zur Ableitung (manuell oder automatisiert) von Tests verwendet wird. Dabei wird ausführbarer Code generiert, der als Basis für Testdatensätze, -treiber oder -Sollergebnisse dienen soll.

Codegenerierung: Der Begriff beschreibt den eigentlichen Vorgang, der aus einem konstruktiven Modell Code erstellt.

Codegenerator: Hierbei handelt es sich um eine Komponente, die aus einem konstruktiven Modell einer höheren Programmiersprache Transformationen in die gewünschte Zielsprache durchführt.

Skript: Dies dient zur Parametrisierung des Codegenerators und damit zur Steuerung für eine plattform- bzw. anwendungsspezifische Codegenerierung.

Template: bezeichnet eine spezielle Form eines Skripts. Hierbei werden Codemuster, in welche die Modellelemente eingesetzt und transformiert werden.

Arten zur Codegenerierung

Die Notationen der zugrundeliegenden Modellierungssprache können, wie z.B. bei der UML/P, die exemplarischen und vollständigen Strukturen und Verhalten eines Softwaresystems erlauben zu beschreiben, jedoch eignen sich einige Formen eindeutiger für die Generierung von Produktivcode als im Vergleich zu Testcode und vice versa. Die Abbildung in Anlehnung an Rumpe beschreibt die Eignung anhand der Breite der Pfeile für die jeweilige Code-Art. [Rum05, Rum12]

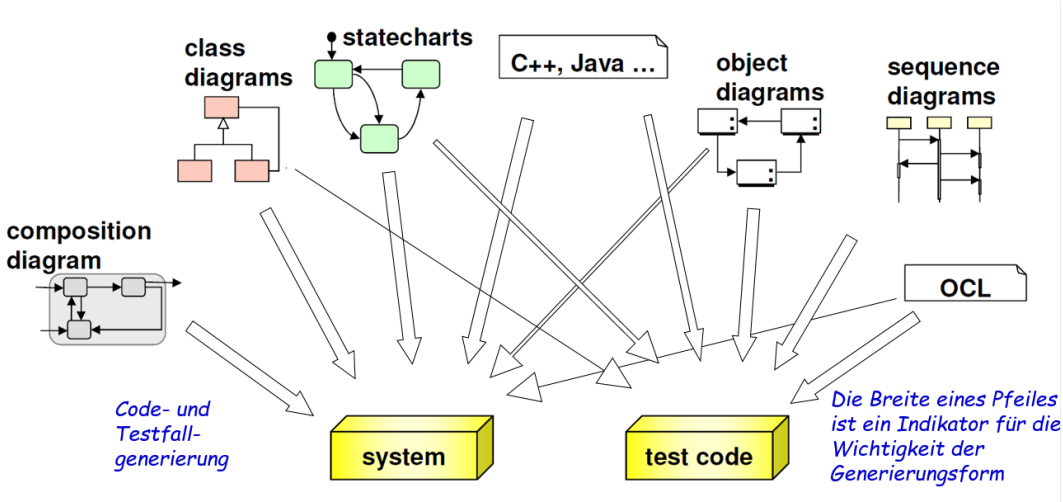


Abbildung 2.17: Diese Diagramme eignen sich für die Codegenerierung nach Rumpe, adaptiert nach [Rum05]

Techniken zur Codegenerierung

Nach Rumpe gibt es bei der Codegenerierung mehrere Dimensionen von Variationen zu beachten, welches insbesondere die Plattformabhängigkeit umfasst. So können je nach Zielplattform mehrere Mechanismen wie z.B. für die Kommunikation von verteilten Systemen, für Einbruchssicherheit sowie für Authentizität und Integrität einbezogen werden. Dabei ist speziell die Hardwareabhängigkeit bei der Wahl der Mechanismen zu berücksichtigen bzw. an die jeweiligen Bibliotheken anzupassen. Aufgrund des Wandels in der Plattform, der Hardware oder auch der Software muss die Generierung so flexibel wie möglich gehalten werden. Dies wird durch grundlegende Ansätze gewährleistet:

1. Entweder durch die Generierung für eine abstrakte Schnittstelle
2. oder durch eine Parametrisierung der Codegenerierung.

Die Abbildungen 2.18 und 2.19 verdeutlichen die Unterschiede in der Generierung. Erstere ist dadurch gekennzeichnet, dass Code gegen eine Schnittstelle generiert wird, somit also noch per Hand implementiert werden muss, um den hardware- oder plattformspezifischen Details gerecht zu werden. Der letztere hingegen umfasst konkrete Generator-Anweisungen, welche die plattformabhängigen Informationen und eventuelle Zusatzfunktionalität enthalten können, wodurch mehr Flexibilität erreicht wird. Wann welche Form geeignet ist und aus welchen Gründen sei den weiteren Ausführungen Rumpes in [Rum05, Rum12] entnommen. Festzuhalten ist jedoch,

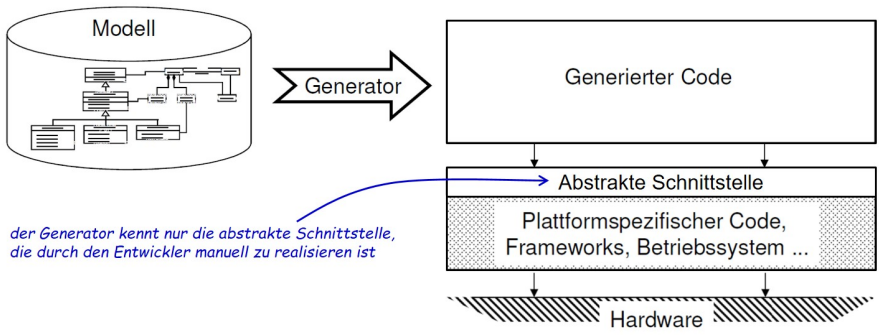


Abbildung 2.18: Beispiel für Codegenerierung gegen eine abstrakte Schnittstelle nach [Rum05].

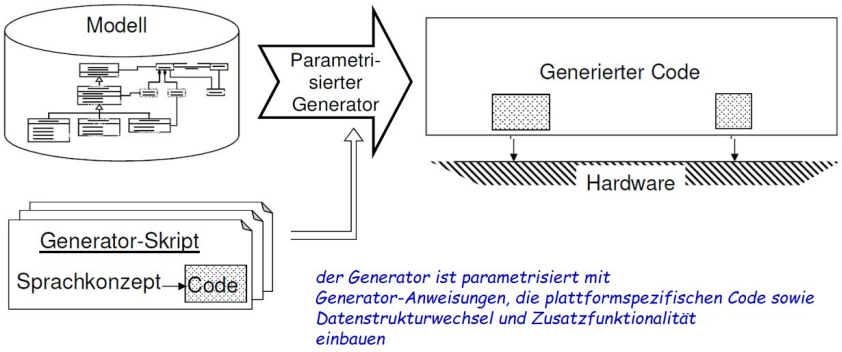


Abbildung 2.19: Beispiel für eine parametrisierte Codegenerierung gegen eine Schnittstelle nach [Rum05].

dass eine Mischform (2.20) der beiden Varianten zu den vielversprechendsten Formen gehören, welche jedoch von den jeweiligen Projektrahmenbedingungen abhängen.

Domänenspezifische Sprachen und Grammatiken

Die Parametrisierung wird häufig durch Skripte übernommen. Durch Sprachkonzepte lassen sich diese Regeln systematisieren und formalisieren, so dass sie für Maschinen zu verarbeiten sind. Im Folgenden wird nun eine Einführung in die Prinzipien und zugrundeliegenden Konzepte von domänenspezifischen Sprachen gegeben. Dabei wird sich auf die Arbeiten von Krahn [Kra10], Schindler [Sch12] und Völkel [Voe11] bezogen und ihre Erkenntnisse in diesem Abschnitt gebündelt.

Als “Domäne” wird grundsätzlich in der Informatik das Anwendungsgebiet verstanden, für das ein (Software-)System entwickelt wird bzw. auf welches sich das System konkret später bezieht,

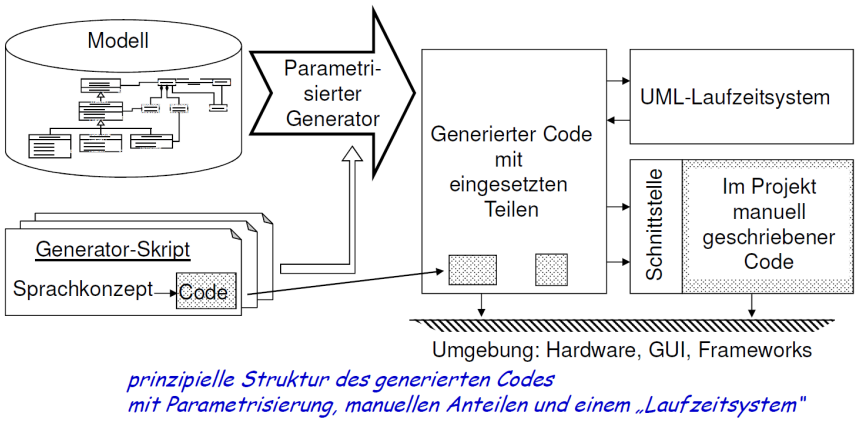


Abbildung 2.20: Mischform der Codegenerierung nach [Rum05].

z.B. in dem hier beschriebenen Kontext handelt es sich um das Anwendungsgebiet der Fahrzeugtechnik mit dem Fokus auf Fahrsituationen im sicherheitskritischen Bereich für die betroffenen Verkehrsteilnehmer. Der Grundgedanke zur DSL besteht darin, dass die vorhandenen Notationen der Domänen so genutzt werden, dass das zu entwickelnde System damit beschrieben werden kann. Ziel ist es, den Zugang für die Domänen-Experten einfacher zu gestalten und den natürlichen Sprachgebrauch in die Entwicklung einzubeziehen. Neben DSLs können auch sogenannte General Purpose Languages (GPLs) unterschieden werden, zu denen beispielsweise Java, C++ und andere höhere Sprachen zählen. Der Vorteil einer Verwendung von DSLs kann darin gesehen werden, dass die Domänen-Experten ein besseres Verständnis über das System selbst erhalten, eigene Optimierungen selbständig entwickeln und einfließen lassen können. [DKV00]. Das so formalisierte Domänen-Wissen erlaubt dann unabhängig von der eigentlichen Implementierung verwendet zu werden, wie z.B. für die Migration von technischen Plattformen, für die Optimierung des Systems selbst oder auch für das Wissensmanagement. Als Risikofaktoren können die hohen Kosten für den Entwurf, die Implementierung und die Wartung der DSLs gesehen werden. Außerdem setzt die Anwendung der Konzepte zu DSLs entsprechende Schulungen der Mitarbeiter voraus. Letztlich ist entscheidend, dass der richtige Aufgaben- und Anwendungsbereich der DSLs identifiziert wird, damit ein Mehrwert gegenüber einem traditionellen Entwicklungsansatz erzielt wird.

Ein gängiges Beispiel ist die Entwicklungsumgebung von MathLab/SimuLink, die mit Hilfe von Codegeneratoren vorhandenen Quellcode in eine GPL generieren, wodurch die Entwicklung von Codegeneratoren prinzipiell verkürzt werden kann und stattdessen die zugehörigen Compiler weiterentwickelt bzw. zertifiziert werden, sofern dies erforderlich ist. Im gerade in der Automobilindustrie ist dies ein übliches Verfahren für die Softwareentwicklung im Fahrzeug

[GKR+08, Kra10].

Krahn führt im Rahmen seiner Arbeit eine Reihe von Definitionen zur Beschreibung des Begriffs "Domain-Specific Languages" an; bei einem Vergleich konnte er diese wesentlichen Eigenschaften identifizieren:

- die Sprache orientiert sich an der Anwendungsdomäne und weniger an der technischen Realisierung
- falls sie ausführbar ist, ist die Ausdrucksmächtigkeit nur eingeschränkt im Sinne der Domäne, weniger als universelle Berechenbarkeit
- sie bietet ein kompaktes Lösungspotential für die jeweilige Domäne.

Dabei ist aus Krahns Sicht eine harte Trennung von GPLs und DSLs an sich nicht möglich.

Zur eigentlichen Definition einer DSL gehören vier wesentliche Teile:

1. die konkrete Syntax, in welcher die Sprache graphisch oder textuell notiert wird.
2. die abstrakte Syntax, in welcher die Sprache werkzeug-intern repräsentiert ist.
3. die Kontextbedingungen, welche die regulären Ausdrücke einer Sprache definieren.
4. und die Semantik, welche die Bedeutung einer Sprache beschreibt.

In der konkreten Syntax werden die Modelle der eigentlichen Sprache verfasst. Im Rahmen der Verarbeitung dient die abstrakte Syntax als Grundlage für die gewünschte Funktionalität der Werkzeuginfrastruktur wie z.B. die Codegenerierung oder die Analyse. Aus den Kontextbedingungen kann auch ein Teil der Analyse erfolgen, da durch sie anhand der Ausdrucksregeln fehlerhafte Konstrukte und mögliche Inkonsistenzen festgestellt werden können. Die Semantik, welche als Abbildungen der enthaltenen Sprachelemente auf bekannte Elemente verstanden werden kann, sichert die Modelle hinsichtlich unpräziser oder nicht eindeutiger Beschreibungen ab und darf sich nicht über Werkzeuggrenzen in ihrer Bedeutung verändern. [GKR+08, Kra10, Sch12]

Zudem gibt es verschiedene Techniken, die Definition einer DSL zu erstellen. Es können

- grammatikbasierte Ansätze
- und modellbasierte Ansätze

Bei den grammatikbasierten Ansätzen sind verschiedene Arten von Grammatiken für die konkrete und die abstrakte Syntax verwendbar. Aber auch Baumstrukturen sind für ihre Beschreibung

denkbar, solange es sich um kontextfreie Grammatiken handelt. Bei modellbasierten Ansätzen werden ähnliche Datenmodellierungstechniken zur Festlegung von abstrakter Syntax verwendet, wie z.B. bei Klassendiagrammen. Sie dienen vornehmlich zur Defintion von graphischen Modellierungssprachen wie der UML.

Prinzipiell können zwei Arten von Sprachen zur Erstellung von Modellen unterschieden werden:

1. Graphische Sprachen
2. und textuelle Sprachen.

Graphische Sprachen benötigen als Voraussetzung einen Editor zur Unterstützung der Notation, wobei er zeitgleich auch als Übersetzer von der konkreten auf die abstrakte Syntax fungiert. Verschiedene Werkzeuge existieren hier am Markt, wie z.B. Eclipse Modeling Project (vgl. weitere bei [Sch12]). Für die Verarbeitung von textuellen Sprachen ist eine Lexer-Parser Infrastruktur Voraussetzung. Dabei identifiziert der Lexer die einzelnen Wörter einer Sprache, während der Parser die syntaktische Analyse der Wörter übernimmt und in die abstrakte Syntax überführt. Üblicherweise werden kontextfreie Grammatiken zur Definition von Wörtern und den Sprachaufbau verwendet. Formal kann dies z.B. in der Extended-Backus-Naur-Form (EBNF) erfolgen. [GKR+08, Kra10, Sch12]

Der MontiCore-Ansatz

Beim MontiCore-Ansatz nach Krahn umfasst das Paradigma der generativen DSL-Entwicklung. Dabei werden für der Erstellung und die Implementierung von DSLs Generatoren gebaut, deren Input Sätze aus der Grammatik der Sprache sind. Sie werden in diesem Zusammenhang auch als Modelle bezeichnet.

Die Nutzungsformen können sich auf viele Aspekte der Software-Entwicklung beziehen, jedoch muss im Rahmen des Projektes identifiziert werden, in welchem Umfang DSLs und zugehörige Generatoren eingesetzt werden. Insgesamt besteht das Ziel darin, nicht ein vollständiges Softwaresystem zwingend zu erstellen, sondern komplexe Sachverhalte, die eine aufwendige Implementierung erfordern, kompakter darzustellen und somit einen Effizienzvorteil gegenüber einer manuellen Implementierung zu erzielen.

Eine ausführlichere Diskussion von Vor- und Nachteilen des Ansatzes ist in den hier angeführten Arbeiten von Krahn [Kra10], Schindler [Sch12], Völkel [Voe11] und Holldöbler/Rumpe [HR17] zu finden.

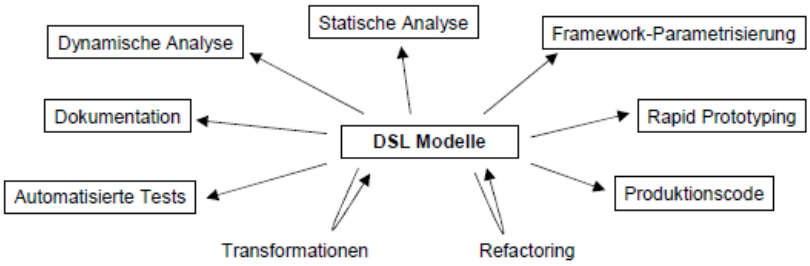


Abbildung 2.21: Nutzungsformen nach Krahn [Kra10]

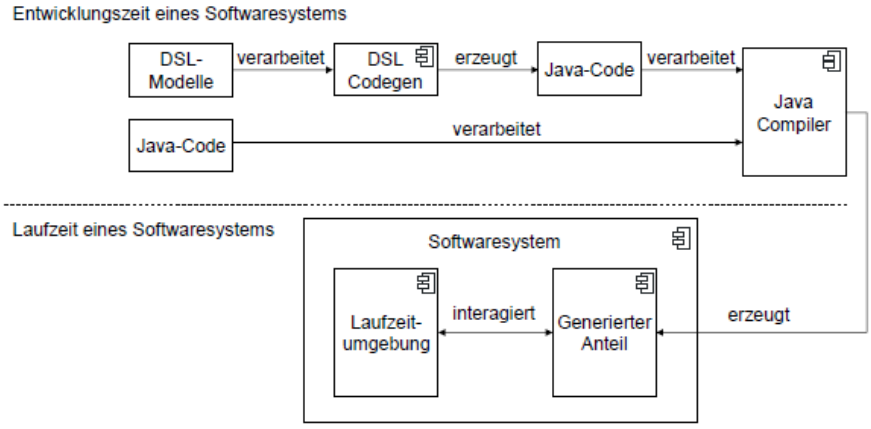


Abbildung 2.22: Prinzip der Codegenerierung für DSLs nach Krahn [Kra10]

Die DSL-Modelle werden durch die Codegeneratoren verarbeitet, die wiederum den gewünschten Quellcode erzeugen. Der entsprechende Compiler baut dann das Softwaresystem oder Teile davon mit dem generierten Anteil und interagiert mit der Laufzeitumgebung.

In den oben genannten Quellen ist auch eine detaillierte Beschreibung vom MontiCore Framework nachzulesen. Der MontiCore-Ansatz wird erneut in Kapitel 7 aufgegriffen.

2.6 Grundlagen zu Integralen Sicherheitssystemen

Der Schwerpunkt dieser Arbeit wird sich auf Integrale Sicherheitssysteme bzw. auf aktive Sicherheitssysteme konzentrieren. Daher wird zunächst allgemein Bezug auf die Fahrerassistenzsysteme genommen, um dann davon die integralen Sicherheitssysteme abzuleiten und an wel-

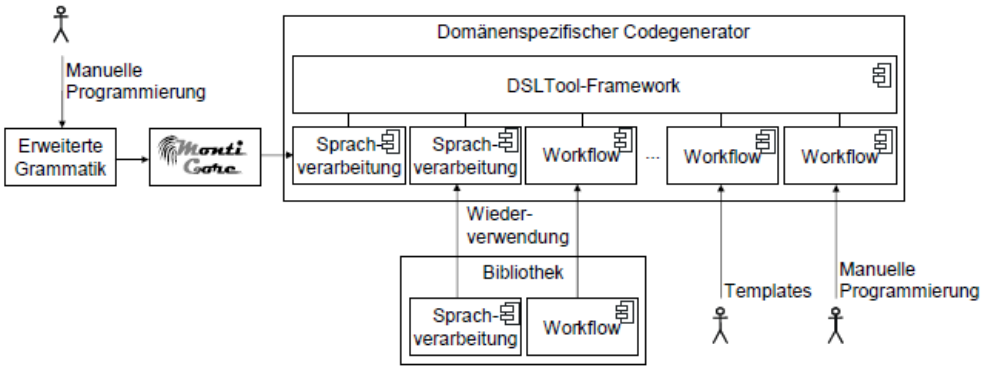


Abbildung 2.23: Codegenerierung mit MontiCore nach Krahn [Kra10]

cher Stelle in einem potentiellen Unfallgeschehen sie zum Einsatz kommen können. Abschließend werden dann Notbrems- und Warnsysteme im Konkreten als zugrundeliegendes System in ihrer Funktionsweise beschrieben.

2.6.1 Allgemeines zu Fahrerassistenzsystemen

Fahrerassistenzsysteme im Allgemeinen sollen den Fahrer in allen denkbaren Verkehrssituationen ein entspanntes, stressfreies und unfallfreies Fahren ermöglichen und können in verschiedene Kategorien eingeordnet werden, je nach Gebiet ihrer Anwendung:

- Fahrerinformationssysteme
- Fahrzeugkommunikationssysteme
- Fahrerassistenzsysteme zur Stabilisierung
- Prädiktive Fahrerassistenzsysteme (Komfortdimension)
- Fahrerassistenzsysteme zur Erkennung des Fahrzeugzustandes und der adaptiven Beeinflussung (Sicherheitsdimension)

Bei den Fahrerinformationssystemen liegt das Hauptaugenmerk auf der Darstellung von Informationen mit Bezug auf die Fahrzeugführung über das Mensch-Maschine-Interface wie z.B. Informationen über die aktuelle Route, Fahrzeugzustandsinformationen oder auch Verkehrsmeldungen. Bei den Fahrzeugkommunikationssytemen liegt der Schwerpunkt auf der Kommunikation zwischen Fahrzeugen sowie zwischen Fahrzeugen und der umliegenden Verkehrsinfrastruktur. Bei den stabilisierenden Fahrerassistenzsystemen handelt es sich vorwiegend um solche,

welche den direkten Fahrzeugzustand mittels interner Sensorik erfassen und im Falle einer definierten Instabilität korrigierend in die Fahrzeugführung eingreifen. Dabei ist die direkte Umfelderfassung zunächst noch unbedeutend. Typische Beispiele hierfür sind das ABS, ASR und das ESP. Durch die prädiktiven Fahrerassistenzsysteme wird mittels Umfeld erfassender Sensorik wie Radar, Kamera oder auch Laser zur Regelung der Quer- und Längsführung umgebende Objekte in Bezug zur eigenen Fahrzeugbewegung gesetzt. Dabei ist die Unfallwahrscheinlichkeit in den konkreten Verkehrssituationen vergleichsweise gering. Anders als bei der letzten Kategorie, die eine Kombination aus der dritten und vierten Kategorie darstellt, ist die Wahrscheinlichkeit eines Unfalls deutlich erhöht. [WR10]

2.6.2 Integrale Sicherheitssysteme

So gilt es grundsätzlich zwischen Komfort- und Sicherheitssystemen zu unterscheiden, wobei die Wirkungsdimension die entscheidende Rolle spielt: Die Komfortsysteme erhöhen das Wohlbefinden des Fahrers während bei Sicherheitssystemen die körperliche Unversehrtheit im Vordergrund steht. Darüber hinaus können Sicherheitssysteme je nach Abhängigkeit des Unfallverlaufes in aktive und passive Sicherheitssysteme unterteilt werden. [Jor06, S. 1] Integrale Sicherheitssysteme bezeichnen demnach eine Kombination aktiver und passiver Maßnahmen, die für bestimmte Phasen des Unfallverlaufs ausgelegt sind, um im Idealfall Unfälle ganz zu vermeiden oder die Folgen eines Unfalls zu verringern. Der integrale Ansatz erlaubt es, eine Gesamtstrategie zum Schutz der beteiligten Verkehrsteilnehmer zu entwickeln und nicht nur allein die Auswirkungen eines Anpralls in Betracht zu ziehen.

Der zeitliche Verlauf eines Unfalls kann dabei modellhaft in verschiedene Phasen gegliedert werden. Das nachfolgende Phasenmodell stammt vom Dachverband der europäischen Automobilhersteller (ACEA):

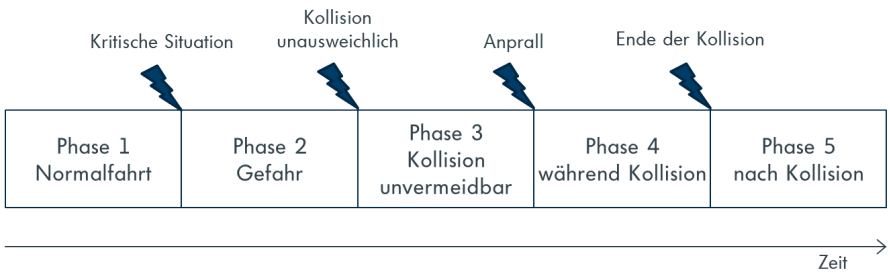


Abbildung 2.24: Das Phasenmodell eines Unfalls in Anlehnung an [Han08, S. 3]

Ein Unfall findet seinen Ursprung immer in einer Normalfahrt (Phase 1). Tritt eine kritische Si-

tuation auf, bspw. wenn ein vorausfahrendes Fahrzeug bremst oder ein Fußgänger in den Fahrschlauch des Fahrzeuges tritt, so schließt sich die Phase 2 “Gefahr” an. Sie ist geprägt davon, dass eine Kollision entweder durch Verzögern, Beschleunigen oder Ausweichen noch vermieden werden kann. Solche Situationen werden auch als Beinahe-Unfälle betitelt.

Mit zeitlichem Verlauf und ohne Reaktion auf die kritische Situation wird eine Kollision zu einem bestimmten Zeitpunkt nicht mehr vermeidbar, so dass die Phase 3 beginnt. Hier steht dann definitiv fest, dass es zu einer Kollision kommen wird.

Mit dem Anprall folgt dann die Phase 4 “während Kollision”, in der üblicherweise die auftretenden Energien durch Deformationen und Impulsübertragungen abgebaut bzw. weitergegeben werden. Speziell in dieser Phase erleiden die beteiligten Personen leichte oder schwere Verletzungen.

Mit Ende der Kollision beginnt die letzte Phase “Nach Kollision”, in welcher bei schweren Unfällen die Maßnahmen zum Retten und Bergen der verunglückten Person durchgeführt werden. [Han08, S. 3ff]

Integrale Sicherheitsfunktionen setzen nun in den einzelnen Phasen durch Kombination aktiver und passiver Maßnahmen an, um den Auswirkungen von Unfällen entweder vorzubeugen oder abzumildern. Dabei gilt: Je früher eine Sicherheitsfunktion einsetzt, desto größer ist der Einfluss auf die Schwere der Kollision am Ende des Unfallgeschehens. Nachfolgend sei eine solche Funktion näher erläutert.

2.6.3 AEB/FCW-Systeme

Bei einem AEB/FCW-System handelt es sich um einen Bremsassistenten für potentiell hochgefährliche bzw. Notsituationen zwischen Verkehrsteilnehmern, in denen das System die Situation über Sensorik erfasst. Mittels eines Algorithmus’ wird die Gefährlichkeit bestimmt und je nach Kritikalität der Fahrer entsprechend informiert oder über eine Aktorik ein autonomer Bremseingriff für das Fahrzeug vorgenommen. Das Wirkprinzip entspricht im wesentlichen einer dreistufigen Eskalation einer Gefahrensituation in Abhängigkeit der Kritikalität eines Objektes für den eigenen Bewegungsverlauf:

1. Stufe: In der ersten Stufe wird bei bei noch geringer Gefährlichkeit über optische und/oder akustische Warnungen vor dem Hindernis aufmerksam gemacht, so dass der Fahrer selbsttätig auf die Situation reagieren kann. Insbesondere wird die Bremsanlage hierbei schon durch eine Bremsdruckerhöhung oder eine Annäherung der Bremsbeläge an die Bremsscheibe vorkonditioniert.

2. Stufe: Sofern der Fahrer auf diese Maßnahmen nicht näher reagieren sollte, wird in der zweiten Stufe zusätzlich haptisch vor der Gefahr gewarnt, üblicherweise in Form eines kurzen Bremsrucks. Unter Umständen unterstützt das System dann den Fahrer zusätzlich für die notwendige Bremsverzögerung.

3. Stufe: Sollte der Fahrer auch hiernach nicht weiter reagieren, so leitet das Bremssystem autonom eine Vollbremsung mit einer hohen Verzögerungsrate ein. Dabei ist entscheidend, das spätestens ab diesem Zeitpunkt eine Unfallvermeidung durch einen normal trainierten Fahrer nicht mehr als möglich angesehen wird. [GSSZ13]

Im Folgenden sei nun auf die drei Hauptkomponenten Sensorik, Algorithmik und Aktorik eingegangen.

2.6.4 Sensorik

Es gibt im wesentlichen folgende Typen von Sensoren für Fahrzeuge, die der Umfelderfassung dienen:

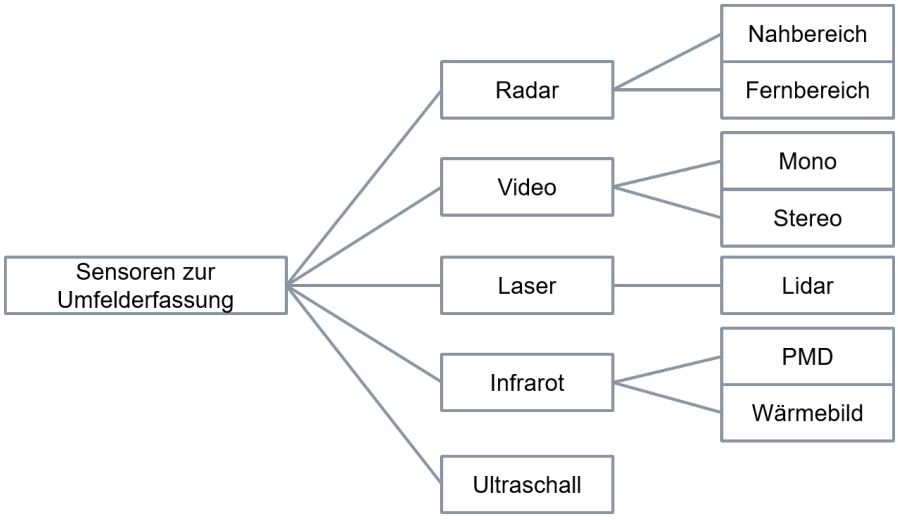


Abbildung 2.25: Übersicht von Sensoren zur Umfelderfassung.

Die einschlägige Literatur behandelt die physikalischen und technischen Details der einzelnen Typen und wird meist ausführlich in anderen Dissertationen dargestellt. In dieser Arbeit ist jedoch der konkrete, technische Aufbau und das komplexe, physikalische Wirkprinzip nur von nachrangiger Bedeutung. Vielmehr ist der Modellcharakter und die Modellbildung in Verbin-

dung mit einer abstrakten Modellwahl entscheidender, so dass eine wesentliche Eigenschaft hervorgehoben sei, die je nach Sensor mehr oder weniger stark zum Tragen kommt. [LKS+13]

Funktionale Unzulänglichkeit

Funktionale Unzulänglichkeit wird insbesondere in Verbindung mit Sensorik wie z.B. mit Radar und Video als gängigste verbaute Sensorik zur prädiktiven Umfelderfassung genannt. Dies bedeutet, dass die Sensorik nach dem jeweilig aktuellen Stand der Technik nur zu einem gewissen Grad in der Lage ist, Objekte im Straßenverkehr korrekt zu identifizieren und ihnen gewisse Eigenschaften und Attribute mit hoher Genauigkeit gegenüber dem tatsächlichen Zustand des Objektes zuzuweisen. Diese Attribute umfassen u.a. Position, Orientierung, Bewegungsverlauf über die Zeit, um nur die wichtigsten aufzuführen, die für eine spätere Bestimmung der Kritikalität für die eigene Fahrzeugbewegung genutzt werden. Jedoch gewährleisten die heutigen Sensoren entsprechende Daten mit einer hinreichenden Genauigkeit, um einen entscheidenden Beitrag zur Unfallvermeidung bzw. -folgenmilderung zu leisten.

Arten von Objekten

Sensoriken wie Radar oder Kamera müssen verschiedene Arten von Objekten erkennen und ihre Position und ihre Bewegung bestimmen können. Dazu zählen in erster Linie:

Fahrzeuge: Sie sind vergleichsweise einfach aufgrund ihrer Größe und ihrer Karosserie, die vorwiegend aus Metall besteht, speziell durch Radar-Sensorik zu erkennen. Ihr Bewegungsverlauf ist üblicherweise in Sachen Beschleunigungsfähigkeit und Richtungsänderung relativ stabil zu prognostizieren, so dass die Vorhersage-Fähigkeit der Bewegung als stabil bezeichnet werden kann. Aktuell werden solche Objekte erkannt und weiter verarbeitet, die sich in die gleiche Richtung wie das eigene Fahrzeug bewegen.

Fußgänger: Sie sind von den Abmessungen her deutlich kleiner, können kurzfristig hohe Beschleunigungswerte auch bei niedrigen Geschwindigkeiten erreichen, wie beispielsweise spontanes Loslaufen oder Stoppen, wodurch die Komplexität der Bewegungserfassung und Vorhersage-Fähigkeit deutlich höher liegt. Auch die materielle Beschaffenheit von Fußgängern erschwert die Erfassung durch Radarsensorik aber auch durch Kameras, die häufig über Kontrastunterschiede Objekte identifizieren. Außerdem verändert sich aufgrund der Bewegung von Armen und Beinen die Siloutte von Fußgängern, welches zusätzlich Berücksichtigung finden muss.

Radfahrer: Radfahrer haben jedoch ein größeres Abmaß, weisen jedoch ähnlich wie zum Fußgänger ein hohes dynamisches Potential hinsichtlich Geschwindigkeitsveränderung auf. Jedoch ist hier nach aktuellem Stand der Technik offen, zu welchem Grad die heutigen Sensoren die Vorhersagbarkeit und Erkennungsleistung von Radfahrern annehmen können. [GSSZ13] Beispielsweise bieten Autohersteller wie Volvo bereits Systeme mit Radfahrererkennung an, weisen allerdings die Einschränkungen der Erkennungsleistung direkt aus. Dazu gehört z. B. die unzuverlässige Erkennung bei Dämmerung, die Deaktivierung bei Dunkelheit, Beschränkung auf Längsverkehr, usw. [Vol19d] Weiterhin hat die Continental AG im Jahr 2019 einen Radarsensor auf 77 GHz Basis vorgestellt, der insbesondere bei Abbiegeszenarien entsprechende Verkehrsteilnehmer Radfahrer, aber auch Fußgänger erkennen können. [Rae19]

2.6.5 Algorithmik

Ein grundlegendes Konzept zur Konstruktion von Algorithmen und eine Sammlung von Ansätzen über ihre Funktionsweise gestaltet sich allein deshalb schon schwierig, da es sich bei diesem Wissen um die Expertise vieler OEMs und Zulieferunternehmen handelt. Jedoch seien hier die wesentlichen Aspekte genannt, die bei der Konstruktion berücksichtigt werden müssen. Dabei wird davon ausgegangen, dass die Information aus der Sensorik valide sind. Wie bereits zuvor erläutert, ist diese Annahme vereinfachend anzusehen, da die funktionale Unzulänglichkeit eine hundertprozentige Genauigkeit bei der Bestimmung der Objekteigenschaften nicht erlaubt.

Zwei Berechnungen müssen zunächst zur Bestimmung der Kritikalität eines Objektes für die eigene Fahrspur durchgeführt werden:

1. Bestimmung des momentanen Zustands
    - der Umgebungsobjekte (aus der Sensorik)
    - des eigenen Fahrzeugs
2. Prädiktion des Zustands
    - der Umgebungsobjekte
    - des eigenen Fahrzeugs

Dies ist aufgrund der mit einer gewissen Wahrscheinlichkeit behafteten Schätzung über die Bewegung nur begrenzt und mit einer entsprechenden Ungenauigkeit verbunden. Der aktuelle Stand der Technik unterstreicht jedoch, dass dies auf einem hohen Niveau zuverlässig erfolgen

kann. Zusätzlich können weitere Aspekte in diese Berechnung einfließen, wie z.B. die Fahrerzustandsschätzung (aktiv/inaktiv), da sich hieraus ein anderes Reaktionsverhalten des Fahrers resultiert. Darauf aufbauend können je nach Umgebungssituation auch Ausweichmöglichkeiten eingebunden werden, so dass die Kritikalität eines Objektes genauer bestimmt und damit robuster gegenüber Fehlauslösungen gestaltet werden können.

Beim Algorithmus wird eine Sicherheitszone vor dem Fahrzeug angenommen, die dazu dient, kritische Objekte von unkritischen Objekten zu unterschieden. Befindet sich der prognostizierte Aufenthaltsort eines Objektes, für das selbst auch ein solcher Sicherheitsbereich angenommen wird, mit einer hohen Wahrscheinlichkeit innerhalb dieses Bereiches, ist das Objekt für den eigenen Bewegungsverlauf des Fahrzeugs kritisch und es besteht die unmittelbare Gefahr einer Kollision. Mit zunehmender Projektion der Bewegung in die unmittelbare Zukunft werden diese Sicherheitsbereiche jedoch in ihren Abmessungen größer, so dass sich eine zunehmende Ungenauigkeit über den möglichen Aufenthaltsort ergibt. Daher muss erst eine gewisse Stabilität in der Projektion über Zeit erfolgen, so dass das aktive Sicherheitssystem auch kontinuierlich diesen Bewertungsprozess vorzunehmen hat. [LKS+13]

In Abhängigkeit von der Bewertung über die Kritikalität eines Objektes gilt es zudem die Maßnahmen festzulegen, wie reagiert werden soll. Üblicherweise gibt es mehrere Kritikalitätsstufen in Abhängigkeit von Abstand und Geschwindigkeit des ausgerüsteten und des vorausfahrenden Fahrzeugs. Im Rahmen einer definierten Wirkstrategie entscheidet das System, welchen Bremsdruck es beispielsweise anfordert. Auch hier können unterschiedliche Funktionen betrachtet werden:

1. Mit Bremseingiff des Fahrers,
2. ohne Bremseingriff des Fahrers.

Bei ersterem “überwacht” das System die Aktionen des Fahrers und erhöht selbsttätig die geforderte Verzögerungsrate, falls die durch den Fahrer initiierte Rate nicht ausreichend ist. Die Funktion entscheidet bei zweiterem unabhängig vom Eingriff durch den Fahrer und agiert entsprechend der Wirkstrategie für die jeweilige Kritikalitätsstufe. (vgl. ähnliche Darstellung in [GSSZ13]).

2.6.6 Aktorik

Im Rahmen dieser Arbeit ist eine vertiefende Sicht hinsichtlich Funktionsweise, Arten von Bremsen und technischer Realisierung nicht von tragender Bedeutung. Dort, wo konkrete Geschwindigkeitsreduzierungen relevant sein werden, wird abstrahiert.

2.7 Zusammenfassung

Das Kapitel 2 hat die wichtigsten Konzepte, Grundlagen und relevanten Besonderheiten der Disziplinen Software Testing, Modellbildung und Simulation Fahrzeugbau und Fahrzeugtechnik und des Software-Engineerings dieser Arbeit vorangestellt, um einerseits einen Einstieg für Leser aus der jeweils anderen Fachrichtung anzubieten und andererseits auch das inhaltliche Fundament zu definieren, welches für die weiteren Kapitel wichtig sein wird. Das sich nun anschließenden Kapitel 3 und 4 werden sich näher mit den Fahrzeugfunktionen speziell auch von Aktiven Sicherheitssystemen befassen und welche Rolle die weltweiten etablierten Consumer-Tests spielen.

3 Allgemeine Ansätze zur Entwicklung und zum Test von Fahrzeugfunktionen

Dieses Kapitel behandelt zunächst in allgemeiner Form, wie im zeitlichen Kontext des Projektes Fahrzeugfunktionen getestet und abgesichert werden. Häufig wird der Schwerpunkt dabei auf die spätere Feldperformance gelegt, um eine bestmögliche Funktion vor Kunde zu gewährleisten. Ergänzend dazu spielen aber auch die Consumer Tests bei diversen Testinstitutionen eine gewichtige Rolle, die ebenfalls in diese Entwicklung einfließen müssen. Da speziell diese Consumer Tests besonderen Randbedingungen wie z.B. bestimmte Toleranzbereiche zu einzelnen Testparametern unterliegen und außerdem diese Tests nicht selten auch die Grenzen der jeweiligen Systeme austesten sollen, sind sie eine besondere Herausforderungen für die Entwicklung. Unbeantwortet bleibt jedoch die Frage in diesem Zusammenhang, welche Konzepte und Methoden hierfür und bezogen auf die Consumer Tests existieren. Diese Frage wird explizit im Kapitel 4 nachgegangen.

Daher werden zunächst hier diejenigen Bedingungen beschrieben, die einerseits für die erfolgreiche Absolvierung der Consumer Tests gelten und andererseits für die Volkswagen AG im Rahmen ihres individuellen Entwicklungsprozesses von Bedeutung sind. Diese Bedingungen stellen auch den konkreten Projektkontext dar, in welcher diese Arbeit eingebettet ist. Im Kapitel 4 wird dann unter Berücksichtigung dieser Projektbedingungen die momentan allgemein zugängliche und zentral organisierte Literatur auf aktuelle Ansätze, Konzepte und Methoden geprüft, die relevant für die weitere Bearbeitung des Themas sind. Um speziell diese Recherche zu systematisieren, wird hierzu die Methode von Kitchenham zum “Systematic Literature Review” herangezogen. Zwar erfolgte die Einführung der Consumer Tests für Aktive Sicherheitssysteme erst mit dem Beginn des Jahres 2014; aber auch im Zuge von Vorveröffentlichungen konnten nur begrenzt konkrete Beiträge zu dieser Thematik identifiziert werden. Die Entscheidung zu Gunsten dieser Methode ist dem Ziel geschuldet, einen möglichst hohen Erfassungsgrad an alternativen Beiträgen zur eigenen skizzierten Methodik zu erhalten.

3.1 Entwicklung und Absicherung von Fahrzeugfunktionen

Es ist zweifellos einleuchtend, dass unentdeckte Fehler in eingebetteten Systemen erhebliche Konsequenzen nach sich ziehen, wenn sich ihre Wirkung vor Kunde entfaltet und damit unmittelbare Auswirkungen auf die Sicherheit von Fahrzeug und Fahrer hat. Daher ist es nicht unüblich, dass mittlerweile 25 bis 50% der Entwicklungskosten auf den Test und die Absicherung von neuen Systemen entfallen. [SL07, S. 2] Insbesondere die immer häufigere Einbeziehung und Erfassung des Umfeldes durch Sensorik erschwert die gesicherte Abnahme neuer Funktionen und Systeme. Dies resultiert daraus, dass das Umfeld praktisch eine unendliche Anzahl an möglichen Objektkonstellationen erlaubt und der Entwicklungsgrad der Technik für die genaue Erfassung und eindeutige Interpretation durch die Sensorik momentan noch Grenzen gesetzt sind.

Der besondere Stellenwert des Freigabeprozesses wird somit in Zukunft noch weiter wachsen, gerade weil die Komplexität der eingebetteten Systeme stetig zunehmen wird. Bisher sind gerade im Automobilsektor diese Test- und Freigabekonzepte von realen Fahrten mit den jeweiligen Komponenten und Systemen geprägt.

Hierzu werden meist im Rahmen des Integrationsstufenmanagements (ISM) verschiedene Softwarestände mit unterschiedlicher Funktionalität schrittweise ins Fahrzeug integriert. Das ISM dient dazu, die Implementierungsumfänge relativ gleichmäßig auf den gesamten Entwicklungszeitraum zu verteilen, so dass nicht am Anfang kaum Umfänge umgesetzt und eine Vielzahl am Ende des Prozesses integriert werden.

Die Applikation einer Funktion auf ein konkretes Fahrzeugprojekt ist ein komplexer Prozess, der immer noch überwiegend durch reale Testfahrten realisiert wird. Dabei werden Anpassungen von unterschiedlichen Parameterwerten in den verschiedenen Komponenten vorgenommen, damit die Funktion im Sinne des Entwicklers eine optimale Performance bietet. Bei diesen Parametern handelt es sich um konkrete Werte, welche innerhalb einer Komponente als Vergleichsgrößen dienen und in Kombination mit Signalwerten aus anderen Komponenten über den Zustand des Fahrzeuges oder des Umfeldes gewisse Entscheidungen treffen. Zum Beispiel kann durch eine Anpassung des Parameters “Time-To-Collision” (TTC) für Warn- und Bremsentscheidungen eine frühe oder spätere Auslösung eingestellt werden. Eine frühestmögliche Warnung oder Bremsung ist im Sinne der Unfallverhinderung unabdingbar; dennoch muss dies im Kontext einer Deaktivierung durch den Fahrer gesehen werden, der sich durch die frequentierte Auslösung einer Funktion in seiner Fahrzeugführung bevormundet fühlt. Daher sind im Rahmen der Applikation eine Vielzahl an Experten über alle Hierarchiestufen eingebunden, so dass durch die vielen subjektiven Eindrücke und objektiven Messwerte über die Performance eine Entscheidung zur Freigabe erzielt wird. Diese Art des Freigabeverfahrens ist auch nicht ersetzbar durch rein

objektive Bewertungsmethoden, die sich anhand konkreter Metriken formal beschreiben lassen. Dennoch kann ein Potential darin gesehen werden, beide Verfahren in Ergänzung zu nutzen und weiterzuentwickeln.

Bereits 2007 hat der TÜV Süd hier eine Bewertungsmethode durch reale Fahrversuche weiterentwickelt, mit welchen das Fahrerassistenzsystem ACC genauer untersucht werden sollte. Dabei war man auch zu der Auffassung gelangt, dass analog zur Bewertung von Fahrdynamik subjektive und objektive Bewertungsmethoden einander ergänzen sollten. Als objektiv wird in diesem Zusammenhang die Bewertung anhand von Kennwerten bezeichnet, während subjektiv auf die funktionale, Kunden relevante Gesamtbeurteilung bezogen ist, hier als Verhalten vor Kunde bezeichnet. [SBBM07, S. 414ff]

In der späteren Erprobung wurde deutlich, dass allein durch die Vielzahl der unterschiedlichen Sensoren, Inertialsysteme, Fahrroboter zur Gewährleistung der Reproduzierbarkeit, etc. das Bewertungsverfahren eine nicht zu vernachlässigende Komplexität aufweist, die zunächst jedoch für die Funktion ACC händelbar erscheint. Allerdings werden zukünftige Funktionen an Komplexität weiter zunehmen, so dass langfristig ein solches Verfahren den Testkostenaufwand enorm steigern wird. Insbesondere ist die Reproduzierbarkeit ebenso als wichtiger Bestandteil für die Freigabe von neuen Fahrerassistenzfunktionen zu erachten.

Winner et al. behandeln in ihrem Beitrag ebenfalls einen nicht zu vernachlässigenden Aspekt in Hinblick auf die Freigabe für das autonome Fahren von Fahrzeugen und kommen zu dem Schluss, dass für solche Funktionen neue Test- und Freigabeprozesse erforderlich sind, da die Randbedingungen der gegenwärtigen Testverfahren auf die neuen Funktionen nicht mehr anwendbar sind. Somit könnte der mangelnde Freigabeprozess für das autonome Fahren durchaus das Potential eines “Show-Stoppers” aufweisen. Denn mit gegenwärtigen Verfahren können Kosten für den Test eines funktionsrelevanten Szenarios bspw. auf der Autobahn schnell mehrere 100 Millionen Euro verursachen. [WWW10, S. 17ff]

Gleichzeitig verringern sich mit Einführung validierter Funktionen die an die Bewertung herangezogenen Referenzwerte für das Unfallrisiko, da mit ihrer Etablierung jenes Risiko weiter abgesenkt werden kann. Im Umkehrschluss verlängert sich jedoch die Testdauer für die nächste Funktionsgeneration und auch die erforderlichen Kosten nehmen zu. Dieser Zusammenhang wird von Winner et al. als Testdilemma bezeichnet. [WWW10, S. 21]

Die Fahrerassistenzsysteme mit integralen Sicherheitsfunktionen bilden hierbei eine Zwischenstufe zu den autonom agierenden Fahrzeugen, weil sie im Falle einer drohenden Kollision autonom in die Fahrzeugführung eingreifen sollen, gleichzeitig aber die Übersteuerung durch den Fahrer (theoretisch) ermöglichen. Aufgrund des engen Zeitfensters von z.T. unter einer Sekunde

bis zur Kollision ist das Aktionsfeld des Fahrers jedoch stark eingeschränkt, so dass der Mensch als letzte Entscheidungsinstanz fraglich erscheint. [WWW10, S. 20]

Es lässt sich nun argumentieren, dass dieses begrenzte Zeitfenster auch die Entscheidungsvarianten begrenzt, so dass durch die vermeintlich geringe Komplexität eine Reproduzierbarkeit der Szenarien gesichert wäre. Selbst wenn es sich heute als praktikabel erwiesen hat, solche Funktionen in erster Linie subjektiv durch reale Testfahrten einer Bewertung zu unterziehen und damit auf Simulationsmethoden verzichten zu können, so wird sich langfristig diese Vorkollisionsphase zeitlich immer weiter ausdehnen und die zu erfassende Szenerie an Komplexität gewinnen. Gleichzeitig orientieren sich die Aufgaben dieser "Collision Mitigation Systems" (CMS) komplementär zu den eigentlichen Fahrerfähigkeiten, so dass der Anteil autonomer Eingriffe auch neue Methoden mit Metriken erforderlich macht. [WWW10, S. 20]

Um dieser Situation im Rahmen des Test und der Absicherung langfristig gerecht zu werden, müssen neue Strategien entwickelt und etabliert werden. Eine Möglichkeit bilden dabei virtuelle Simulationsmethoden in einer entsprechenden Testumgebung, mit dem allgemeinen Ziel den Aufwand für reale Testfahrten zu verringern. Das Beispiel der Finiten-Elemente-Methode führt auf, wo die Entwicklung solcher Werkzeuge hingehen kann: FEM erlaubt die Festkörpersimulation von bspw. Karosserieteilen des Fahrzeuges, so dass viele Tests und Absicherungen in diesem Bereich zunächst erst am Rechner durchgeführt werden, um schließlich anhand vereinzelter, real durchgeführter Tests die getroffenen Annahmen zu validieren und die Eigenschaften sowie Freigabebedingungen nachzuweisen.

Allerdings ist festzuhalten, dass ein immenser Unterschied darin besteht, dass bei FEM fast alle Informationen zu den Problemstellungen in Form von Anfangs-, Rand- und Übergangsbedingungen zu einem großen Teil bekannt und als Differenzialgleichungen darstellbar sind. Zwar werden zu gewissen Aspekten der Simulation auch Annahmen getroffen, jedoch lassen viele der Anfangs- und Übergangsbedingungen numerisch auflösen. Die Randbedingungen für ein Fahrerassistenzsystem sind meist jedoch nur unvollständig aufzuschlüsseln und schlecht zu klassifizieren, so dass einerseits das System und andererseits die Simulation mit unvollständiger Information in deutlich größerem Umfang als bei FEM umgehen bzw. abbilden müssen. Weiterhin sind manche Randbedingungen aufgrund ihrer physikalischen Eigenschaften soweit komplex, dass eine mathematische Modellierung und insbesondere die Komposition der Randbedingungen nur unvollständig erfolgen können (z. B. Entscheidungsprozesse anderer Verkehrsteilnehmer). Daher ist die Übernahme solcher Simulationsmethoden aus dem Bereich der FEM im Detail unwahrscheinlich.

Weitere etablierte Testverfahren gerade im Bereich der Entwicklung eingebetteter Systeme se-

hen vor, dass jeweilige Testobjekt “in-the-loop” zu nehmen. Sie kommen bisweilen auch auf den jeweiligen Abstraktionsstufen des V-Modells zum Einsatz, wobei diese nicht zwangsläufig nur auf eine Stufe beschränkt sind. Vielmehr wird in der Literatur wie z.B. bei [Nie05] eine gewisse Durchgängigkeit der einzelnen “in-the-Loop” Verfahren postuliert. Dabei handelt es sich um

- Model-in-the-loop (MiL)
- Software-in-the-loop (SiL)
- Processor-in-the-loop (PiL)
- Hardware-in-the-loop (HiL)

Beim MiL wird die Anforderungsspezifikation zunächst mit Hilfe einer graphischen Programmiersprache, z. B. Matlab/SimuLink oder der Unified Modeling Language (UML) umgesetzt, aus denen sich Zustandsautomaten implementieren lassen. Anhand der dazugehörigen Testspezifikationen werden dann entsprechende Testfälle entwickelt, so dass das Modell entsprechend validiert werden kann. Hierzu ist ein geeignetes Umgebungsmodell notwendig, welches jedoch einen hohen Abstraktionsgrad aufweisen darf, damit Sensoren und Aktuatoren nicht aufwendig modelliert werden brauchen. Zum Testen wird daher auch keine besondere Hardware benötigt. Sobald alle Anforderungsspezifikationen und zugehörige Testspezifikationen umgesetzt sind, liegt ein ausführbares Lastenheft vor.

Beim SiL-Verfahren wird idealerweise das Modell der Funktion direkt in ausführbaren Code für ein späteres Steuergerät überführt. Dies kann entweder manuell oder aber auch durch Generierung erfolgen. Wenn kein Code direkt vorliegt, weil die Funktion von einem Zulieferer entwickelt wird und dieser sein Know-How schützen möchte, kann das Modell zumindest noch zur Testfallgenerierung herangezogen werden. Zudem kommen auch hier anfänglich keine Hardwarekomponenten zum Einsatz, so dass die Tests viel schneller als in Echtzeit durchgeführt werden können, da hierdurch die Testausführung auf vielen Rechnerknoten parallelisierbar ist. Sollte es sich um eine Software-Eigenentwicklung handeln, können in Ergänzung auch Whitebox-Tests durchgeführt werden.

Das PiL-Verfahren ist als eine Zwischenstufe zum HiL-Verfahren, da der für den Zielprozessor kompilierte Code und der Prozessor selbst im Fokus der Tests stehen, so dass zunächst nur ein spezielles Testboard verwendet wird. Mit Beginn dieser Tests liegt das Testobjekt üblicherweise nur noch als Blackbox vor, so dass Validierungen nur noch in Form von Ein- und Ausgabedaten durchgeführt werden können.

Das HIL-Verfahren, entweder als einzelnes Steuergerät oder auch als Gesamtverbund, sieht Tests immer mit der dazugehörigen Hardware vor, so dass das Umfeldmodell die nötigen Rohdaten

für das Testobjekt erzeugt. Dies impliziert auch, dass das System harte Echtzeitanforderungen an die Kommunikation der Komponenten untereinander stellt. [Nie05, S. 13ff]

Zudem ist insgesamt die Serienentwicklung gegenwärtig stark durch Beauftragung entsprechender Zulieferer geprägt, so dass ohne vorherige Absprache und Übereinkunft mit dem Auftragnehmer die Softwarekomponenten nur als direkt für das Steuergerät kompilierter Code vorliegen. Dies führt ebenfalls dazu, dass überwiegend nur Blackbox-Tests durchführbar sind. Unter gewissen Umständen kann sich der Zulieferer aus verschiedenen Gründen in der Bereitstellung seines späteren Seriencodes für andere Targets zurückhaltend äußern. Dieser seltenen, aber ungünstigen Situation ist ebenfalls Rechnung zu tragen, um gegebenenfalls Lösungen für das Emulieren von Steuergräten zu entwickeln. Weiterhin ist davon auszugehen, dass in nicht allzu ferner Zukunft die Funktionseigenentwicklung vermehrt in den Vordergrund treten wird, um sowohl den Abfluss von Know-How zu vermeiden als auch eine verbesserte Testtiefe zu erlangen. (vgl. hierzu [LK19])

3.2 Test und Absicherung von Aktiven Sicherheitssystemen in Consumer-Tests

Im vorangegangen Abschnitt ist dargestellt worden, wie Fahrzeugsysteme allgemein entwickelt werden und welche Methodiken sich als Best-Practices in die Entwicklung von Fahrerassistenzsystemen verfestigt haben. Bevor sich in Kapitel 4 konkret neueren Methodiken gewidmet wird, welche speziell mit Blick auf Aktive Sicherheitssysteme entwickelt und ggf. in einem industriellen Kontext etabliert sind, werden nun abschließend für dieses Kapitel die projektspezifischen Randbedingungen wie z.B. die Consumer-Test-Protokolle aufgeführt. Dabei wird sich bewusst für eine ältere Version des Test Protokolls entschieden, damit eine Veröffentlichung der Ergebnisse auch außerhalb des Unternehmens Volkswagen möglich ist und somit keine vertraulichen Erkenntnisse das Unternehmen verlassen. Danach wird in Kapitel 5 mit der Entwicklung einer Methodik zur zielorientierten Konzeption und Verwendung einer Simulationsumgebung für Aktive Sicherheitssysteme begonnen.

3.2.1 Die Consumer-Test-Organisation EuroNCAP

Die ersten Bestrebungen für die Bewertung des Sicherheitsniveaus von Fahrzeugen geht auf die 1970er Jahre zurück, in denen durch mehrere europäische Regierungen, organisiert im “European Experimental Vehicles Committee (EEVC)” an konkreten Bewertungsverfahren für ver-

schiedene Sicherheitseigenschaften von Fahrzeugen gearbeitet wurde. Allerdings konnte dieses Verfahren erst in den frühen 1990er Jahren durch verschiedene Crash Test Prozeduren für Front- und Seitenbereich auf einen Stand gebracht werden, der als Gesetzesentwurf auf europäische Ebene vorgelegt wurde. Aufgrund von Unstimmigkeiten zwischen der EEVC und der Automobilindustrie zog das UK Department of Transport ein eigenes NCAP in Betracht, welches nach einer gewissen Zeit auf ganz Europa hätte ausgeweitet werden können, wobei das Programm selbst umfangreicher als der Gesetzentwurf gewesen wäre. [Eur14]

Die ersten Tests beschränkten sich zunächst nur auf die Supermini Fahrzeuge und man bat die Automobilhersteller der Fahrzeuge um weitere Informationen. Da man jedoch langfristig auch andere Fahrzeugklassen testen wollte, musste der Standard deutlich höher als im Vergleich für eine Gesetzgebung angesiedelt werden, so dass ein detailliertes Test Protokoll entwickelt wurde, der diesen Standard sichern sollte. Ein besonderer Wert wurde dabei auf die Wissenschaftlichkeit und die Validität gelegt, welche man durch die Einbindung von verschiedenen Fachexperten gewährleistete, um später eine Vergleichbarkeit zwischen den verschiedenen Fahrzeugen zu ermöglichen. Gegen Ende des Jahres 1996 traten die Swedish National Road Administration, die Federation Internationale l'Automobile (FIA) und International Testing dem Programm bei, woraus letztlich EuroNCAP hervorging. Die konstituierende Sitzung erfolgte dann im Dezember 1996. [Eur14]

Nach der Veröffentlichung mehrerer Testberichte zu Fahrzeugen schlossen sich weitere Organisationen und Regierungen in Europa EuroNCAP an. Folgende Mitglieder zählen heute zum Konsortium, dessen Hauptsitz in Brüssel, Belgien liegt und belgischem Recht unterliegt:

- Allgemeiner Deutscher Automobil Club e.V. (ADAC)
- Bundesministerium für Verkehr, Bau und Stadtentwicklung
- UK Department for Transport (DfT)
- Dutch Ministry of Transport, Public Works and Water Management
- Département des Transports
- Generalitat de Catalunya
- International Consumer Research and Testing
- FIA
- Swedisch Transport Administration Trafikverket
- Thatcham

- Ministère de l'Ecologie, du Développement durable et de l'Energie
- Automobile Club d'Italia

Das Ziel von EuroNCAP besteht darin, motorisierten Konsumenten eine realistische und unabhängige Bewertung hinsichtlich der Sicherheitsperformance derjenigen Fahrzeuge anzubieten, die in Europa am meisten verkauft werden. Dabei versucht EuroNCAP durch eine weitere Verfeinerung und Weiterentwicklung der Test Prozeduren einen Katalysator für weitere Verbesserungen der Fahrzeugsicherheit zu bilden. [Eur14]

Das Bewertungsverfahren umfasst seit 2009 vier Bereiche, die sich auf unterschiedliche Aspekte konzentrieren:

1. Adult Occupant Protection
2. Child Occupant Protection
3. Pedestrian Protection
4. Safety Assist

In den verschiedenen Tests werden je nach individueller Performance des Fahrzeugs Leistungspunkte vergeben, welche zusammengerechnet eine Gesamtpunktzahl ergeben und letztlich zu einer Sterne-Vergabe führen. Insgesamt können fünf Sterne für das Top-Rating durch das Fahrzeug erreicht werden. Außerdem muss eine bestimmte Mindestpunktzahl in den einzelnen Bereichen für die jeweilige Anzahl an Sternen erzielt werden, sogenannte Imbalance Grenzen. Diese dienen dazu, dass hohe Punktzahlen z.B. in den Tests zur Adult Occupant Protection andere weniger gute Ergebnisse in anderen Bereichen wie der Pedestrian Protection kompensieren. Damit möchte EuroNCAP erreichen, dass ein gleichmäßig verteiltes Sicherheitsniveau über alle Bereiche sichergestellt ist. Das führt beispielsweise dazu, dass ein Fahrzeug, welches in der Safety Assist Säule nur mit einer Punktzahl im Vier-Sterne-Bereich und in den anderen Säulen jeweils Fünf-Sterne Ergebnisse erzielt hat, dann "nur" eine vier Sterne Gesamtbewertung erhält. [ENCAP16]

Jahr	2013	2014	2015
Für fünf Sterne, wenigstens	80%	75%	75%
Für vier Sterne, wenigstens	70%	65%	65%
Für drei Sterne, wenigstens	60%	50%	50%
Für zwei Sterne, wenigstens	55%	40%	40%
Für einen Stern, wenigstens	45%	30%	30%

Tabelle 3.1: Relativer Anteil an der Gesamtpunktzahl für Sterne-Bewertung [ENCAP16]

Jahr	2013	2014	2015
Bereich 1: Adult Occupant Protection	50%	40%	40%
Bereich 2: Child Occupant Protection	20%	20%	20%
Bereich 3: Pedestrian Protection	20%	20%	20%
Bereich 4: Safety Assist	10%	10%	20%

Tabelle 3.2: Gewichtungsfaktoren für die einzelnen Bereiche [ENCAP16]

2014	Bereich 1: Adult Occupant	Bereich 2: Child Occupant	Bereich 3: Pedestrian	Bereich 4: Safety Assist
5 Sterne	80%	75%	60%	65%
4 Sterne	70%	60%	50%	55%
3 Sterne	50%	30%	40%	30%
2 Sterne	30%	25%	20%	20%
1 Stern	20%	15%	10%	10%

Tabelle 3.3: Imbalance Grenzen für das Jahr 2014 [ENCAP16]

2015	Bereich 1: Adult Occupant	Bereich 2: Child Occupant	Bereich 3: Pedestrian	Bereich 4: Safety Assist
5 Sterne	80%	75%	65%	70%
4 Sterne	70%	60%	50%	60%
3 Sterne	50%	30%	40%	40%
2 Sterne	30%	25%	20%	20%
1 Stern	20%	15%	10%	10%

Tabelle 3.4: Imbalance Grenzen für das Jahr 2015 [ENCAP16]

Im Zeitraum von 1997 bis 2009 wurden durch EuroNCAP eine ganze Reihe von Fahrzeugen speziell auf ihre passive Sicherheit getestet und bewertet, obwohl der Widerstand durch die Automobilindustrie hoch war. Mittlerweile sind solche Consumer Tests jedoch wichtige Entwicklungstreiber in der Fahrzeugsicherheit und eine Reihe von Organisationen haben sich auch auf anderen Kontinenten gegründet und führen ihrerseits aufwändige Crash Tests durch. Zu den wichtigsten Tests gehören:

- Frontal Impact
- Car to Car Side Impact
- Pole Side Impact
- ergänzend: Kinder Dummies in den Crash Szenarien
- Fußgängerschutz Tests durch Beschuss von Motorhaube und Frontschürze mit Kopf- und Beinattrappen

Die Punkteberechnung und Bewertungskriterien in diesen Crash Tests sind hochkomplex und eine detaillierte Darstellung in Zusammenhang mit dieser Arbeit ist nicht erforderlich, da sie sich speziell auf die Tests hinsichtlich Aktiver Sicherheit konzentriert, so das auf die jeweiligen Protokolle zu den Tests verwiesen wird. [ENCAP16] Eine erste Berücksichtigung dieser Sicherheitssysteme erfolgte mit der Verschärfung der Tests zum Jahr 2009. Neben dem Whiplash Test, bei dem ein Heckanprall durch ein Fahrzeug nachgebildet wird, um die Vermeidung von möglichen Schleudertrauma im Nackenbereich der Insassen zu bewerten, können die Fahrzeuge Punkte für den Verbau von ESC und ABS oder eines Seatbelt Reminders erhalten, sofern diese Systeme serienmäßig im Fahrzeug vorhanden waren und nicht per Aufpreis bezahlt werden mussten.

Seit 2014 sind die Imbalance Grenzen erneut angehoben worden, so dass allein in dem Bereich “Safety Assist” nun ein weiteres aktives Sicherheitssystem zur Serienausstattung des Fahrzeugs gehören muss, wie zum Beispiel ein Geschwindigkeitsbegrenzungsassistent, Spurverlassenswarner oder auch Notbrems- bzw. Kollisionswarnsysteme. Speziell bei letzteren erfolgt nicht mehr nur eine quantitative Bewertung, also ob das System “in-Serie” verbaut ist, sondern viel mehr eine qualitative Bewertung, bei der das System auf die Fähigkeit, aus unterschiedlichen Geschwindigkeiten Unfälle zu vermeiden bzw. seine Folgen zu mindern, überprüft wird. Die Motivation zu einer solchen Bewertung kommt aus der Analyse von Unfallstatistiken, die sich auch schon in der Vergangenheit als Treiber weiterer Sicherheitssysteme insbesondere im passiven Bereich erwiesen haben. Beispielsweise wurden Studien durchgeführt, welche den Effekt zur Einführung von ESC Systemen untersucht haben. [ENCAP16] Darin konnte festgestellt werden, dass die Zahl der tödlichen Unfälle etwa um 20% reduziert werden konnte.

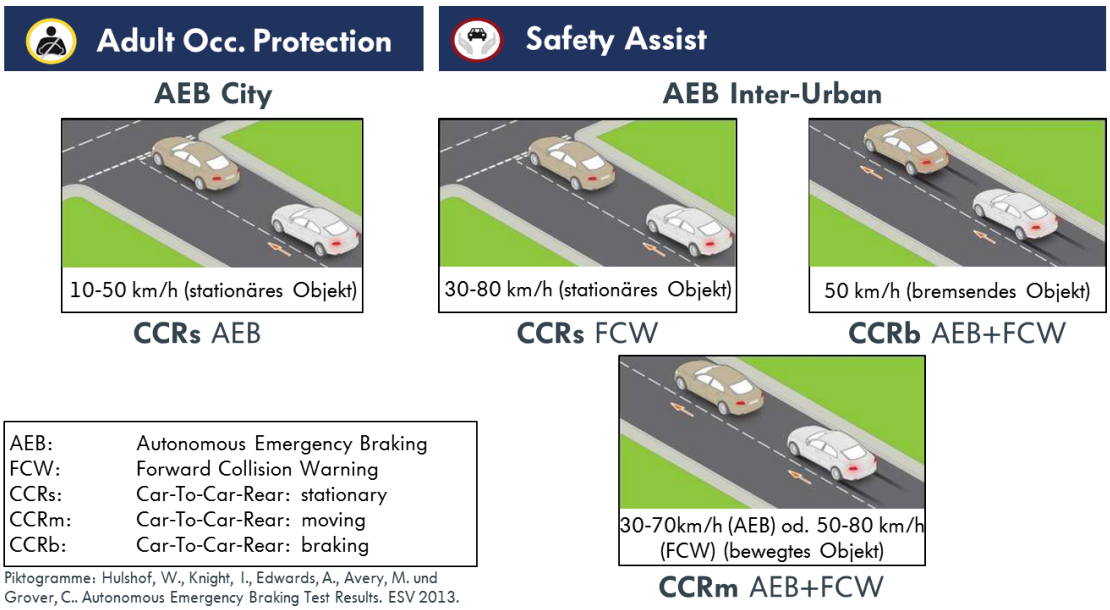


Abbildung 3.1: Notbremsszenarien für die Bewertung von AEB/FCW Systemen (vgl. [HKE+13])

Da die Bewertung der aktiven Sicherheitssysteme z. T. in unterschiedlichen Bereichen erfolgt und damit auch unterschiedliche zum Gesamtergebnis beitragen können, wird im folgenden die grundsätzliche Punktebewertung und insbesondere auch die Abhängigkeiten für eine Berücksichtigung dargestellt.

Bzgl. der Notbremssysteme gibt es mehrere Kategorien, in denen Punkte für die Gesamtbewertung erzielt werden können. Im konkreten Fall handelt es sich dabei um die beiden Kategorien

“AEB City” und “AEB Interurban”, wobei erstere zum Bewertungsbereich “Occupant Protection” und letztere zum Bereich “Safety Assist” zählt.

Für die Kategorie “AEB City” können maximal drei von 38 Punkten erreicht werden, die wiederum in ihrer Gesamtheit zu 40% in die Sterne-Bewertung einfließen können. Die Imbalance-Grenze für 5-Sterne liegt im übrigen bei 30,4 Punkten und entspricht somit 80%. Bevor eine Berücksichtigung überhaupt erfolgen kann, müssen ein paar Eingangsvoraussetzungen erfüllt sein: So muss der Whiplash-Test für die Vordersitze, also die Bewertung zur Schleudertrauma-Vermeidung, mit 1,5 Punkten bzw. “good” absolviert werden. Außerdem muss die Funktion nach jedem Zündwechsel (hier: Zündschlüssel in Nullstellung und erneutes Motor starten) automatisch eingeschaltet sein, auch wenn sie in der Fahrt davor deaktiviert wurde. Weiterhin müssen die Testfälle für “AEB City” kleiner gleich 20 km/h vollständig vermieden werden, andernfalls wird der Test mit 0 Punkten bewertet.

In der Kategorie “AEB Interurban” können maximal drei weitere Punkte im Bereich “Safety Assist” erzielt werden. Hier ist der Anteil an der Gesamtpunktzahl mit 13 Punkten deutlich höher, wobei die Imbalance-Grenze für das Jahr 2014 bei 8,5 Punkten bzw. 65% und für 2015 bei 9,1 Punkten respektive 70% liegt. Insgesamt beträgt die Gewichtung dieses Bereiches 20% an der Sterne-Bewertung. Wenn man sich nun die anderen Kategorien in diesem Bereich anschaut, dann stellt man fest, dass ein zusätzliches aktives Sicherheitssystem vorhanden sein muss, sofern ein OEM 5-Sterne für sein Fahrzeug erhalten möchte. Denn allein durch Seatbelt Reminder und ESC können nur 6 von 13 Punkten erreicht werden, so dass die Imbalance-Grenze für 5-Sterne mit 8,5 Punkten nicht erreicht werden kann. Für LDW kann man zusätzliche 1 Punkt erhalten, der ebenfalls nicht ausreichend ist. Daher ist entweder eine hohe Bewertung bei “AEB Interurban” bzw. auch bei den “Speed Assisting Systems” erforderlich. Da jedoch auch innerhalb der Kategorien weitere, z. T. hohe Anforderungen an die jeweiligen Systeme gestellt werden, ist eine genaue Prüfung und Vorhersage über die Leistungsfähigkeit der einzelnen Systeme Pflicht, damit am Ende eines Fahrzeugprojektes das Sterne-Ziel erreicht wird.

3.2.2 Das EuroNCAP AEB Test Protokoll

Die Konzentration im Folgenden wird nun auf die Kategorien von “AEB City und Interurban” gelegt, welche in [ENCAP13] beschrieben sind, da sie einen wesentlichen Testfaktor für die Volkswagen AG und seine Sicherheitsansprüche vor Kunde darstellen. Innerhalb dieser Kategorien werden ebenfalls wieder Punkte für die jeweilige Performance des Systems bzw. der Funktion im jeweiligen Testfall vergeben. Es wird dabei Wert auf einen älteren Stand des Testprotokolls gelegt, da das spätere Prinzip einer Methodik auch auf Änderungen im Testprotokoll

anwendbar sein wird. Jedoch erlaubt die Berücksichtigung einer älteren Version des Protokolls einen gewissen Know-How-Schutz, weil man sich dann auf das Prinzip konzentriert, die Detaillösungen jedoch nicht offen gelegt werden.

Die folgenden Szenario-Typen können dabei unterschieden werden:

- Car-to-Car Rear: stationary (CCRs)
- Car-to-Car Rear: moving (CCRm)
- Car-to-Car Rear: braking (CCRb)

Das Szenario CCRs ist insbesondere dadurch gekennzeichnet, dass sich das Fahrzeug, welches mit dem System ausgestattet ist und nachfolgend Vehicle-Under-Test (VUT) genannt wird, einem stationären Objekt nähert, im Folgenden EuroNCAP-Vehicle-Target (EVT) bezeichnet. In allen Szenarien existiert ein "idealer Pfad", der eine Gerade zwischen der Mitte des vorderen Stoßfängers vom VUT und der Mitte des hinteren Stoßfängers des EVTs entspricht und orthogonal zu beiden Fahrzeugen verläuft, so dass die jeweiligen Mitten in einer Flucht stehen. Die Geschwindigkeit des VUT wird dabei sukzessive erhöht, zunächst nur in $10\ km/h$ Schritten, sobald jedoch ein Anprall erfolgt ist, in $5\ km/h$ Schritten. Zudem wird nach dem ersten Testfall mit Kontakt zum EVT die Testgeschwindigkeit erst um $5\ km/h$ reduziert, um zu prüfen, wie das System bei dieser Geschwindigkeit abschneidet. Diese Regelung "erst-10-dann-5" hat das Ziel, den Testaufwand zu reduzieren, wobei damit erst in späteren Jahren zu rechnen ist und die Leistungsfähigkeit des Systems weiter zunimmt. Zudem ist das Geschwindigkeitsband, über das die einzelnen Testfälle erfolgen, abhängig davon, in welcher Kombination die Funktionen AEB und FCW vorliegen. Beispielsweise muss ein System, welches nur AEB und nicht FCW unterstützt, auch im Rahmen von "AEB Interurban" seine Performance im Band von $30 - 80\ km/h$ zeigen, obwohl dafür eigentlich nur die Funktion FCW vorgesehen ist. Erneut muss bereits hier wieder eine strategische Entscheidung aus Sicht des OEMs getroffen werden, welche Funktion ins Fahrzeug integriert werden soll.

Das CCRm Szenario unterscheidet sich dahingehend, dass das EVT nun eine eigene Geschwindigkeit im Testfall hat, welche bei $20\ km/h$ liegt. Das Geschwindigkeitsband für das Szenario, welches ebenfalls sukzessive nach der gleichen Methode wie oben beschrieben gesteigert wird, reicht von $30\ bis\ 80\ km/h$. Hier ist strategisch entscheidend, ob sich AEB/FCW einzeln oder in Kombination im Fahrzeug angeboten wird. Ist beispielsweise nur AEB und nicht FCW verfügbar, so kommen die Testfälle $75\ km/h$ und $80\ km/h$ potentiell hinzu, die bei einem kombinierten System nur für FCW vorgesehen sind.

Im CCRb Szenario wird die Geschwindigkeit des EVTs auf $50\ km/h$ angehoben. Das VUT

Abbildung 3.2: Das Car-to-Car Rear: stationary Szenario(vgl. [ENCAP13])

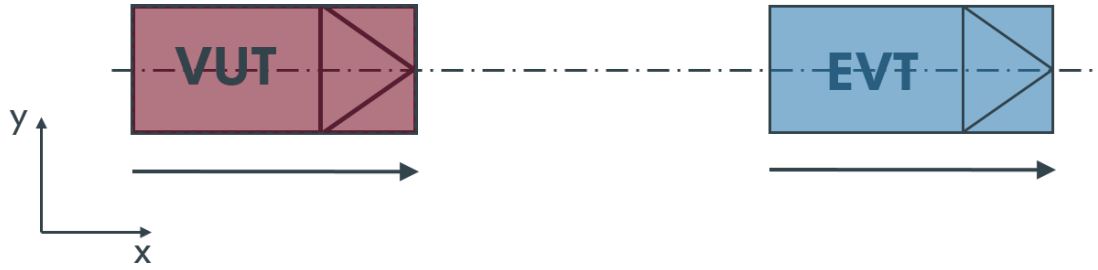


Abbildung 3.3: Das Car-to-Car Rear: moving Szenario (CCRm)(vgl. [ENCAP13])

fährt ebenfalls mit gleicher Geschwindigkeit hinter dem EVT in einem bestimmten Abstand her. Das Szenario wird einmal mit einem Abstand von 12 m und 40 m durchgeführt. Zudem beginnt das EVT zu einem bestimmten Zeitpunkt mit einer Verzögerung von 2 m/s^2 oder 6 m/s^2 die Geschwindigkeit zu reduzieren, so dass sich in Kombination mit dem Abstand jeweils vier Testfälle in Summe für dieses Szenario ergeben.

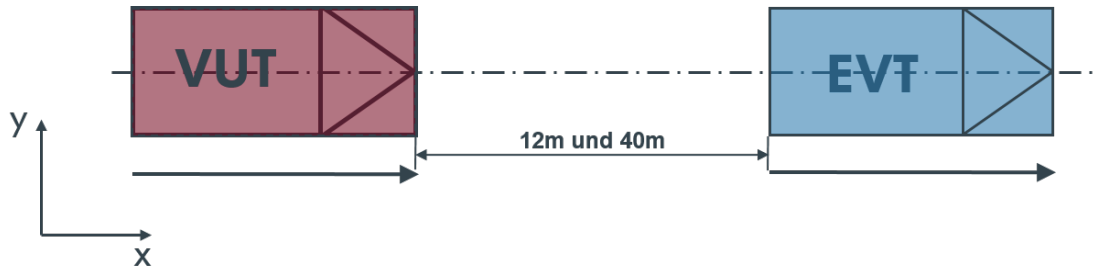


Abbildung 3.4: Das Car-to-Car Rear: braking Szenario (CCRb)(vgl. [ENCAP13])

Insgesamt werden folgende nominellen Testgeschwindigkeiten in den Kategorien “AEB City” und “AEB Interurban” getestet. Als grundlegende Funktionalität des Systems wird angenommen, dass es sich um ein kombiniertes System mit AEB und FCW handelt.

V_{ego} [km/h]	AEB City „CCRs“ Stationary	AEB Interurban „CCRs“ Stationary	AEB Interurban „CCRm“ Moving
10	X		
15	X		
20	X		
25	X		
30	AEB – X	X	X
35	X	X	X
40	X	X	X
45	X	X	X
50	X	X	AEB – X
55		X – FCW	X
60		X	X
65		X	X – FCW
70		X	X
75		X	X
80		X	X

V_{ego} [km/h]	AEB Interurban „CCRb“ Braking
50 (40m / 2m/s²)	X
50 (40m / 6m/s²)	X – AEB & FCW
50 (12m / 2m/s²)	X
50 (12m / 6m/s²)	X

AEB: Autonomous emergency braking
CCR: Car to car rear
FCW: Forward collision warning

Abbildung 3.5: Übersicht aller Testfälle

3.2.3 Toleranzen in den EuroNCAP Testfällen

Um die Vergleichbarkeit unterschiedlicher Systeme über verschiedene Fahrzeughersteller zu gewährleisten, müssen in den einzelnen Szenarien gewisse Toleranzen bei verschiedenen Parametern eingehalten werden. Andernfalls ist der Testfall ungültig und wird nicht gewertet. Der Zeitpunkt zur Berücksichtigung der erlaubten Toleranzen beginnt bei T_0, welcher geschwindigkeitsabhängig einer TTC von $4\ s$ entspricht. Das Ende des Testfalls liegt bei T_{end}, wobei mehrere Bedingungen zu einem Testende führen können: Zum einen kann die Geschwindigkeit des VUTs vollständig abgebaut werden bzw. kleiner als beim EVT sein, so dass kein Anprall erfolgt. Zum anderen kann es jedoch zu einem Anprall kommen, welcher auch das Testende bedeutet; entscheidend hierbei ist jedoch dann die Restgeschwindigkeit v_{res}, die dann in die Punkteberechnung einfließt. Ein weiterer wichtiger Zeitpunkt während des Testfalls bildet T_{AEB}, welcher den Auslösezeitpunkt des aktiven Sicherheitssystems bezeichnet. Die jeweiligen Toleranzen müssen dann innerhalb von T_0 und T_{AEB} bzw. bis T_{end} eingehalten werden, sofern kein Auslösung erfolgt ist. Da der letzte Fall allerdings der denkbar schlechteste Fall während der Bewertung des VUT ist, bleibt dieser nur theoretischer Natur. [ENCAP13]

Aufgrund der Charakteristik der jeweiligen Szenarien dürfen folgende Parameter in bestimmten Wertebereichen variieren:

CCRs:

- Die nominale Geschwindigkeit des VUTs um $+ 1,0\ km/h$,
- die seitliche Abweichung des VUT vom idealen Pfad um $0\ \pm\ 0,1\ m$,
- die Gier-Rate des VUT um $0\ \pm\ 1,0\ °/s$,
- die Lenkwinkelrate für das VUT um $0\ \pm\ 15,0\ °/s$.

CCRm:

- Die Bedingungen für das VUT aus *CCRs* gelten analog,
- zudem die nominale Geschwindigkeit des EVTs um $\pm\ 1,0\ km/h$
- die seitliche Abweichung des EVTs vom idealen Pfad um $0\ \pm\ 0,1\ m$,
- die Gierrate des EVTs um $0\ \pm\ 1,0\ °/s$,
- die Lenkwinkelrate für das EVT um $0\ \pm\ 15,0\ °/s$.

CCRb:

- Die Bedingungen für das VUT aus *CCRs* gelten analog,
- ebenso die Bedingungen für das EVT aus *CCRm*,
- der Abstand zwischen beiden Fahrzeugen um $\pm\ 0,5\ m$

Ein Testfall ist somit dann gültig, wenn die Toleranzen zwischen T_0 und T_{AEB} innerhalb der oben beschriebenen Wertebereiche liegen.

Da die erzielte Punktzahl davon abhängig ist, mit welcher Restgeschwindigkeit zum Zeitpunkt T_{end} vorliegt, ergibt sich folgende Formel zur Berechnung der Punkte p für jeden Testfall:

$$p_{tc}\ =\ [\frac{v_{nominal}\ -\ v_{rest}}{v_{nominal}}]\ *\ p_{test} \tag{3.1}$$

Für die unterschiedlichen Kategorien “AEB City” bzw. “AEB Interurban” werden in jedem nominalen Testfall z.T. unterschiedlich viele Punkte vergeben. [SWR13] Dies gilt sowohl für die Bewertung von AEB Systemen als auch für FCW Systeme bzw. bei Kombination beider Systeme. [ENCAP13]

Testfall	AEB City, AEB		AEB Interurban, AEB		AEB Interurban, FCW		
	Anforderung	CCRs	CCRm	CCRb	CCRs	CCRm	CCRb
10.0	true	1	-	-	-	-	-
15.0	true	2	-	-	-	-	-
20.0	true	2	-	-	-	-	-
25.0		2	-	-	-	-	-
30.0		2	1	-	2	-	-
35.0		2	1	-	2	-	-
40.0		1	1	-	2	-	-
45.0		1	1	-	2	-	-
50.0		1	1	-	3	1	-
55.0		-	1	-	2	1	-
60.0		-	1	-	1	1	-
65.0		-	2	-	1	2	-
70.0		-	2	-	1	2	-
75.0		-	-	-	1	2	-
80.0		-	-	-	1	2	-
50.0	$2\ m/s^2,\ 12\ m$	-	-	1	-	-	1
50.0	$2\ m/s^2,\ 40\ m$	-	-	1	-	-	1
50.0	$6\ m/s^2,\ 12\ m$	-	-	1	-	-	1
50.0	$6\ m/s^2,\ 40\ m$	-	-	1	-	-	1
$\sum\ p_{tc}$		14	11	4	18	11	4
$\sum\ scaled$		2,5	1,5		1,0		

Abbildung 3.6: Punktebewertung bei EuroNCAP

Anschließend wird die absolute Punktzahl in den prozentualen Anteil an der Gesamtpunktzahl umgerechnet und mit dem Faktor für die jeweilige Kategorie versehen. Für “AEB City” beträgt dieser Faktor $2,5$; für “AEB Interurban” hingegen wird der Faktor $1,5$ hinsichtlich AEB respektive der Faktor $1,0$ hinsichtlich FCW unterschieden.

$$p_{stars} = \frac{p_{tc}}{p_{max,tc}} * factor_{x,y}, x \in [City; Interurban]; y \in [AEB, FCW] \tag{3.2}$$

In Ergänzung kommen noch jeweils $0,5$ Punkte hinzu, welche bestimmten funktionalen Eigenschaften von HMI, Gurtstraffern, usw. Rechnung tragen. Sie sind jedoch für diese Arbeit nicht weiter von Bedeutung, da diese Funktionen Teil von anderen Subsystemen sind.

Zur Einhaltung der Toleranzen werden Fahr- bzw. Bremsroboter überwiegend von ABD (vgl. [Dyn19a]) verwendet, welche die Steuerung des Fahrzeugs im Test mit recht hoher Genauigkeit übernehmen. Die Firma wirbt damit, dass alle 7 offiziellen EuroNCAP Testcenter ABD Fahrroboter im Einsatz haben.

3.2.4 Sonstige Consumer-Test-Organisationen

Oben wurde bereits eine Übersicht über die verschiedenen Consumer-Test-Organisationen gegeben. Der US NCAP von der National Highway Traffic Safety Administration, das Pendant zum EuroNCAP führt in einer ähnlichen Art und Weise durchgeführt, was die Szenarien betrifft. Allerdings sind die Parameter und zugehörige Toleranzen in ihrem Wertebereich weiter gefasst, welches auf die manuelle Fahrzeugführung während der Tests zurückzuführen ist. Insgesamt unterscheiden sich die EuroNCAP Szenarien CCRs, CCRm und CCRb von ihren US NCAP Äquivalenten (vgl. die Abbildungen 3.2, 3.3 und 3.4 in folgenden Aspekten):

- Es gibt weniger konkrete Testfälle in den einzelnen Szenarien. Insgesamt reduziert sich die Anzahl durch die Auswahl von zwei Geschwindigkeiten für das VUT auf sechs konkrete Testfälle.
- Beim CCRs Äquivalent mit einem stehenden Hindernis fährt das VUT (=SV) mit einer Geschwindigkeit von $25\ mph\ (40,2\ km/h)$.
- Im CCRm Gegenstück fährt das VUT zusätzlich zu den $25\ mph\ (40,2\ km/h)$ auch $45\ mph\ (72,4\ km/h)$ und das EVT (=POV) mit jeweils $10\ mph\ (16,1\ km/h)$ bzw. $20\ mph\ (32,2\ km/h)$ zu den jeweiligen Geschwindigkeiten des VUT aus dem CCRs Pendant (niedrig-niedrig) und (hoch-hoch).
- Beim CCRb Szenario für den US NCAP fahren beide Fahrzeuge die gleiche, jeweilige Geschwindigkeit von $25\ mph\ (40,2\ km/h)$ bzw. $35\ mph\ (56,3\ km/h)$, während die Verzögerungsrate bei $3\ m/s^2$ liegt. Der Abstand zwischen den beiden Fahrzeugen darf für die Nominalgeschwindigkeit $328\ ft\ (100\ m)$ bzw. $45,3\ ft\ (13,8\ m)$ betragen. Speziell im $25\ mph$ Fall führt dies dazu, dass das EVT zum Stehen kommt, bevor das VUT das Target erreicht.
- die maximale Gierrate ϕ' darf $2\ °/s$ nicht überschreiten.

Eine Reihe von Parametern dürfen hierzu ebenfalls in bestimmten Grenzen variieren, wobei dieses Toleranzband etwas weiter gefasst ist als beim europäischen NCAP. Dazu zählen:

- Im Gegensatz zu EuroNCAP dürfen beide Fahrzeuge hier um bis zu $\pm\ 1,0\ mph\ (1,6\ km/h)$ von der Nominalgeschwindigkeit abweichen.
- Die seitliche Abweichung vom idealen Pfad darf maximal $0,6\ m$ für jeweils beide Fahrzeuge betragen.

- Zusätzlich dürfen aber auch beide Fahrzeuge in sich nur $0,6\ m$ seitlich voneinander abweichen, welches eine zusätzliche Anforderung darstellt.
- Der Abstand darf max. um $\pm\ 8\ ft\ (2,4\ m)$ variieren.

Die Einhaltung der Testparameter beginnt mit einer TTC von $5,1\ s$ oder umgerechnet in einer Entfernung von $187\ ft\ (57\ m)$ zum EVT für die CCRs Äquivalent. Für CCRs gilt eine TTC von $5,0\ s$ bzw. mit Bezug auf die jeweilige Geschwindigkeit in einer Entfernung von $110\ ft\ (34\ m)$ respektive $183\ ft\ (56\ m)$; beim bremsenden EVT beginnt diese bei $3\ s$, nachdem das EVT mit der Bremsung begonnen hat. Die Schwelle für das Abbremsen ist dabei mit $0,5\ m/s^2$ spezifiziert. Die Toleranzbedingungen gelten dann jeweils bis zu dem Zeitpunkt, wo ein Kontakt zwischen beiden Fahrzeugen entsteht oder $1\ s$ nachdem das VUT die Geschwindigkeit des EVTs erreicht bzw. unterschritten hat.

Die Bewertung ist allerdings anders als bei EuroNCAP organisiert: Hier werden in der Regel von jedem Testszenario acht valide Testläufe eingefahren, von denen in sieben Testfällen das VUT jeweils eine bestimmte Geschwindigkeitsreduktion erreichen muss. In Abhängigkeit der jeweiligen Testgeschwindigkeit entspricht die Reduktion einer TTC von $0,6\ s$, zu der die Auslösung der Notbremsung eingeleitet sein muss. [NHTSA14]

Pre-Crash Scenario	SV Speed Reduction or SV-to-POV Crash Avoidance					
	SV: 25 mph POV: 0 mph	SV: 45 mph POV: 0 mph	SV: 25 mph POV: 10 mph	SV: 45 mph POV: 20 mph	SV: 35 mph POV: 35 mph	SV: 25 mph POV: 25 mph
Stopped POV	$\geq$ 9,8 mph (15,8 km/h) for a least 7 or 8 valid test trials	-	-	-	-	-
Slower POV	-	-	No SV-to-POV impact for at least 7 or 8 valid test trials	$\geq$ 9,8 mph (15,8 km/h) for a least 7 or 8 valid test trials	-	-
Decelerating POV	-	-	-	-	$\geq$ 10,5 mph (16,9 km/h) for a least 7 or 8 valid test trials	$\geq$ 9,8 mph (15,8 km/h) for a least 7 or 8 valid test trials

Abbildung 3.7: CIB Performance Requirements [NHTSA14]

Pre-Crash Scenario	SV Deceleration					
	SV: 25 mph POV: 0 mph	SV: 45 mph POV: 0 mph	SV: 25 mph POV: 10 mph	SV: 45 mph POV: 20 mph	SV: 35 mph POV: 35 mph	SV: 25 mph POV: 25 mph
False positive Check (STP) in lieu of POV	$\leq$ 0,25g for 7 or 8 valid test trials	$\leq$ 0,25g for 7 or 8 valid test trials	-	-	-	-

Abbildung 3.8: CIB Non-Activiation Requirements [NHTSA14]

3.3 Die Bedeutung von Consumer Tests für die Volkswagen AG

Für die Volkswagen AG wie für jeden anderen Automobilhersteller gilt die Maximierung des Sicherheitsniveaus eines Fahrzeugs sowie des Kundennutzens bei gleichzeitiger Optimierung des finanziellen Aufwandes. Dieses Spannungsfeld beeinflusst maßgeblich die Produktplatzierung und damit auch den Vertriebserfolg eines Fahrzeugs. Somit sind die Bewertungen durch Consumer Test Organisationen weltweit ein wichtiges Kriterium zur Positionierung der einzelnen Fahrzeuge im Markt. Als führender Automobilhersteller auch im Hinblick auf die Jahresproduktion von neuen Fahrzeugen spielt der Aspekt Fahrzeugsicherheit eine gewichtige Rolle, so dass hier das bestmögliche Ergebnis im bereits angedeuteten Spannungsfeld zwischen Sicherheit, Kundenzufriedenheit und Kosten erreicht werden soll. Das jeweilige Sicherheitsniveau entspricht daher dem Gleichgewicht zwischen technischer Machbarkeit, der individuellen Anforderung des Kunden als maßgeblichen Treiber für die Produktentwicklung und der finanziellen Ausgewogenheit sowohl für den Kunden als auch für das Unternehmen, aggregiert zu einem Automobil mit heutigen Technologien.

Selbst rechtliche Vorgaben und Regelungen konkretisiert in Gesetzesanforderungen spiegeln mittelbar den Kundenwunsch wider, da eine Gesellschaft als kollektiv über seine Vertreter in Parlament und Ausschüssen die Rahmenbedingungen für die Automobilwirtschaft gestaltet.

Eine Bestbewertung kann dabei je nach Produkt und Markt das angestrebte Ziel für einen Automobilhersteller darstellen, wie die Historie der Fahrzeugresultate gezeigt hat. [ENCAP16] Durch die Verschärfung des Bewertungsprozesses bleibt zunächst jedoch noch offen, welche Auswirkungen diese Veränderung auf das Spannungsfeld haben wird.

Die Projektbedingungen

Da speziell die Consumer-Test-Organisationen, die mittelbar im Auftrag des Kunden die Bewertung über die sicherheitstechnische Leistungsfähigkeit der verschiedenen Fahrzeuge über alle Automobilhersteller hinweg durchführt, einen bedeutenden Anteil bei der Entwicklung von Fahrzeugen einnehmen, wird dieser Bedingung auch im Entwicklungsprozess Rechnung getragen. Hierzu werden aufbauend auf den oben genannten Bestimmungen und Test Protokollen die Anforderungen an die Ausstattung und die Sicherheitssysteme für die einzelnen Märkte definiert. Da jeder Markt unterschiedlichsten Randbedingungen unterliegen kann, sind diese z. T. deutlich voneinander abweichend; prinzipiell gilt jedoch, dass das Sicherheitsniveau eines Fahrzeugs mit dem technologischen und wirtschaftlichen Entwicklungsgrad ansteigt.

Aufbauend auf dem Entwicklungsprozess aus [GSSZ13] wird ein Lastenheft als Ergänzung zum Funktionslastenheft erstellt. Dieses stellt einerseits die genauen Anforderungen für den Zulieferer bereit, andererseits bietet es auch weitere Vorteile mit Blick auf die interne Organisation beim Auftraggeber. Denn die Erstellung des Funktionslastenheftes aus reiner Feldperformance Sicht und die Ergänzung durch ein spezielles Consumer-Test Lastenheft erlaubt auch die Trennung der Verantwortlichkeiten innerhalb des Auftraggebers. Dadurch, dass beide Seiten für die Erfüllung der jeweiligen Anforderungen eintreten, kann ein bestmögliches Ergebnis im Sinne des Kunden erfolgen, ohne bewusste oder unbewusste Schwerpunktverschiebungen in der Performance hinterher zum Ende der Entwicklung einseitig zuzulassen.

Durch die detaillierte Darstellung der Punkte zu den einzelnen Testfällen und wie diese von der Restgeschwindigkeit abhängen, können prinzipiell zu jedem Testfall auch eine Anforderung definiert werden, wieviel Geschwindigkeit pro Testfall abgebaut werden muss, damit in der Gesamtbetrachtung die erforderliche Punktzahl für das Sterne-Ziel erreicht wird. In Verbindung mit einem Integrationsstufenmanagement, also der schrittweisen Integration von Funktionen und Systemen, kann dann zu jeder Stufe ein eigenes Ziel für das aktive Sicherheitssystem konkretisiert werden, um die Erfüllung der Anforderungen am Ende des Entwicklungsprozesses sicherzustellen.

Die Prüfung, ob die Anforderungen erfüllt werden bzw. erfolgt zur Zeit durch reale Versuchsfahrten auf dem Prüfgelände. Sofern in einem Testfall nicht das geforderte Ergebnis erreicht wird, werden die Einschätzungen des verantwortlichen Ingenieurs häufig mit weiteren Experten diskutiert und Maßnahmen abgeleitet, wie eine Verbesserung erreicht werden kann. Dabei bleibt die Sicht des Funktionsverantwortlichen “Consumer Test” auf das System von außen begrenzt, so dass sich in der Regel nur ein Teil der gesamten Signale auf dem CAN-Bus auswerten lassen. Es liegt somit eher eine Black-Box bzw. in Teilen eine Grey-Box im Entwicklungsprozess vor, die insbesondere auch durch die Zulieferung durch den Auftragnehmer von entsprechenden Systemkomponenten und eingebetteter Software manifestiert wird.

In den jeweiligen Protokollen zum Bewertungsprozess werden detaillierte Angaben auch zum Messequipment gemacht und welche Anforderungen indirekt auch die Fahr- und Bremsroboter hinsichtlich ihrer Genauigkeit in der Fahrzeugführung aufweisen müssen. Auch das EVT, ein aus Schlauchboot-Gewebe entwickelte, aufblasbare Attrappe eines Fahrzeugs ist im EuroNCAP Protokoll genau spezifiziert. Beim US NCAP Protokoll wird wie bereits vorher beschrieben, nur das EVT bzw. POV automatisiert gesteuert.

Allerdings ist der Aufwand zur Einrüstung nicht unerheblich. Zum Teil werden für die Installation der Messtechnik und des Versuchsaufbaus mitunter zwei Personen benötigt, die mind. einen

Abbildung 3.9: Fahr- und Bremsroboter von AB Dynamics. Sie sind speziell für die Wiederholbarkeit von genauen Parametervorgaben entwickelt worden [Dyn19a, Dyn19b]

Arbeitstag dafür benötigen; das Gleiche gilt im Übrigen auch für die Rückrüstung des Equipments.

Die beiden Konzerntools Virtual Test Drive [Aud14b] und Automotive Data and Time-triggered Framework [Aud14a] waren für das Projekt gesetzt. Bei ersterem handelt es sich um ein Tool zur Modellierung des Systemkontextes und bei ADTF um ein Framework, welche softwareseitige Komponenten im Fahrzeug bzw. im Aktiven Sicherheitssystem modelliert. Über eine Schnittstelle sind beide Tools miteinander verbunden, um den entsprechenden Datenaustausch zu gewährleisten. Eine detaillierte Darstellung erfolgt in Kapitel 7.

3.4 Zusammenfassung

Die Entwicklung und der Test werden immer komplexer und werden auch bei der Volkswagen AG weiter zunehmen. Allein die Anforderungen, die sich aus der Bewertung und der Einhaltung der Consumer-Testszenarien ergeben und welche Bedeutung sie für die Marktpositionierung vor

dem Kunden haben, führen zu einem stetig steigenden Aufwand. Die Consumer Tests helfen, eine Vergleichbarkeit über das Sicherheitsniveaus von modernen Fahrzeugen sicherzustellen und gleichzeitig eine Orientierung dem Kunden zu liefern, wie sicher sein Fahrzeug gegenüber anderen Wettbewerbern ist. Daher ist für die Volkswagen AG sowie auch für andere Hersteller eine exzellente Bewertung in der Consumer-Test-Bewertung essentiell, auch wenn sie auf den ersten Blick nur eine Randerscheinung für eine tatsächliche Kaufentscheidung zu sein vermag.

Um nun die Anforderungen darin zu erfüllen, gilt es neben einer grundsätzlichen Effizienzsteigerung der Entwicklung auch Toleranzbetrachtungen in exakt solchen Consumer-Test-relevanten Szenarien durchzuführen. Entsprechende Simulationstechniken gilt es hier einzusetzen, da eine abschließende Beurteilung von Toleranzbetrachtungen nur eingeschränkt im realen Betrieb möglich ist; auch sind solche Untersuchungen real nur eingeschränkt reproduzierbar zu gestalten.

Im nachfolgenden Kapitel 4 wird somit geprüft, welche Methodiken dabei bereits eingesetzt werden. Dazu wird ein Systematic Literature Review durchgeführt.

4 Moderne Ansätze zum Test und zur Absicherung von Aktiven Sicherheitssystemen in Consumer Tests - ein Systematic Literature Review

Aus dem vorherigen Kapitel wurde ersichtlich, dass diverse Ansätze entwickelt und weiter verfeinert wurden, um aus einer generellen Perspektive Fahrzeugfunktionen zu testen und abzusichern. Dabei wird angestrebt insbesondere simulationsbasierte bzw. virtuelle Methoden zur Anwendung zu bringen. Aufgrund diverser Herausforderungen, wie z.B. im Bereich der Modellbildung und bei der Schaffung einer adäquaten Infrastruktur für die Simulationsschritte Pre-Processing, Berechnung und Post-Processing, wird weiter stark Forschungsarbeit geleistet.

Das Thema dieser Arbeit bezieht sich hingegen auf den speziellen Bereich hinsichtlich des Tests und der Absicherung einer aktiven Sicherheitsfunktion, die unter besonderen Bedingungen und unter der Maßgabe einer möglichst hohen Wiederholbarkeit durchgeführt werden: Im Rahmen von real durchgeführten Consumer Tests. Auf die genauen Aspekte zur Entwicklung solcher Sicherheitsfunktionen zu diesen Tests und eine detaillierte Beschreibung der Consumer-Tests bei EuroNCAP wurde bereits in Kapitel 3 eingegangen.

Ein weiterer, wichtiger Aspekt dieser Arbeit ist zudem, dass diese Tests eine hohe Kundenrelevanz haben, da die erzielten Ergebnisse in diesen Tests zentral veröffentlicht werden und aufgrund der Reproduktionsfähigkeit einen Vergleich unterschiedlicher Automobilmarken und Fahrzeugtypen ermöglicht, so dass für das Prestige eines Fahrzeugs das bestmögliche Ergebnis erzielt werden sollte.

Um nun eine geeignete Methode zur simulativen bzw. virtuellen Unterstützung für die Funktionsentwicklung Aktiver Sicherheitssysteme in diesen Tests abzuleiten und umzusetzen, erfolgt eine Prüfung der darauf bezogenen Literatur über entsprechende, spezialisierte Ansätze, die über die allgemeinen Ansätze hinausgehen. Daher wird im folgenden ein Systematic Literature Review initial durchgeführt, welches wichtige Ansätze identifiziert und auch den Grad ihrer Inte-

gration in der Industrie herausstellt. Weiter wird kurz die Methodik und das zugehörige Protokoll zur Durchführung aufgeführt.

4.1 Methodik für ein Systematic Literature Review

Als Methodik zur Planung, Durchführung und Auswertung des Systematic Literature Review "Simulation von Consumer Tests" wird der technische Bericht EBSE-2007-01 - "Guidelines for performing Systematic Literature Reviews in Software Engineering" von Kitchenham et al. herangezogen. [KC07]

Es wird insofern nur von der Methodik aus Gründen der Lesbarkeit abgewichen, als dass andere Begriffe für die Prozessschritte dargestellt werden. Der Schwerpunkt der Erläuterung wird auf das entworfene Protokoll gelegt. Anschließend werden darauf aufbauend die konkreten Ergebnisse herausgestellt. Für eine weitere, detaillierte Darstellung der Methodik selbst sei auf den technischen Bericht verwiesen. [KC07, S. 4 und 44]

4.2 Hintergrund und Motivation

Es ist zuvor deutlich geworden, welchen Stellenwert Consumer Tests — und im speziellen EuroNCAP — für die Automobilbranche haben und wie wichtig ein gutes Resultat für ein Fahrzeug ist. Es konnte auch gezeigt werden, dass insbesondere die zulässigen Toleranzen in den entsprechenden Testszenarien einen nachhaltigen Einfluss auf die Restgeschwindigkeit und damit auf das Gesamtergebnis haben können. Durch den Aufbau eines Notbremssystems und durch die momentan noch zu berücksichtigenden funktionalen Unzulänglichkeiten der Sensorik ist eine breit angelegte Toleranzuntersuchung mit einer Vielzahl von Parametervariationen als Eingangsdaten für das System bzw. eine Systemkomponente durch reale Testfahrten nur begrenzt durchführbar. Daher werden simulative Ansätze in diesem Bereich als hilfreich angesehen.

Die Motivation besteht nun darin, entsprechende Konzepte und Ansätze zur Simulation von Consumer Tests insbesondere im industriellen Kontext zu identifizieren, die dort auch in hinreichendem Maße Anwendung finden. Ziel ist im Hinblick auf die Methodenentwicklung vorherige Ansätze und Konzepte einzubeziehen bzw. neue Herausforderungen der zugrundeliegenden Domäne zu berücksichtigen.

4.3 Protokoll der Studie

In diesem Abschnitt werden die relevanten Forschungsfragen zunächst formuliert, um eine hinreichende Aussagekraft der Literaturrecherche zu gewährleisten. Anschließend wird der logische Suchausdruck benannt, der für die ausgewählten Online-Bibliotheken und -Datenbanken verwendet wird. Die Auswahl an Selektionskriterien, die entscheiden, welche Literaturbeiträge für eine genauere, inhaltliche Analyse herangezogen werden, ist darauffolgend aufgeführt. Abschließend wird die Methodik beschrieben, anhand derer die Primärstudien zur Beantwortung der Forschungsfragen bewertet werden.

4.3.1 Forschungsfragen

Abgleitet aus der Motivation wurden folgende Forschungsfragen (*engl.: Research Questions (RQ))* formuliert:

- *[RQ-1]: Welche Ansätze und Konzepte zur Simulation von Consumer Tests, insbesondere bezogen auf EuroNCAP und NHTSA, existieren hinsichtlich Aktiver Sicherheit? (Konzeptsuche zur Simulation von Consumer Tests)*
- *[RQ-2]: Sofern [RQ-1] hinreichend durch entsprechende Beiträge beantwortet: Wie werden in diesen Ansätzen und Konzepten Aktive Sicherheitssysteme oder deren Komponenten bezüglich ihrer zulässigen Toleranzen simulativ bewertet? (Toleranzuntersuchung und ihre Bewertung)*
- *[RQ-3]: Welche Methodik wurde in diesen Ansätzen und Konzepten verwendet, um zielgerichtet eine solche Simulationsumgebung zu entwickeln, die das Ziel verfolgt, ein Aktives Sicherheitssystem in seinen zulässigen Toleranzen zu bewerten? (Methodische Entwicklung von Simulationsinfrastruktur)*

4.3.2 Suchausdruck für die Online-Bibliotheken und -Datenbanken

Der Suchausdruck ist von entscheidender Bedeutung für die spätere Qualität der Antworten zu den Forschungsfragen. Durch die Wahl der entsprechenden Schlüsselbegriffe können erste Eingrenzungen zu der Fülle an wissenschaftlichen Beiträgen erfolgen. Gleichzeitig müssen die Begriffe so gewählt werden, dass wichtige Beiträge nicht verloren gehen. Im Zuge der Recherchen zu den Publikationen [BHK+14, BBH+14a, BBH+15b] sind relevante Beiträge für einen Eigen-

kontrollmechanismus identifiziert worden, die in Folge der Datenbanksuche wieder auftauchen mussten. Dies konnte mit dem unten genannten Suchausdruck gewährleistet werden.

Folgender Suchausdruck wurde für die Studie verfasst:

> ((“virtual test” OR (“consumer tests” OR “consumer test” OR “EuroNCAP” OR “NHTSA” OR “USNCAP”)) AND simulation AND (ADAS OR “driver assistance system” OR “crash avoidance”) NOT medicine)

Die ersten Schlüsselbegriffe innerhalb der Klammern sorgen dafür, dass mindestens einer der genannten Begriffe in dem Artikel vorkommt; sie sind essentiell für die Identifikation der Ansätze und speziell bezogen auf den Projekthintergrund in Form der Consumer Tests. Eine gewisse Entkräftung dieser Ergebnisse ist hingegen durch den Begriff des “virtual testing” erfolgt, welcher durch die OR-Verknüpfung auch ohne einen der nachfolgenden Begriffe einbezogen wird. Dieser Ausdruck ermöglicht es, auch evtl. verwandte Beiträge, die nicht aus dem Consumer Test Umfeld kommen, einzubeziehen.

Mit Hilfe der AND-Verknüpfung in Verbindung mit “simulation” sollen zudem nur solche Studien ausgewählt werden, die potentiell simulative Aspekte enthalten, sich somit nicht ausschließlich auf reale Testfahrten beziehen, da solche nicht Gegenstand dieser Studie sind. Durch den nachfolgenden logischen Ausdruck wird berücksichtigt, das sowohl Fahrerassistenzsysteme als auch dessen Akronym sowie die Unfallvermeidung bei den Beiträgen enthalten sein können. Durch die Exklusion von medizinischen Beiträgen mit Hilfe des negierenden Ausdrucks “medicine” konnte sichergestellt werden, dass keine domänenfremden “consumer tests” enthalten sind. Hierdurch konnte die Zahl der nicht relevanten Beiträge deutlich reduziert werden.

4.3.3 Auswahl der Online-Bibliotheken und Datenbanken

Als Online-Bibliotheken und -Datenbanken werden wie folgt ausgewählt:

- ACM Online Library
- ELSEVIER Science Direct
- IEEE Xplore Online Library
- SAE Digital Library
- SpringerLink

Auf eine Einbeziehung von Google Scholar wurde an dieser Stelle verzichtet, da hierdurch eine mögliche Dopplung der Einträge erfolgt wäre. Auch wären evtl. Einträge gefunden worden, die ohne einen Herausgeber-Hintergrund oder ein Peer-Review veröffentlicht wurden und damit eine qualitative Einschätzung der Artikel erschwert ist.

4.3.4 Auswahlkriterien für die Studien und Beiträge

Im Folgenden sind nun mehrere Kriterien definiert worden, anhand derer die Beiträge und Studien in einem zweistufigen Prozess ausgewählt werden. Hierzu werden in einem ersten Schritt zunächst diejenigen Resultate ausgeschlossen, die definitiv nicht in die Thematik passen, jedoch durch den Suchstring ausgewählt wurden. In einem zweiten Schritt werden dann anhand weiterer Auswahlkriterien die konkreten Beiträge und Studien identifiziert, die eine Relevanz zur Beantwortung der Forschungsfragen aufweisen. Da sich diese Studie insbesondere auf Beiträge bezieht, die in englischer Sprache verfasst sind, werden die Kriterien ebenfalls in englischer Sprache aufgeführt, wo es als sinnvoll erscheint.

Ausschlusskriterien:

- keine eigenen Beiträge oder Studien, die bereits zur Thematik veröffentlicht wurden.
- keine doppelten Veröffentlichungen.
- keine Beiträge ohne Peer-Review.
- keine Lehrbuchsammlungen.
- keine Zweitliteratur, wie z.B. andere Literature Reviews
- keine Drittliteratur, wie Literature Reviews über andere Reviews.
- keine Umfragen unter Experten oder Probanden.
- keine Fahrerverhaltensstudien mit Realdaten oder Fahrsimulatoren, die speziell psychologische Aspekte bei Fahrern behandeln.
- keine Beiträge, die sich ausschließlich auf die reine Entwicklung von Fahrzeugfunktionen konzentrieren.
- keine Beiträge, die nicht explizit einen Teil über eine Simulationsarchitektur enthalten.
- keine Beiträge, die sich mit Vehicle-to-Infrastructure Kommunikation befassen.

- keine Beiträge, die sich mit V2V Kommunikation befassen und nicht explizit ein aktives Sicherheitssystem als Basis verwenden.
- Beiträge, die sich ausschließlich mit passiver Sicherheit beschäftigen, speziell mit den Themen Fahrzeugstruktur und Rückhaltesysteme.

Auswahlkriterien:

- Beiträge, die sich auf Consumer Tests in der Automobilindustrie beziehen.
- Beiträge, die sich auf Aktive Sicherheit beziehen.
- Beiträge mit Bezug zu AEB/FCW Systemen.
- Beiträge mit Bezug zu CIB/DBS Systemen (amerikanisch geprägte Begriffe, synonym für AEB/FCW).
- Beiträge, die sich auf Toleranzbetrachtungen bei Tests beziehen.
- Beiträge mit methodischer Beschreibung von Tests oder Simulationen.
- Beiträge, die experimentelle Studien enthalten.
- zugehörige Dissertationen.

Zur Begründung der Auswahl der Ausschlusskriterien: Das erste Kriterium schließt eigene Beträge und Studien aus, da diese sonst das Ergebnis verfälschen würden, denn die eigenen Beiträge beschäftigen sich direkt mit der Thematik, die durch die Forschungsfragen versucht wird zu klären. Doppelte Beiträge im Sinne einer tatsächlichen doppelten Veröffentlichung werden ebenfalls ausgeschlossen. Weiterhin wird Zweit- und Drittliteratur nicht betrachtet, da sie über spezielle Forschungsfragen und Vorgehensweisen z.B. eine abweichende Auswahl an Kriterien verfügen, die somit eine Vergleichbarkeit erschweren. Es wurde jedoch im Vorfeld geklärt, dass keine SLR existieren, die bereits die hier formulierten Forschungsfragen behandeln.

Neben diesen rein formalen Kriterien sind auch einige inhaltliche Ausschlusskriterien aufgeführt. Zunächst werden Umfragen unter Experten oder Probanden ausgeschlossen, da diese zwar einen Bedarf bzw. eine Problemstellung skizzieren können, jedoch offen bleibt, ob dabei unternehmensbezogene Interessen die jeweiligen Antworten beeinflusst haben. In diesem Zusammenhang sei vorab auch erwähnt, dass diese Art der Beiträge kaum als Kategorie in den Suchergebnissen aufgetaucht ist.

Weiterhin werden auch keine Fahrerstudien eingeschlossen. Zwar werden hierfür meistens Fahrsimulatoren eingesetzt, jedoch spielt der technische Aspekt in solchen Beiträgen eine unter-

geordnete Rolle; der Fokus liegt vielmehr auf den psychologischen Aspekten bei Fahrer oder Umfeld. Beiträge, die sich überwiegend mit der reinen Funktionsentwicklung z.B. eines neuen Steuerungsalgorithmus' befassen, werden einbezogen, da hier der Schwerpunkt auf der Funktion selbst liegt. Sollten jedoch Beiträge existieren, die keine Simulationsarchitektur enthalten, so sind diese auch von dieser Studie ausgenommen. Weiterhin werden solche Beiträge nicht eingeschlossen, die sich mit Fahrzeug-zu-Fahrzeug oder Fahrzeug-zu-Infrastruktur Kommunikation beschäftigen, da sie zwar als Teil der Aktiven Sicherheit kategorisiert werden können, hier jedoch die Simulation der Kommunikation und ihre Modellierung in Betracht gezogen wird. Dies liegt nicht im Fokus der Analyse.

Da virtuelle Tests auch im Rahmen von passiver Sicherheit denkbar sind, diese sich jedoch auf Struktur und Minderung von Unfallfolgen nach dem eigentlichen Anprall beziehen, werden solche Beiträge ebenfalls ausgenommen, da speziell die Phase vor dem Unfall untersucht wird. Als letztes Kriterium werden auch Lehrbuchsammlungen ausgeschlossen, da diese meist einen Überblick über bereits etablierte Verfahren und Ansätze wiedergeben und eher allgemeiner Natur sind. Sie werden ohnehin im Rahmen der Grundlagen vorausgesetzt.

Ein Papier wird ausgeschlossen, wenn **eines** der genannten Kriterien erfüllt ist.

Nach der Prüfung eines Ausschlusses werden folgende Kriterien für einen Einschluss der Papiere herangezogen. Auch hier gilt, dass eines oder mehrere Kriterien erfüllt sein müssen, um in die engere Auswahl zu kommen. Die ersten vier Kriterien stellen einen Bezug zum Thema dieser Arbeit, also zu Consumer Tests, aktiver Sicherheit oder den konkreten Systemen her. Weiterhin sind aber auch Beiträge interessant, die sich mit Toleranzbetrachtungen beschäftigen. Auch Beiträge, die sich methodisch mit Tests oder Simulation auseinandersetzen, können für die Entwicklung einer geeigneten Methode interessant sein und werden in die Auswahl übernommen. Experimentelle Studien sowie Dissertationen werden als letzte Kriterien herangezogen, da sie evtl. die vorliegenden Beiträge ergänzen.

Sollten am Ende noch Beiträge nicht anhand von Ein- und Ausschlusskriterien bewertet werden können, dann sind jene Beiträge in die Menge der bewerteten Beiträge zu überführen, da nicht ausgeschlossen werden kann, dass doch eine Relevanz vorliegt.

Nach Abschluss dieser Überprüfung wird sich eine Teilmenge an Fachartikeln ergeben, die dann einer weiteren, qualitativen Einordnung unterzogen wird.

4.3.5 Study Quality Assessment Kriterien

Der vorherige Abschnitt hat diejenigen Kriterien definiert, nach denen die Papiere für eine weitere Bearbeitung der Forschungsfragen ausgewählt und hinsichtlich ihrer Relevanz für die Beantwortung der Forschungsfragen untersucht werden sollen. Damit jedoch beurteilt werden kann, in welchem Maße die Beiträge zur Beantwortung der Fragen helfen, werden weitere Qualitätskriterien eingeführt. Für diese ergibt sich eine Gesamtpunktzahl, welche dann die Wertigkeit im Sinne der jeweiligen Forschungsfrage beschreibt.

RQ-1: Konzeptsuche zur Simulation von Consumer Tests

- Active safety: +0,2 Pkt.
- Consumer test(s): +0,2 Pkt.
- AEB/FCW/CIB/CBS: +0,2 Pkt.
- Industrial context or partners: +0,2 Pkt.
- Threats of validity or other validation process: +0,2 Pkt.

RQ-2: Toleranzuntersuchung und ihre Bewertung

- Punkte aus der Bewertung für *RQ-1* werden halbiert: max. 0,5 Pkt.
- Einzelne Komponenten eines Systems: +0,2 Pkt.
- Ganzes System: +0,2 Pkt.
- Modellierung der Toleranzen: +0,2 Pkt.
- Methodenbeschreibung: +0,2 Pkt.
- Automatisierte Parametervariation ohne Zufallsvariablen: +0,2 Pkt.

RQ-3: Methodische Entwicklung von Simulationsinfrastruktur

- Modellierung Aktives Sicherheitssystem: +0,2 Pkt.
- Szenariengenerierung: +0,2 Pkt.
- Auswertungsmethodik: +0,2 Pkt.
- Validierungsprozess: +0,2 Pkt.
- Dezidierte Beschreibung einer Meta-Methodik: +0,2 Pkt.

Für *RQ-1* wird ein Beitrag als wertvoll angesehen, wenn er sich auf Aktive Sicherheit, im weiteren auf Consumer Tests und auf Notbrems- oder Warnsysteme für Kollisionen bezieht (jeweils 0,2 Punkte). Da hier ein industrieller Kontext für die Praxistauglichkeit als positiv zu werten ist, werden hierfür ebenfalls 0,2 Punkte vergeben. Sollte zudem im Rahmen der Reflexion der eigenen Arbeit ein Abschnitt zur Validierung im Beitrag enthalten sein, werden weitere 0,2 Punkte hinzugerechnet. Speziell dieser Punkt bezieht sich auf eine kritische Auseinandersetzung mit den jeweils vorgestellten Arbeiten. Dies kann eine Validierung der Modelle sein, jedoch ist auch ein anderer Bezug zur kritischen Prüfung der Ergebnisse möglich, wie z. B. eine bewusste Einschränkung der vorgestellten Ergebnisse. Insgesamt können somit 1,0 Punkte erreicht werden.

Um für *RQ-2* zunächst hinreichend passende Beiträge zu erhalten, werden alle aus *RQ-1* oberhalb von 0,6 Punkten bewerteten Papiere herangezogen und ihre Punktzahl halbiert. Damit wird dem Umstand Rechnung getragen, dass die vorangegangene Bewertung den wichtigen Eingangsschritt in Richtung relevante Ansätze ermöglicht hat. Anschließend wird geprüft, ob die Papiere über einen Ansatz verfügen, der ein ganzes Aktives Sicherheitssystem oder nur eine einzelne Komponente modelliert. Bei positiver Bewertung gibt es jeweils 0,2 Punkte für diese Kriterien. Grundsätzlich sollte es positiver sein, dass ein Simulationsansatz sowohl einzelne Systeme als auch einzelne Komponenten zu modellieren erlaubt. Es können aber auch jeweils das eine oder andere Kriterium erfüllt sein. Als weiteres Kriterium wird die Modellierung von Toleranzen im Rahmen der Simulation bewertet. Wird z. B. die Ausgangslage des Szenarios variiert bzw. im weiteren Verlauf die Trajektorie eines modellierten Fahrzeugs, so wird dies auch mit 0,2 Punkten bewertet. Sollte der Beitrag zudem über eine konkrete Methodenbeschreibung verfügen, auf welche Art und Weise z. B. der Simulationsansatz verwendet werden kann und wie die Ergebnisse zu interpretieren sind, so werden weitere 0,2 Punkte vergeben. Am Ende wird als Kriterium herangezogen, ob die Parametervariation ohne eine Zufallsvariable in Abgrenzung zur Monte-Carlo Simulation auskommt, für welches weitere 0,2 Punkte vergeben werden.

Die Bearbeitung der Frage *RQ-3* soll durch fünf weitere Kriterien erfolgen. Um bewerten zu können, aus welchen Elementen die Methodik besteht, werden diese einzeln überprüft. Diese setzen sich zusammen aus einer Modellierung des aktiven Sicherheitssystems, welches den Grundstein legt; als weiteres Element kommt hier eine mögliche Szenariengenerierung in Betracht, die automatisiert eine Erstellung von Testfällen erlaubt. Außerdem erfolgt die Abfrage, ob im Beitrag auch eine Auswertungsmethodik aufgeführt wird, mit deren Hilfe die Testfälle zielgerichtet bewertet werden. Auch ein möglicher Validierungsprozess und die dazugehörigen Schritte werden hier positiv zu einer Gesamtbewertung hinzugezogen. In Abgrenzung zum Kriterium zu RQ-1 wird hier eine detaillierte Beschreibung einer Validierung vorausgesetzt und nicht nur eine kritische Auseinandersetzung erwähnt. Hierdurch kann zwar das Kriterium aus RQ-1 zutreffen,

jedoch nicht bei RQ-3 erfüllt sein; der umgekehrte Fall ist hingegen nicht möglich. Als letztes Kriterium wird berücksichtigt, ob bei der Entwicklung des Ansatzes eine Meta-Methodik verwendet wurde. Grundlage dafür bilden die identifizierten Beiträge aus *RQ-2*, da diese bereits als die relevanten Ansätze ausgewählt wurden.

4.4 Ergebnisse zum Systematic Literature Review "Simulation von Consumer Tests"

In diesem Abschnitt wird zunächst eine Übersicht aus der Eingabe des Suchstrings gegeben und gezeigt, wie sich die Anzahl der Beiträge nach den Ein- und Ausschlusskriterien reduziert hat. Danach werden die Ergebnisse aus den einzelnen Bewertungsgängen der jeweiligen Forschungsfragen dargestellt.

4.4.1 Ergebnisse nach Eingabe des Suchstrings

Die Anfrage mit dem oben definierten Suchstring ergab für die einzelnen Datenbank folgende Ergebnisse, die in den Grafiken 4.1 und 4.2 mit absoluten und relativen Werten dargestellt sind.

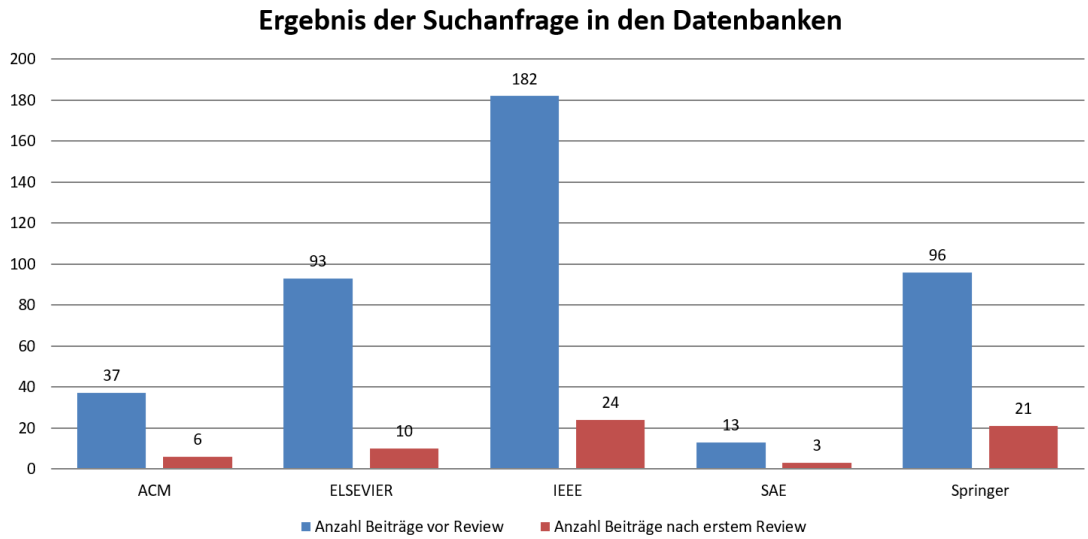


Abbildung 4.1: Ergebnis der Suchanfrage in den Datenbanken

Die Eingabe hat zu unterschiedlichen Trefferzahlen geführt. Während bei ACM 37 Treffer (8,8%) gefunden wurden, ergaben sich bei ELSEVIER (auch als sciencedirect bekannt) 93 Tref-

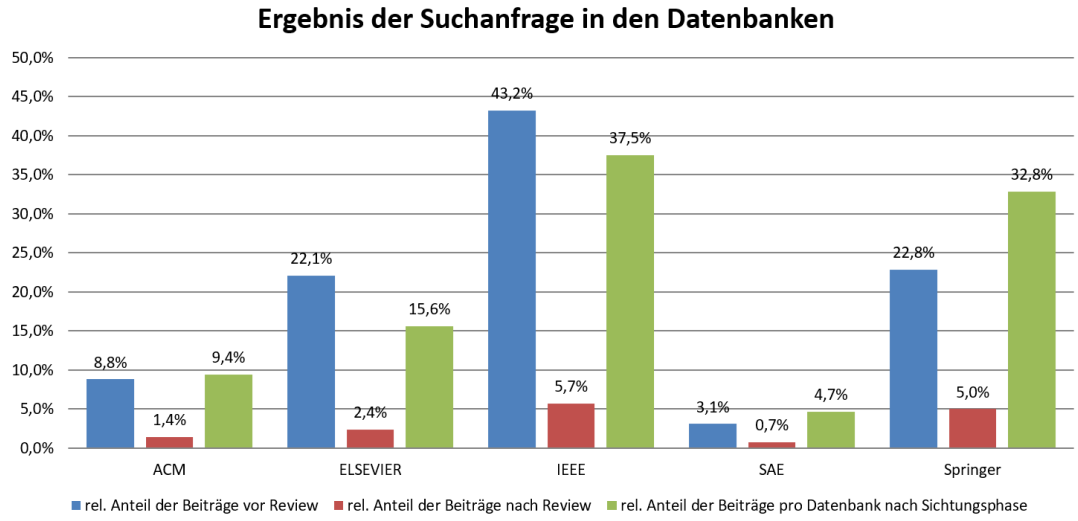


Abbildung 4.2: Relatives Ergebnis der Suchanfrage in den Datenbanken

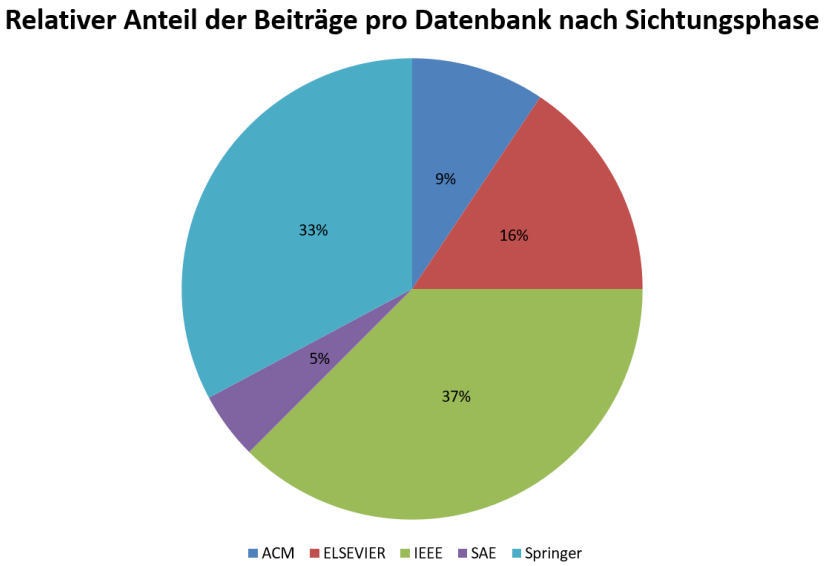


Abbildung 4.3: Relativer Anteil der Beiträge pro Datenbank nach Sichtungsphase

fer (22,1%). Bei IEEE führte der Suchstring zu einer Trefferzahl von 182 (43,2%) während bei SAE nur 13 Treffer (3,1%) gefunden wurden. Die Suche bei Springer erbrachte als erstes Zwischenergebnis 96 Beiträge (32,8%).

Diese Menge an Papieren und Beiträgen wurde dann anhand der oben genannten Aus- und Einschlusskriterien bewertet, um zunächst eine grundsätzliche Relevanz festzustellen. Dafür wurden die einzelnen Kriterien der Reihe nach im Rahmen einer Volltextsichtung bearbeitet. Sobald ein Kriterium als zutreffend für einen Beitrag gewertet werden kann, wird es ausgeschlossen. Danach wurde geprüft, ob der Beitrag eines der Einschlusskriterien erfüllt. Am Ende ergaben sich folgende übrig gebliebene Beiträge pro Datenbank, welches in den Abbildungen 4.1 und 4.2 zu entnehmen ist.

Danach verbleiben aus der Menge der ACM Papiere 6 Beiträge (9%), von ELSEVIER 10 Beiträge (16%), bei IEEE 24 Beiträge (37%), bzgl. SAE 3 Beiträge (5%) und bei Springer 21 Beiträge (33%) in der Auswahl zur Beantwortung der Forschungsfragen. Die Abbildung 4.3 zeigt die relative Verteilung dazu.

4.4.2 RQ-1: Konzeptsuche zur Simulation von Consumer Tests - Ergebnisse

Die folgende Tabelle 4.1 spiegelt die Bewertung der verbliebenen Beiträge aus der ersten Sichtungsphase wider. Spalte 1 bezieht sich auf die vergebene ID innerhalb der hier durchgeführten Studie; Spalte 2 repräsentiert den Index in dieser Arbeit mit Bezug auf das Literaturverzeichnis. In den Spalten 3 bis 7 folgen die Bewertungskriterien in englischer Sprache zur ersten Forschungsfrage. In der letzten Spalte wird die Summe der erzielten Punktzahlen dargestellt.

An dieser Stelle sei noch ein Hinweis für die Lesart der Punktezahl gegeben: Die jeweiligen Punkte beschreiben, welche Aspekte ein Papier oder Beitrag enthält. Dabei spielt es eine untergeordnete Rolle, ob beispielsweise ein Papier mit 0,4 Punkten aus Kriterium 1 und 3 höher als eines aus Kriterium 2 und 5 zu bewerten ist. Vielmehr zielt diese Bewertung darauf ab, dass ein Beitrag dann als hochwertig zur Klärung der Fragen anzusehen ist, wenn möglichst viele Kriterien erfüllt sind. Eine Bewertung im Verhältnis untereinander ist von nachrangiger Bedeutung.

Study ID	Reference ID	active safety	consumer test	AEB/FCW/ CIB/DBS	industrial context	threats to validity part	Summe für RQ-1
#0005	[Ber14]	0	0	0	0	0,2	0,2
#0006	[BBG+11]	0,2	0	0	0	0,2	0,4
#0009	[CC09]	0,2	0	0,2	0	0	0,4
#0017	[HH10]	0,2	0	0,2	0	0	0,4

#0029	[SC11]	0,2	0	0,2	0	0	0,4
#0037	[ZTWL13]	0,2	0	0	0,2	0	0,4
#0061	[GPSV09]	0,2	0	0,2	0,2	0,2	0,8
#0063	[HYRS09]	0,2	0	0	0	0	0,2
#0067	[HO00]	0,2	0	0	0,2	0	0,4
#0079	[KQW15]	0,2	0	0	0	0,2	0,4
#0083	[LCY+14]	0,2	0	0	0,2	0,2	0,6
#0084	[MKD08]	0	0	0	0	0	0
#0097	[RAY97]	0	0	0	0	0,2	0,2
#0099	[TVA10]	0,2	0	0	0,2	0	0,4
#0110	[WPH+14]	0,2	0	0	0,2	0,2	0,6
#0121	[YS15]	0,2	0	0,2	0	0	0,4
#0141	[HTP+10]	0,2	0	0,2	0,2	0	0,6
#0145	[HyY06]	0,2	0	0,2	0	0,2	0,6
#0168	[PBT10]	0,2	0	0,2	0	0	0,4
#0169	[GDN+12]	0,2	0	0,2	0	0,2	0,6
#0170	[NS10]	0,2	0	0	0,2	0,2	0,6
#0171	[HY06]	0,2	0	0,2	0	0,2	0,6
#0172	[NEI12]	0,2	0	0,2	0,2	0	0,6
#0173	[BMF07]	0,2	0	0	0,2	0,2	0,6
#0174	[MPH+97]	0,2	0	0,2	0	0,2	0,6
#0175	[PHHJ10]	0,2	0	0	0	0,2	0,4
#0180	[HBQS13]	0,2	0	0	0	0	0,2
#0181	[SC13]	0,2	0	0,2	0	0	0,4
#0185	[KO03]	0,2	0	0,2	0	0	0,4
#0187	[SWB+97]	0,2	0	0,2	0,2	0	0,6
#0192	[GBK12]	0,2	0,2	0,2	0,2	0	0,8
#0194	[GKB12]	0	0	0	0,2	0,2	0,4
#0195	[NMS12]	0	0	0	0,2	0,2	0,4
#0196	[TH13]	0	0	0	0	0	0
#0197	[CYSK14]	0,2	0	0,2	0,2	0,2	0,8
#0199	[HZHW11]	0	0	0	0,2	0	0,2
#0205	[KG12]	0,2	0	0	0,2	0	0,4
#0223	[GKG13]	0,2	0	0	0,2	0	0,4
#0245	[NS11]	0,2	0	0	0,2	0,2	0,6
#0262	[CLA+12]	0,2	0,2	0,2	0,2	0	0,8
#0291	[CLC13]	0,2	0,2	0,2	0	0	0,6
#0330	[ZAS+12]	0,2	0	0,2	0,2	0,2	0,8

#0332	[AZS+11]	0,2	0	0,2	0,2	0,2	0,8
#0333	[ZAC+09]	0,2	0	0,2	0,2	0,2	0,8
#0337	[SS12]	0,2	0	0	0,2	0	0,4
#0339	[ASZ+12]	0	0	0	0,2	0	0,2
#0344	[NSM11]	0,2	0	0,2	0,2	0	0,6
#0345	[TBL+12]	0,2	0,2	0	0,2	0	0,6
#0346	[SHWK15]	0,2	0	0,2	0,2	0	0,6
#0354	[ZFB+14]	0,2	0	0	0	0	0,2
#0356	[SLL12a]	0	0	0	0,2	0	0,2
#0358	[SLZB14]	0	0	0	0,2	0	0,2
#0363	[MSK+09]	0	0	0	0,2	0	0,2
#0366	[GPW+13]	0	0	0	0,2	0	0,2
#0368	[RHZE15]	0,2	0,2	0,2	0	0	0,6
#0369	[SFF14]	0	0	0	0,2	0	0,2
#0370	[KSL12]	0	0	0	0,2	0	0,2
#0371	[HSS+11]	0,2	0	0,2	0,2	0,2	0,8
#0374	[Tid10]	0,2	0	0,2	0,2	0	0,6
#0385	[SPM13]	0,2	0	0	0,2	0	0,4
#0390	[RBL+09]	0	0	0,2	0	0,2	0,4
#0393	[ZCLL14]	0,2	0	0,2	0	0,2	0,6
#0395	[AABL12]	0,2	0	0	0	0,2	0,4
#0396	[KEB07]	0	0	0	0,2	0	0,2

Tabelle 4.1: Qualitative Untersuchung der Beiträge zu RQ-1

Die Abbildung 4.4 illustriert, wie häufig jeweils in der jeweiligen Kategorie ein Beitrag genannt wurde. Demnach befassen sich 48 Beiträge aus der Auswahl mit dem Thema “active safety”, während in der Kategorie “consumer test” fünfmal 0,2 Punkte vergeben wurden. 29 Beiträge befassen sich inhaltlich mit einem Warn- oder Notbremssystem und weitere 38 Beiträge verfügen über einen industriellen Hintergrund. Eine Textpassage als kritische Auseinandersetzung enthalten 25 der 64 ausgesuchten Beiträge.

In der Abbildung 4.5 wird die Verteilung der erzielten Punkte für jeden Beitrag dargestellt. So erzielen 2 Beiträge 0 Punkte, 14 Beiträge 0,2 Punkte, jeweils 21 und 19 Beiträge die Punktezahlen 0,4 und 0,6. Weitere 8 Beiträge konnten die Punktzahl von 0,8 erreichen, während die Bestbewertung von 1 Punkt hier nicht erzielt wurde.

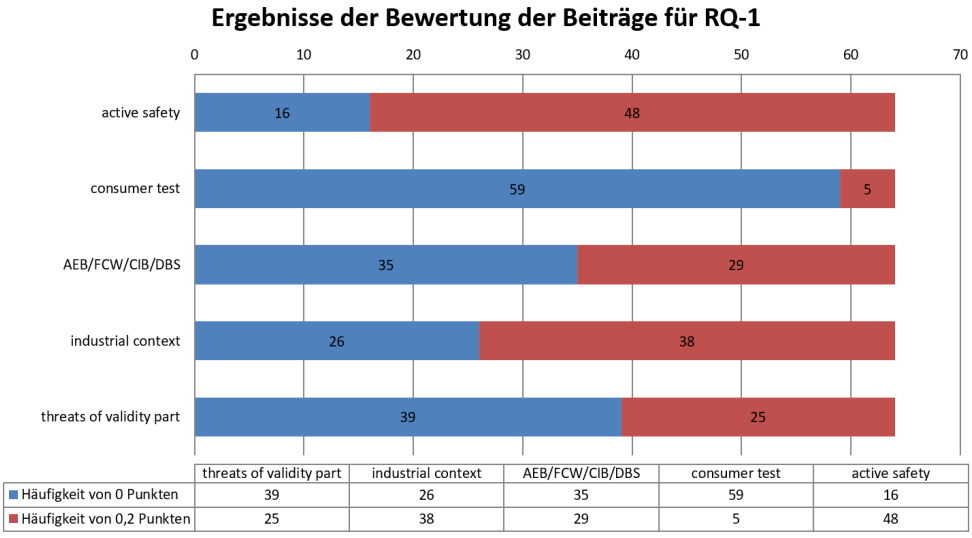


Abbildung 4.4: Ergebnisse der Bewertung der Beiträge für RQ-1

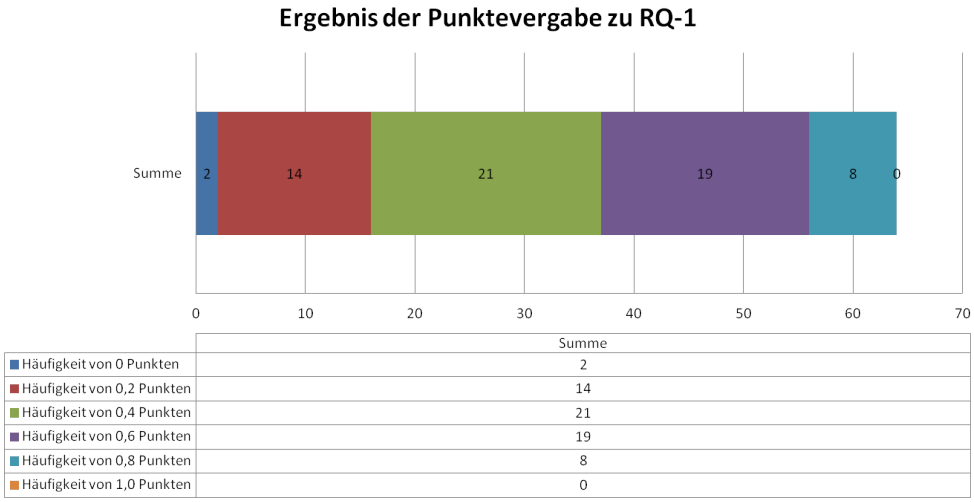


Abbildung 4.5: Ergebnis der Punktevergabe für RQ-1

4.4.3 RQ-2: Toleranzuntersuchung und ihre Bewertung - Ergebnisse

Um die zweite Forschungsfrage hinreichend beantworten zu können, wurden weitere Kriterien definiert, die in gleicher Weise ermitteln, wie umfassend ein Papier zur Antwort beitragen kann. Da eine gewisse Abhängigkeit hinsichtlich der relevanten Papiere aus der ersten Forschungsfrage erwächst, ist eine gewisse Qualität hieraus erforderlich. Daher wird für die zweite Frage auf das Ergebnis der ersten zurückgegriffen und diejenigen Papiere ausgewählt, die mindestens 0,6 Punkte erhalten haben. Abgesehen von Spalte 3 sind die übrigen Spalten analog zur Tabelle 4.1 zu lesen; Spalte 3 spiegelt die halbierte Punktzahl aus RQ-1 wider. In Tabelle 4.2 ist das Bewertungsergebnis für RQ-2 zu entnehmen.

Study ID	Reference ID	Halbe Punkt-zahl	Einzel-Kompo-nente	ganzes System	Tole-ranzen-model-lierung	Metho-denbe-schrei-bung	automatische Parametervа-riation o. Zu-fallsvariable	Summe für RQ-2
#0061	[GPSV09]	0,4	0,2	0,2	0	0,2	0	1,0
#0083	[LCY+14]	0,3	0,2	0	0	0	0	0,5
#0110	[WPH+14]	0,3	0,2	0,2	0	0	0	0,7
#0141	[HTP+10]	0,3	0,2	0,2	0	0,2	0	0,9
#0145	[HyY06]	0,3	0,2	0,2	0	0,2	0	0,9
#0169	[GDN+12]	0,3	0	0,2	0	0,2	0	0,7
#0170	[NS10]	0,3	0,2	0,2	0	0,2	0	0,9
#0171	[HY06]	0,3	0,2	0,2	0	0,2	0	0,9
#0172	[NEI12]	0,3	0,2	0,2	0,2	0,2	0,2	1,3
#0173	[BMF07]	0,3	0,2	0,2	0	0,2	0	0,9
#0174	[MPH+97]	0,3	0	0,2	0,2	0,2	0	0,9
#0187	[SWB+97]	0,3	0,2	0	0,2	0,2	0	0,9
#0192	[GBK12]	0,4	0,2	0,2	0	0,2	0	1,0
#0197	[CYSK14]	0,4	0,2	0,2	0	0,2	0	1,0
#0245	[NS11]	0,3	0,2	0,2	0	0,2	0	0,9
#0262	[CLA+12]	0,4	0,2	0,2	0	0,2	0	1,0
#0291	[CLC13]	0,3	0,2	0,2	0	0,2	0	0,9
#0330	[ZAS+12]	0,4	0,2	0,2	0	0,2	0	1,0
#0332	[AZS+11]	0,4	0,2	0,2	0	0,2	0	1,0
#0333	[ZAC+09]	0,4	0,2	0,2	0	0,2	0	1,0
#0344	[NSM11]	0,3	0,2	0,2	0	0,2	0	0,9
#0345	[TBL+12]	0,3	0,2	0,2	0	0,2	0	0,9
#0346	[SHWK15]	0,3	0,2	0,2	0	0,2	0	0,9
#0368	[RHZE15]	0,3	0	0,2	0	0,2	0	0,7

#0371	[HSS+11]	0,3	0	0,2	0	0,2	0	0,7
#0374	[Tid10]	0,3	0,2	0,2	0	0,2	0	0,9
#0393	[ZCLL14]	0,3	0,2	0,2	0	0,2	0	0,9

Tabelle 4.2: Qualitative Untersuchung der Beiträge zu RQ-2

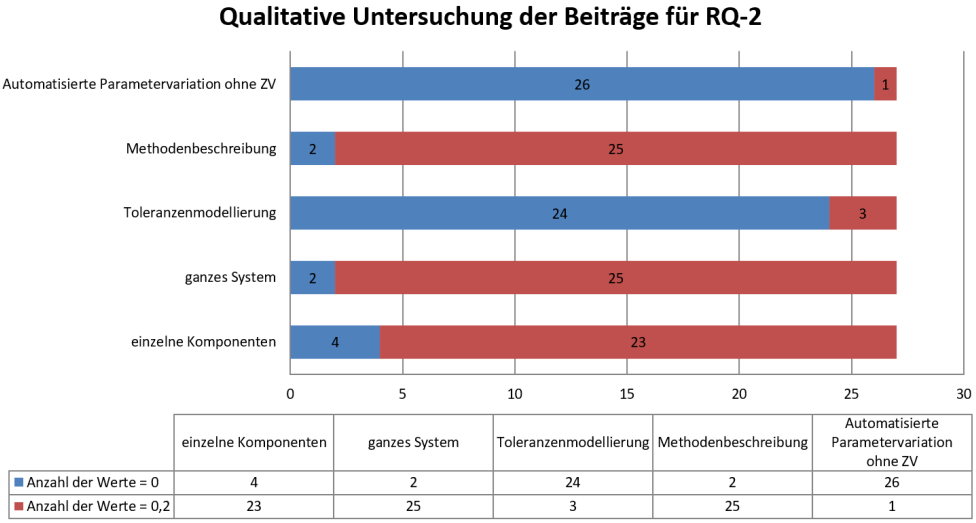


Abbildung 4.6: Ergebnisse der Bewertung der Beiträge für RQ-2

Aus der Tabelle 4.2 und der Abbildung 4.6 geht hervor, dass 23 Beiträge über jeweils einen Ansatz verfügen, der sich auf eine einzelne Komponente und 25 Beiträge auf ein ganzes System beziehen. Hingegen beschäftigen sich nur 3 der 27 verbliebenen Beiträge mit der Modellierung von Toleranzen. Lediglich ein einzelner Beitrag aus der Auswahl beschreibt eine automatisierte Parametervariation, die nicht auf Zufallsgrößen wie z.B. einer Monte-Carlo-Simulation basiert. Über eine Methodenbeschreibung verfügen bis auf 2 Beiträge alle übrigen.

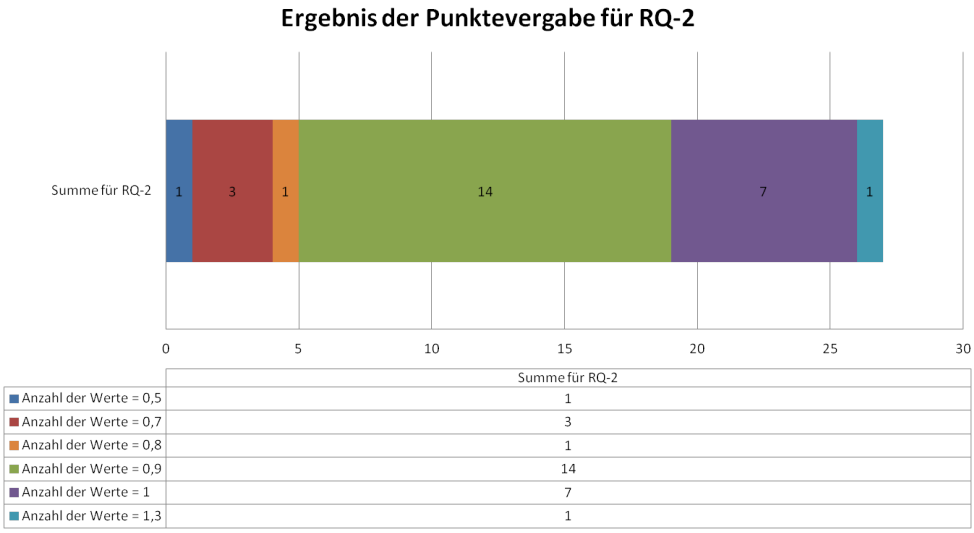


Abbildung 4.7: Ergebnis der Punktevergabe für RQ-2

Ein Beitrag konnte lediglich die Punktzahl von 0,5 erreichen, vier weitere Beiträge immerhin 0,7 Punkte. Eine Mehrheit an Beiträgen, hier 14 Papiere umfassend, erzielte eine Punktzahl von 0,9 Punkten, während 7 Beiträge eine Punktzahl von 1,0 erreichten. Ein Beitrag konnte die höchste Bewertung mit 1,3 Punkten erlangen. Die übrigen Punktzahlen inklusive der Höchstpunktzahl von 1,5 Punkten blieben unbesetzt.

4.4.4 RQ-3: Methodische Entwicklung von Simulationsinfrastruktur - Ergebnisse

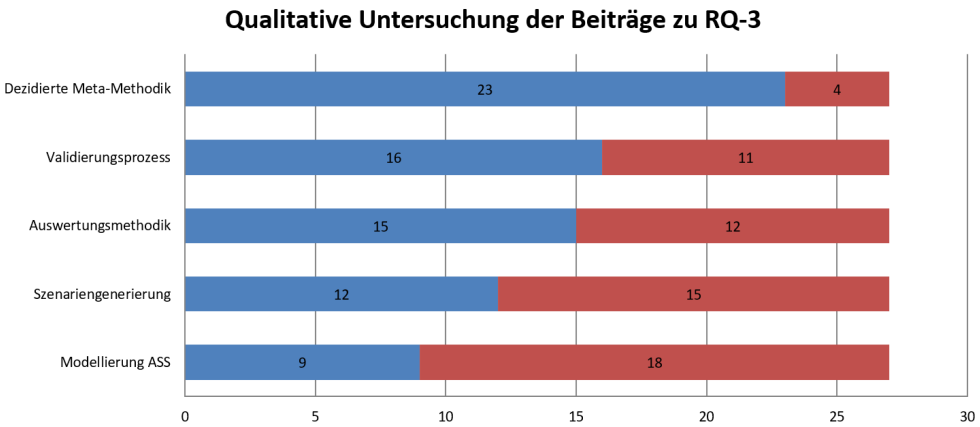
Abschließend seien in diesem Abschnitt die Ergebnisse zur Bewertung der Beiträge hinsichtlich der Forschungsfrage RQ-3 beschrieben. Die nachfolgende Tabelle 4.3 umfasst die Einordnung der fünf identifizierten Kriterien Modellierung eines Aktiven Sicherheitssystems, Szenariengenerierung, Auswertungsmethodik, Toleranzenmodellierung, Validierungsprozess und dezidierte Meta-Methodik analog zu den vorangegangenen Forschungsfragen. Wie bereits oben erläutert, werden hier die gleichen Beiträge wie in RQ-2 betrachtet.

Study ID	Reference ID	Modellierung ASS	Szenariengenerierung	Auswertungsmethodik	Validierungsprozess	Dezidierte Meta-Methodik	Summe für RQ-3
#0061	[GPSV09]	0,2	0	0	0,2	0	0,4
#0083	[LCY+14]	0,2	0	0	0,2	0	0,4
#0110	[WPH+14]	0	0,2	0	0	0,2	0,4

#0141	[HTP+10]	0,2	0,2	0	0	0	0,4
#0145	[HyY06]	0,2	0	0	0,2	0	0,4
#0169	[GDN+12]	0,2	0,2	0	0	0	0,4
#0170	[NS10]	0	0,2	0	0,2	0	0,4
#0171	[HY06]	0,2	0	0	0,2	0	0,4
#0172	[NEI12]	0	0,2	0,2	0	0,2	0,6
#0173	[BMF07]	0	0	0,2	0,2	0	0,4
#0174	[MPH+97]	0,2	0,2	0,2	0	0	0,6
#0187	[SWB+97]	0,2	0	0	0	0	0,2
#0192	[GBK12]	0,2	0	0	0	0	0,2
#0197	[CYSK14]	0,2	0	0	0	0	0,2
#0245	[NS11]	0	0,2	0,2	0,2	0	0,6
#0262	[CLA+12]	0	0,2	0,2	0	0	0,6
#0291	[CLC13]	0	0,2	0,2	0	0	0,6
#0330	[ZAS+12]	0,2	0,2	0,2	0,2	0	0,8
#0332	[AZS+11]	0,2	0,2	0,2	0,2	0	0,8
#0333	[ZAC+09]	0,2	0,2	0,2	0,2	0	0,8
#0344	[NSM11]	0,2	0	0,2	0	0	0,4
#0345	[TBL+12]	0,2	0,2	0	0	0,2	0,6
#0346	[SHWK15]	0	0	0	0	0,2	0,2
#0368	[RHZE15]	0,2	0,2	0,2	0	0	0,6
#0371	[HSS+11]	0	0	0,2	0,2	0	0,4
#0374	[Tid10]	0,2	0	0	0	0	0,2
#0393	[ZCLL14]	0,2	0,2	0	0	0	0,4

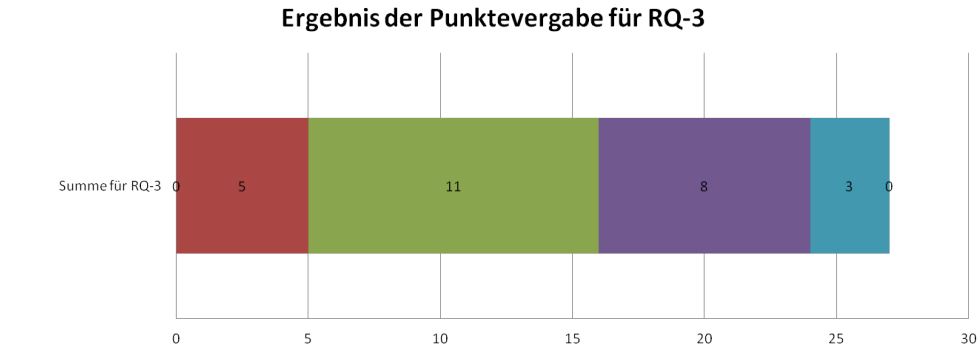
Tabelle 4.3: Qualitative Untersuchung der Beiträge zu RQ-3

Demnach können gemäß Abbildung 4.8 18 der 27 aufgeführten Beiträge mit einer Modellierung eines Aktiven Sicherheitssystems identifiziert werden, 15 Beiträge verwenden zudem eine Szenariengenerierung in ihrem Ansatz. Eine konkrete Auswertungsmethodik wird in 12 der 27 Beiträge genannt und in 11 Beiträgen wird ein Prozess der Validierung angesprochen. Hingegen wird in nur 4 Beiträgen eine dezidierte Meta-Methodik erläutert. Somit ergeben sich als summierte Punktzahlen folgende Verteilung (siehe Abbildung 4.9): Es existieren keine Beiträge ohne ein Kriterium. 5 Papiere erfüllen immerhin ein Kriterium und erreichen damit 0,2 Punkte. Weitere 14 Beiträge können eine Punktzahl von 0,4 Punkten vorweisen und weitere 5 Papiere erhalten bei der Bewertung 0,6 Punkte. Die Punktezahl 0,8 erreichen insgesamt 3 Papiere, während eine Bestbewertung von einem Punkt nicht erzielt werden konnte.



	Modellierung ASS	Szenariengenerierung	Auswertungsmethodik	Validierungsprozess	Dezidierte Meta-Methodik
Anzahl der Werte = 0	9	12	15	16	23
Anzahl der Werte = 0,2	18	15	12	11	4

Abbildung 4.8: Ergebnisse der Bewertung der Beiträge für RQ-3



	Summe für RQ-3
Anzahl der Werte = 0	0
Anzahl der Werte = 0,2	5
Anzahl der Werte = 0,4	11
Anzahl der Werte = 0,6	8
Anzahl der Werte = 0,8	3
Anzahl der Werte = 1	0

Abbildung 4.9: Ergebnis der Punktevergabe für RQ-3

4.5 Diskussion und Analyse der Ergebnisse

Im nun folgenden Abschnitt sollen diese Ergebnisse zu den einzelnen Forschungsfragen näher betrachtet und bewertet werden.

4.5.1 RQ-1: Konzeptsuche zur Simulation von Consumer Tests

Hinsichtlich der Bewertung der Beiträge zu RQ-1 kann dem Ergebnis entnommen werden, dass sich 48 von 64 der Beiträge auf Aktive Sicherheit beziehen, aber nur 29 von 64 auf ein Warn- oder Notbremssystem. Somit gibt es auch keinen Beitrag, der sich umgekehrt auf ein solches System, aber nicht auf aktive Sicherheit bezieht. Allerdings können insgesamt nur 5 Beiträge einen Bezug zu Consumer Tests herstellen; 4 davon beziehen sich auf einen AEB/FCW Kontext, ein einzelnes Papier hat ein LKA System im Fokus.

Eine indifferente Sicht ergibt sich darauf, ob ein Papier in Rahmen eines industriellen Kontextes eingebettet ist oder nicht. Zwar ist der Anteil an Beiträgen mit industriellem Hintergrund etwas höher als rein universitäre Beiträge, so dass das Verhältnis sich auf 38 zu 26 beläuft und somit auch den besonderen Stellenwert dieser Themen für die Industrie kennzeichnet; allerdings lassen sich daraus keine Kausalitäten ableiten, ob der jeweilige Beitrag über einen Abschnitt zur Validitätsbetrachtung verfügt. Weder eine besondere Vernachlässigung noch eine besondere Häufigkeit an solchen Erörterungen lassen sich dem einen oder anderen Lager zuschreiben. Auch indiziert ein universitärer Hintergrund nicht zwingend einen Abschnitt bzgl. Validität. Nimmt man beispielsweise alle Beiträge, die Aktive Sicherheit umfassen, so ergeben sich folgende Verteilung:

Industrie/ToV-P	0	0,2	Summe
0	14	12	26
0,2	25	13	38
Summe	39	25	64

Tabelle 4.4: Vier-Felder-Tafel zu RQ-1: Industrial Context und Threats to Validity Part

Industrie/ToV-P	0	0,2	Summe
0	21,9%	18,8%	40,7%
0,2	39,1%	20,3%	59,4%
Summe	61,0%	39,1%	$\approx$ 100,0%

Tabelle 4.5: Relative Vier-Felder-Tafel zu RQ-1: Industrial Context und Threats to Validity Part

Somit ist für diese Studie die Wahrscheinlichkeit, dass ein Beitrag unabhängig des Kontextes eine Threats to Validity Section umfasst, mit etwa $\approx 0{,}2$ gleich hoch. Hingegen ist im Falle des Fehlens eines solchen Abschnittes die Wahrscheinlichkeit mit 0,39, dass es sich um einen industriellen Kontext handelt, größer als für einen rein universitären Kontext mit $\approx 0{,}22$.

Durch die geringe Anzahl an Beiträgen, die sich konkret auf Consumer Tests beziehen (hier: 5), war es entscheidend, die Punktezahl, die für die nachfolgenden Fragen berücksichtigt werden sollen, auf 0,6 einzubeziehen. Insgesamt liegt die Wahrscheinlichkeit, dass ein Papier 0,6 Punkte oder mehr erzielt, bei ca. 0,422.

4.5.2 RQ-2: Toleranzuntersuchung und ihre Bewertung

Durch die Wahl einer Mindestpunktzahl von 0,6 verbleiben 27 Ansätze aus der vorherigen Forschungsfrage. Die weitere Auswahl wurde anhand der Punktzahl festgelegt, da eine reine Konzentration auf ein spezielles Kriterium wie z.B. Consumer Tests als nicht hinreichend genug angesehen wurde. Denn ein Beitrag kann auch dann wertvoll sein und einen guten Ansatz liefern, wenn die Simulation zunächst einen anderen Fokus hat. Gleichzeitig wird diese Punktegrenze als ausreichend angesehen, da hierdurch ein Mindestmaß an notwendig erfüllten Kriterien für eine weitere Betrachtung sichergestellt wird.

Ein großer Teil dieser 27 Beiträge erlaubt die Modellierung eines Systems als Ganzes. Dabei ist jedoch nicht entscheidend, welchen Detailgrad bzw. welche Komplexität das Modell des Systems letztlich aufweist, also ob das Modell durch viele einzelne mathematische Gleichungen oder durch eine einzelne mathematische Funktion definiert ist. Auch erlauben die meisten Ansätze einzelne Systemkomponenten zu untersuchen und in den Mittelpunkt der Simulationsbestrebungen zu stellen, während die übrigen Komponenten dann ergänzend nachgebildet und/oder idealisiert werden. Nur wenige Beiträge erlauben ausschließlich die Simulation von einzelnen Komponenten oder ganzen Systemen, so dass auf eine Modularität verzichtet wurde.

Komponente/System	0	0,2	Summe
0	0	4	4
0,2	2	0	23
Summe	2	25	27

Tabelle 4.6: Vier-Felder-Tafel zu RQ-2: Modellierung Einzel-Komponenten bzw. ganzes System

Komponente/System	0	0,2	Summe
0	0,0%	14,8%	14,8%
0,2	7,4%	77,8%	85,2%
Summe	7,4%	92,6%	100,0%

Tabelle 4.7: Relative Vier-Felder-Tafel zu RQ-2: Modellierung Einzel-Komponenten bzw. ganzes System

Insgesamt ist die Wahrscheinlichkeit, dass ein Ansatz sowohl eine einzelne Komponente als auch ein ganzes System abbilden kann, mit 0,778 vergleichsweise hoch, während die Wahrscheinlichkeit für eine ausschließliche Abbildung von einzelnen Komponenten oder eines Gesamtsystems mit 0,074 respektive 0,148 deutlich geringer ist.

Hingegen wird in kaum einem Beitrag auf eine konkrete Toleranzenmodellierung eingegangen, wie beispielsweise von Fahrkurven oder andere Eingangsgrößen. Vielmehr werden in den Beiträgen Testarten genannt, die auf die reine Untersuchung von Funktionsfähigkeit ausgerichtet sind. Die Untersuchung, wie gut eine Funktion in fast identischen Szenarien auslöst, ist hingegen noch nicht Gegenstand der hier aufgeführten Konzepte. Nur ein einzelner Beitrag befasst sich mit beiden Aspekten, zwei weitere immerhin mit einer Toleranzenmodellierung. Die Wahrscheinlichkeit, dass ein Beitrag keine Erwähnung der Kriterien enthält, liegt bei $\approx 0{,}89$.

Toleranzen/Variation	0	0,2	Summe
0	24	0	24
0,2	2	1	3
Summe	26	1	27

Tabelle 4.8: Vier-Felder-Tafel zu RQ-2: Toleranzmodellierung und automatisierte Parametervariation ohne Zufallsvariable

Toleranzen/Variation	0	0,2	Summe
0	88,9%	0,0%	88,9%
0,2	7,4%	3,7%	11,1%
Summe	96,3%	3,7%	100,0%

Tabelle 4.9: Relative Vier-Felder-Tafel zu RQ-2: Toleranzmodellierung und automatisierte Parametervariation ohne Zufallsvariable

Die Methodenbeschreibung ist in der Regel gegeben, so dass auf eine tiefere Analyse verzichtet wird; denn nur 2 Beiträge geben darauf keine hinreichende Antwort. Die hier erwähnte Methodenbeschreibung bezieht sich jedoch allein auf eine Beschreibung einer funktionalen Lösung;

die übergeordnete Methodik, mit welcher nach einer technischen Lösung gesucht und im Anschluss umgesetzt wurde, wird nicht durch dieses Kriterium abgedeckt.

4.5.3 RQ-3: Methodische Entwicklung von Simulationsinfrastruktur

Den Grafiken aus 4.8 und 4.9 kann entnommen werden, dass sich aus den ausgewählten Beiträgen zwei Drittel auf die Modellierung eines Aktiven Sicherheitssystems konzentrieren. Auch die Szenariengenerierung ist bei etwas mehr als der Hälfte der Beiträge ein wichtiger Aspekt.

ASS Modellierung/Szenariengenerierung	0	0,2	Summe
0	3	6	9
0,2	9	9	18
Summe	12	15	27

Tabelle 4.10: Vier-Felder-Tafel zu RQ-3: Modellierung von Aktiven Sicherheitssystemen und einer Szenariengenerierung

ASS Modellierung/Szenariengenerierung	0	0,2	Summe
0	11,1%	22,2%	33,3%
0,2	33,3%	33,3%	66,6%
Summe	44,4%	55,5%	$\approx$ 100,0%

Tabelle 4.11: Relative Vier-Felder-Tafel zu RQ-3: Modellierung von Aktiven Sicherheitssystemen und einer Szenariengenerierung

Bezogen auf die Auswahl der in dieser Studie untersuchten Beiträge beträgt die Wahrscheinlichkeit, dass weder das Kriterium einer Systemmodellierung noch eines zur Szenariengenerierung erfüllt ist, 0,11. Eine gleich hohe Wahrscheinlichkeit mit 0,33 ergibt sich für den Fall, dass der Beitrag sich auf eine ASS Modellierung bezieht und dabei entweder eine Szenariengenerierung enthalten oder nicht enthalten ist. Eine reine Konzentration auf die Szenariengenerierung hat hier eine Wahrscheinlichkeit von 0,22.

Auswertungsmethodik/Validierungsprozess	0	0,2	Summe
0	10	5	15
0,2	6	6	12
Summe	16	11	27

Tabelle 4.12: Vier-Felder-Tafel zu RQ-3: Auswertungsmethodik und ein Validierungsprozess

Auswertungsmethodik/Validierungsprozess	0	0,2	Summe
0	37,0%	18,5%	55,5%
0,2	22,2%	22,2%	44,4%
Summe	59,2%	40,7%	$\approx$ 100,0%

Tabelle 4.13: Relative Vier-Felder-Tafel zu RQ-3: Auswertungsmethodik und ein Validierungsprozess

Hier ergibt sich ein recht indifferentes Bild, so dass sich eigentlich keine bedingte Abhängigkeit ableiten lässt. Lediglich die Wahrscheinlichkeit, dass weder Auswertungsmethodik noch Validierungsprozess als Kriterium erfüllt sind, ist mit 0,37 vergleichsweise hoch zu den anderen Kombinationen.

Das Kriterium für eine dezidierte Meta-Methodik können nur vier der 27 Papiere erfüllen. Eine Verknüpfung mit einem zweiten Kriterium erscheint an dieser Stelle nicht erforderlich, da sich eine inhaltliche Abhängigkeit nicht ergibt. Es bleibt jedoch festzuhalten, dass zu dieser Studie nur wenige Papiere vorhanden sind.

4.6 Inhaltliche Analyse der Beiträge

Welche Aussagen hinsichtlich der formulierten Forschungsfragen getroffen werden können, soll in dem folgenden Abschnitt untersucht werden. Zudem werden hier die relevanten Ansätze inhaltlich vorgestellt. Da sich viele Beiträge in einer gewissen Punktzahl mit anderen Beiträgen gleichen, stellt sich zunächst die Frage nach der zielführenden Reihenfolge für eine inhaltliche Betrachtung. Wegen der Prüfung auf Relevanz für Consumer Tests wird dieses Kriterium für eine diesbzgl. Unterscheidung herangezogen.

Begonnen wird somit zur RQ-1 mit denjenigen Beiträgen, welche die höchste Punktzahl in der Bewertung erhalten haben UND das Kriterium zu “Consumer Tests” erfüllt haben. Anschließend folgen die Beiträge, welche weniger Punkte erhalten haben und ebenfalls das zuletzt genannte Kriterium positiv bewertet wurde. Danach folgenden die Beiträge mit den höchsten Punktzahlen. Betrachtet werden nur diejenigen Beiträge, die mehr als 0,6 Punkte erreicht haben. Dabei werden Beiträge, welche eine ähnliche Autorenschaft umfassen und sich auch inhaltlich ähneln bzw. aufeinander aufbauen, im Zusammenhang dargestellt.

Für RQ-2 und RQ-3 wird dann erneut ausgehend von der höchsten Punktzahl und dann absteigend inhaltlich zur Forschungsfrage Bezug genommen.

4.6.1 RQ-1: Konzeptsuche zur Simulation von Consumer Tests

Der Beitrag von Garate et al. handelt von der numerischen Modellierung von ADAS für Fußgängerschutz mit Hilfe von Radar und Kamerasensorik und stellt eine Simulationsmethode vor, welche zur Funktionsentwicklung und zum Test von Fußgängerschutzsystemen genutzt werden kann; dazu verwenden sie eine Reihe von Software Tools. Neben PREScan und Matlab ist ein weiteres Framework integriert, welches beim virtuellen Design und beim Test von Aktiven Sicherheitssystemen zum Einsatz kommt. Beschrieben werden zudem die relevanten Unfallszenarien mit Fußgängern und Radfahrern und anschließend das System selbst sowie die Funktionsweise inkl. der Fusion beider Sensoren. [GBK12]

Der Beitrag von Chien et al. befasst sich mit der Entwicklung einer Auswertungsmethode für CIB Systeme. Dabei schaffen sie eine Metrik, mit der sie den prozentualen Energieabbau messen und damit eine Vergleichbarkeit verschiedener Systeme gewährleisten wollen. Zudem zeigen sie auf, wie systematisch eine Bewertung der Performance durchgeführt werden kann und unter welchen Annahmen diese sinngerecht eingesetzt werden kann. In [CLA+12] wird diese Methode zunächst exemplarisch für einige allgemeine Szenarien und ein CIB System aus dem Jahr 2011 evaluiert, während in [CLC13] sowohl für CBS als auch für CIB Systeme in Kombination dieselbe Methodik angewendet wird. Der Fokus liegt hier überwiegend auf der Evaluation von realen Testszenarien. Allerdings schlagen die Autoren auch die Anwendung als simulationsbasierte Tests und konkret als mögliche Consumer Test Methodik vor. [CLA+12, CLC13]

Raudszus et al. stellen in ihrem Beitrag eine Methodik vor, mit der insbesondere die neuen Herausforderungen für Fahrzeuge hinsichtlich des Fußgängerschutzes bewertet werden können. Hierbei verweisen sie insbesondere die auf ab 2016 gestiegenen Anforderungen für die Consumer Tests wie z.B. beim EuroNCAP. Die Methodik zielt darauf ab, den Nutzen von aktiven und passiven Maßnahmen zum Fußgängerschutz zu bestimmen, wie bspw. die Reaktionszeit des Fahrers, Geschwindigkeitsreduktion und die daraus resultierende Verletzungsschwere durch einen möglichen Anprall. Dabei liegt der Fokus auf einer umfassenden Bewertung von Technologien im Vorfeld möglicher Entwicklungen und den Randbedingungen der jeweiligen Szenarien. [RHZE15]

Tideman et al. stellen ebenfalls eine Methode zur Simulation während der Entwicklung von Fahrerassistenzsystemen vor. Dabei steht das Software Paket PreSCAN als zentrales Element auch in ihren anderen Arbeiten im Fokus. Im oben genannten Beitrag geht es um die Verknüpfung mit der Fahrdynamik von dSPACE, welche auf MatLab/Simulink basiert. Der Bezug zu Consumer Test wird im Rahmen der Vorstellung von traditionellen Testmethoden vorgenommen. Zwar beziehen sich die Autoren hier auf die passiven Sicherheitsfunktionen, jedoch wird sich dabei auf

Forschungsprogramme gestützt, welche das Ziel der Standardisierung auch für Aktive Systeme umfasst. Als Fallbeispiel dient hier ein LKA System als Basis, so dass nicht direkt ein Bezug zu der Fragestellung in dieser Arbeit besteht. In Tideman et al. 2009 wird eine Verknüpfung zwischen der Arbeit von Tideman et al. und der Arbeit von Gietlink et al. hergestellt. Gietlink et al. haben eine umfangreiche Anlage zur Simulation von Fahrszenarien mit Hilfe eines realen Fahrzeugs auf einem Rollenprüfstand und einer sich in einer Halle bewegenden hochdynamisch fahrenden Plattform nachgebildet. Über die Auswertung der Relativgeschwindigkeit wird dann die Fahrszene für das System erfassbar. [Tid10, TBL+12, GPSV09]

Hier werden auch Prozessthemen adressiert, entsprechend für die Entwicklung erforscht und abgebildet. Beispielsweise wird hier die Entwicklung von Fahrerunterstützungssystemen durch eine entsprechende Methodik beschrieben. Dabei steht im Fokus, entsprechende Entwürfe direkt den potentiellen Kunden im Fahrsimulator zu präsentieren und auf diese Weise in den Entwicklungsprozess aktiv einzubinden. [TVA10]

Es liegen hier bereits Beiträge vor, die keinen direkten Bezug zu Consumer Tests herstellen, jedoch inhaltlich eine Verknüpfung aufweisen. Wie bereits beschrieben, werden sie jedoch im Zusammenhang zu den “Leitartikeln” genannt. Bei allen weiteren Beiträgen konnte keine direkte Verknüpfung mit Consumer Tests erfolgen; dennoch lassen gewisse Aspekte eine eventuelle Übertragung zu und seien ergänzend erwähnt.

In der Arbeit von Lee et al. steht zunächst die Entwicklung einer Funktion im Vordergrund, welche in einer Notsituation durch gezielte Eingriffe bei Lenkung und einzelnen Bremsen eine Unterstützung für Ausweichmanöver bietet. Diese Funktion wird jedoch mit Hilfe eines full scale Fahrsimulators erprobt. [CYSK14] Dieser wird in Donghoon et al. im Rahmen einer Vergleichsstudie von ACC und Kollisionsvermeidungsalgorithmen näher beschrieben. [HY06, HyY06] Auch ein LKA System wurde mit Hilfe des Virtual Test Tracks näher untersucht. [LCY+14]

Im Rahmen der Forschungsarbeit von van Auken, Zellner et al. der Dynamic Research Inc. in Zusammenarbeit mit Honda ist eine Methodik zur Bewertung des Sicherheitsniveaus von Notbremssystemen entwickelt worden. Diese wurde unterstützt von der NTHSA, um zwei Ziele zu erreichen:

1. Zur Entwicklung einer Methode zur standardisierten Safety Impact Methodology (SIM) und
2. zur Entwicklung von objektiven Tests und der späteren Demonstration anhand eines realen CIB Systems.

Dabei ging es vorrangig um die theoretische und softwareseitige Erweiterung der Safety Impact

Methodik und des zugrundeliegenden Tools (objektive Testprozeduren). Die Charakteristik und Performance einer Sicherheitstechnologie sollte zudem quantifizier-, wiederhol- und reproduzierbar gestaltet werden, damit diese auf einem Prüfgelände festgestellt werden kann. Weiterhin wurden Fahrerverhaltensstudien in solchen Szenarien durchgeführt, um das Verhalten von Fahrern zu erfassen und diese Daten als Vergleichsgrundlage zu nutzen. Außerdem wurde ein Rekonstruktionstool zur Erstellung von Szenarien aus Unfalldaten entwickelt. Letztlich erfolgte die beispielhafte Anwendung der gesamten Methodik in Verbindung mit einem in der Entwicklung befindlichen Notbremssystems von Honda. Insgesamt handelt die Forschungsarbeit von der Effektivitätsbestimmung Aktiver Sicherheitssysteme [ZAC+09, AZS+11, ZAS+12]. Im Beitrag von Stapleford et al. wird der Fahrsimulator von DRI und Honda und seine Anwendung in Verbindung mit einem ACC System beschrieben. [SWB+97]

Die Arbeit von Helmer et al. konzentriert sich ebenfalls auf die Effektivitätsbewertung von Aktiven Sicherheitssystemen mit dem Fokus auf Fußgängerschutzsysteme. Konkret bezieht sich die Arbeit insgesamt auf die Monte-Carlo Simulation als stochastische Simulationsmethode. Es wird angemerkt, dass bei der Bewertung die Physik eine zentrale Rolle spielt, hingegen die Umrechnung auf die Vorteile für den Menschen Verletzungs- und Schadensrisiko-Modelle erfordert. Dies ermöglicht eine multivariate Untersuchung über die Effektivität von Aktiven Fußgängerschutzsystemen anhand von US Unfalldaten und mit besonderer Betrachtung der Verletzungsschwere von Fußgängern. [HSS+11]

In der Arbeit von Gruyer et al. wird eine Simulationsarchitektur vorgestellt, die sich auf kooperative Kollisionswarnsysteme als Testsystem konzentriert. Dazu wird die bereits vorhandene Sensor Simulationsplattform SiVIC mit einer Protyping platform RTmaps(TM) verbunden, wodurch reale Messungen mit simulierten Daten ersetzt werden können. Trotz des Bezuges zu einer Car-2-X Technologie ist dieser Beitrag in der Wertung geblieben, da sich ein signifikanter Anteil auf Simulationsmethoden und ein Kollsionsvermeidungssystem bezieht. [GDN+12]

Noth et al. präsentieren in ihrem Ansatz eine Methodik mit einer integrierten Architektur, basierend auf einem simulierten Realitätsframework. Darin enthalten sind einfache Verhaltensmodelle von autonom agierenden Menschen, sich physikalisch plausibel verhaltende Szeneobjekte und ein erweitertes Aufzeichnungs- und Auswertungssystem. Ziel ist die Entwicklung, Bewertung und Auswertung von Fahrerassistenzsystemen und deren Komponenten. Der Beitrag beschreibt die grundlegenden Softwarekomponenten, welche das Framework realisieren. [NEI12]

McMillan, Pape et al. befassen sich in ihrem Beitrag ähnlich wie Helmer et al. mit einer statistikbasierenden Simulationsmethodik zu Assistenzsystemen, welche Gegenmaßnahmen bei drohenden Kollisionen ergreifen. Dabei beziehen sie sich auf Kollisionen, die auf ein Abkommen von

der Fahrbahn und weniger auf zwei Fahrzeuge, die aufeinandertreffen könnten. Die Simulation zielt hier auf die potentielle Vermeidung von Run-Off-Road Situationen, in dem sie zunächst eine normale Fahrt betrachtet, um verschiedene Charakteristiken von Fahrern und Beschaffenheit der Fahrbahn zu erzeugen. Stufe 3 umfasst sodann die gleiche Fahrt mit aktiviertem System. [MPH+97]

Die Arbeit von Nentwig et al. handelt von der Hardware-in-the-Loop Simulation für kamerabasierte Assistenzsysteme und wie diese simulativ getestet werden können. Dazu wird beispielsweise die reale Fahrt als Simulationsszenario rekonstruiert, um einen Vergleich der Objektbildungsfähigkeit und Erkennungsleistung der Kamera zu testen und auf welche Weise methodisch eine Auswertung erfolgen kann. Auch die Automatisierung von Testabläufen wird in einer allgemeinen Form dargestellt und inwiefern sich eine Anwendung der HIL Methodik von Nentwig auf ADAS allgemein beziehen kann. [NS10, NS11, NSM11, NMS12]

Schmidt et al. stellen in ihrem Beitrag das Produkt der Fa. IPG Automotive GmbH vor und zeigen auf, an welcher Stelle im Entwicklungsprozess eine simulationsorientierte Validierung von ECU Software erfolgen kann. Dazu zeigen sie Herausforderungen dieser Fragestellung auf und wie virtuell solche Tests durchgeführt werden können. Dazu nutzen sie virtuelle Steuergerät als Basis, damit die Performance des Codes mit den dazugehörigen Betriebssystemen gelingen kann. Die Integration dieser Funktionalität in CarMaker als Produkt von IPG wird ebenfalls vorgestellt. [SHWK15]

Der Inhalt der Arbeit von Bock et al. beschreibt die Methodenentwicklung zur Verknüpfung von realen Prüfgeländefahrten mit virtuell eingeblendeten Objekten über eine VR Brille. Der Beitrag beschreibt den technischen Aufbau des Fahrzeugs und die Methode zur Platzierung von Fahrzeugen im Sichtfeld des Fahrers. Diese Simulationsmethode wird hier als vehicle-in-the-loop bezeichnet. Der Beitrag schließt mit der Validierung von mehreren Test Hypothesen. [BMF07]

4.6.2 RQ-2: Toleranzuntersuchung und ihre Bewertung

RQ-1 zielte zunächst auf eine allgemeinere Einordnung von Ansätzen ab, wozu auch entsprechende Kriterien herausgearbeitet wurden. In RQ-2 sind die Kriterien jedoch spezifischer auf den Ansatz bezogen, um die Unterschiede herauszustellen. Daher wird im Folgenden keine Clusterung der Ansätze vorgenommen, sondern die jeweiligen Beiträge in absteigender Reihenfolge der Punktebewertung für RQ-2 inhaltlich dargestellt.

#0172: Das Simulationsframework von Noth et al. eignet sich aufgrund der Plug-In Konzeption und der beschriebenen Software-Architektur sowohl für die Einbindung von einzelnen Kompo-

nenten eines Aktiven Sicherheitssystems als auch für den Einsatz als Ganzes. Durch das Random Test Plug-In, eine Art von Toleranzmodellierung, die neben der zufallsbasierten Variation auch über einen Triangle Variator verfügt und innerhalb eines Intervals einen Parameter linear bis zu einem Maximum ansteigen lässt und wieder absenkt. Allerdings ist die Änderungszeit auch hier zufallsbasiert. Durch die Beschreibung des Frameworks und die exemplarische Anwendung in drei Beispielen mit Fahrer, Software- und Hardwarekomponenten verfügt der Beitrag über eine hinreichende Methodenbeschreibung. [NEI12]

#0061: Der VeHIL Projekt von Gietelink et al. verfügt aufgrund der Einbindung eines realen Fahrzeugs auf einem Rollenprüfstand sowohl über die Möglichkeit, ein ganzes System, welches im Fahrzeug verbaut ist, als auch nur einzelne Komponenten zu testen. Eine methodische Beschreibung ist ebenfalls inkl. Einbettung in die vorhandene und bekannte Entwicklung angefügt. [GPSV09]

#0192: Der Ansatz von Garate et al. konzentriert sich sowohl auf einzelne Komponenten als auch auf ein ganzes System für ein Aktives Sicherheitssystem, welches auf Fußgänger und Radfahrer bremst. Weiterhin wird eine Methodik präsentiert, welche sich auf die numerische Darstellung eines Aktiven Sicherheitssystems in Verbindung mit der PreScan Software spezialisiert hat. Auch werden entsprechende Testszenarien vorgestellt, welche sich jedoch nicht auf eine systematische Variation beziehen, sondern vielmehr auf einen Einzelversuch. Daher wird weder auf eine Toleranzmodellierung der Szenarien noch auf eine automatisierte, systematische Variation eingegangen. [GBK12]

#0197: Im Rahmen des Beitrags von Choi et al. wird der Virtual Test Track präsentiert, welcher sowohl für einzelne Komponenten als auch für ganze Systeme geeignet ist. Eine methodische Beschreibung ist ebenfalls Bestandteil der Arbeit, wobei der Kern des Beitrags auf der Funktionsentwicklung liegt. [CYSK14]

#0262: Der Beitrag von Chien et al. beschreibt grundsätzlich eine Methode zur Bewertung von Aktiven Sicherheitssystemen in entsprechenden Testszenarien unabhängig von der Methodik, wie die entsprechenden Fahr- und Systemdaten entstanden sind. Im vorliegenden Fall handelte es sich um real eingefahrene Daten aus einem Versuchsträger. Daher ist die Methodik prinzipiell sowohl zur Bewertung auf einzelne Komponenten als auch vollständige Systeme anwendbar; eine detaillierte Erläuterung der Methodik ist in dem Beitrag inkludiert. [CLA+12]

#0330: Die Arbeit von Zellner et al., welche eine Erweiterung der Honda-DRI Safety Impact Methodik umfasst, verfügt sowohl über die Möglichkeit, einzelne Komponenten eines Systems als auch dieses als Ganzes auf ihren Sicherheitsbeitrag zu untersuchen. Die äußerst detailliert beschriebene Methodik mit ihren jeweiligen Bausteinen und Systemkomponenten werden in

mehreren Veröffentlichungen behandelt, die über die gleiche Bewertung verfügen. Insgesamt besteht die Datengrundlage für die Simulationsszenarien aus real aufgetretenen Unfällen, die zu einem gewissen Detailgrad dokumentiert wurden. Dies entspricht jedoch nicht einer Toleranzbetrachtung von Consumer Tests oder einer Parametervariation, so dass hierfür keine Punkte vergeben werden konnten. Die Methodik umfasst eine Reihe von Simulationsmethoden, die sich sowohl auf den Fahrer als auch auf reiner Berechnung berufen ohne über menschlichen Input zu verfügen. [ZAS+12]

#0332: Entspricht dem Beitrag von **#330** und fügt keinen weiteren Erkenntnisgewinn hinzu. [AZS+11]

#0333: Entspricht dem Beitrag wie **#330** und fügt keinen weiteren Erkenntnisgewinn hinzu.. [ZAC+09]

#0141: In dieser Veröffentlichung wird die Kopplung der VEHIL Methodik aus **#0061** mit einer Umfeld-Simulation vorgestellt, bei welcher es sich um PreScan handelt. Auch hier erlaubt die Verwendung eines realen Fahrzeugs sowohl Einzel- als auch Komponentenverbünde zu testen. Die Toleranzbetrachtung und Parametervariation sind nicht in dem Beitrag behandelt. [HTP+10]

#0145: Im Beitrag von Han et al. wird der Virtual Test Track als Grundlage für eine Untersuchung eines integrierten ACC Systems genutzt. Die methodische Beschreibung zum Simulator steht dabei im Vordergrund; hier wird verdeutlicht, dass der Simulator im Rahmen von funktionalen Einzeltests einerseits Komponenten und andererseits auch vollumfängliche Systeme einbinden kann. [HyY06]

#0170: Schwerpunktmäßig behandelt Nentwig in seinem Beitrag die Reproduktion von realen Fahrszenarien, die mit einer Monokamera eines ADAS erstellt wurden. Dabei spielt die Komponente Kamera eine Hauptrolle, die hier ohne nachgelagerte Funktion bewertet wird. Prinzipiell lässt sich ein vollständiges System basierend auf einem Kamerasensor in die Simulationsumgebung integrieren. Wie eine solche Reproduktion erfolgt und wie die Auswertung durchgeführt wird, ist Bestandteil der methodischen Beschreibung. [NS10]

#0171: Dieser Beitrag ist eine inhaltliche Anlehnung an [HyY06] unter **#0145** und beschreibt hier erneut eine human-in-the-loop Studie für ein ACC System mit Hilfe der Virtual Test Track Infrastruktur. [HY06]

#0173: Der Ansatz von Bock sieht vor, Aktive Sicherheitssysteme innerhalb eines realen Fahrzeugs zu simulieren, welches sich in einem abgesperrten Bereich frei bewegen kann. Die Interaktion mit anderen Fahrzeugen oder Verkehrsteilnehmern erfolgt auf virtueller Ebene, indem die Daten dafür synthetisch erzeugt und in das simulierte oder auch real verbaute System eingespeist

werden. Das Ergebnis dieser Daten wird dem Fahrer mittels VR Brille ins Sichtfeld projiziert. Es ist ebenfalls möglich, die Aktuatorik einzubinden, so dass sich evtl. Bremsmanöver direkt und gefahrloser durchführen lassen als mit realen Fahrzeugen. [BMF07]

#0174: Im Beitrag von McMillan et al. wird zur Bewertung von Systemen von Gegenmaßnahmen wie z.B. einen Spurhalteassistenten ein drei-stufige Methodik inkl. Monte-Carlo Simulation verwendet. Dabei wird ein vollständiges System modelliert, welches in dem Beitrag nicht modular dargestellt wird. Aufgrund der Monte-Carlo Simulation können diverse Parameter in dem Modell variiert werden. [MPH+97]

#0187: Der Beitrag von Stapleford et al. beschreibt den von Honda und DRI entwickelten full-scale Fahrsimulator, der auch den Reports **#0330**, **#0332** und **#0333** als Grundlage gedient hat. Hier wird jedoch nur die Verwendung einer einzelnen Komponente wie z.B. eines Steuergeräts beschrieben, welches in eine Infrastruktur aus mehreren mathematischen Modellen für Fahrdynamik oder Sensorik eingebettet ist. Die Toleranzmodellierung wird hier allerdings nur indirekt verdeutlicht. So wird darauf verwiesen, dass eine Reihe von NTHSA Aktivitäten auf die Entwicklung von Methodiken abzielt, für die Positionen und Fahrwege von eigenen und anderen Fahrzeugen relevant sind, die im Rahmen einer Simulatorstudie erprobt werden können. Speziell werden Systeme zur Kollisionsvermeidung neben anderen in den Fokus genommen. Letztlich ist es nur eine Vorstufe einer Toleranzmodellierung, jedoch ist sie hier konkreter formuliert als in den übrigen Beiträgen zu dieser RQ. [SWB+97]

#0245: Nentwig et al. präsentieren in ihrem Beitrag eine HiL Methode, welche sowohl für eine einzelne Komponente als auch für ein komplettes System gedacht ist. Hierzu werden entsprechende Fahrzeugmodelle, Fahrermodelle, Sensormodelle, Aktormodelle und Umgebungsmodelle miteinander verknüpft. Die Methodik beschreibt zudem, wie über eine Szenariengenerierung die Auswertung eines Spurerkennungsalgorithmus' bzw. Fahrzeugerkennungsalgorithmus' erfolgen kann. [NS11]

#0291: Der Beitrag von Chien ist hier eine Erweiterung zu dem bereits unter **#262** genannten Beitrag und ohne weitere, signifikante Ergänzungen im Sinne der Forschungsfrage. [CLC13]

#0344: Dieser Beitrag ergänzt den Beitrag aus **#0245** um die weitere methodische Beschreibung, wie im allgemeinen die HiL Methode bei Nentwig et al. zum Einsatz kommt und welche Erweiterungen dort vorgenommen worden sind, wie z.B. Messaufzeichnung, manuelles Testen und automatisiertes Testen. Die Anwendbarkeit auf einzelne Komponenten oder Systeme ist weiterhin gewährleistet. [NSM11]

#0345: Tideman et al. erläutert in ihrem Beitrag die Verknüpfung von der Umgebungssimulation PreScan mit dem auf Matlab Simulink basierenden Ergänzungstool Automotive Simulation

Models von dSPACE, welches umfassende Fahrdynamikmodelle und die Anbindung von realer Hardware erlaubt. Durch die integrierten Modelle in PreScan zur Aktorik ist somit eine Untersuchung von einzelnen Komponenten und vollständigen Fahrerassistenzsystemen möglich. [TBL+12]

#0346: Schmidt et al. zeigen in ihrem Beitrag auf, wie die Funktionalität von virtuellen Steuergeräten, integriert in die IPG Simulationsumgebung "CarMaker´´, dazu dient, sowohl Einzelkomponenten wie Algorithmen als auch vollständige Fahrerassistenzsysteme, bis hin zu autonom agierenden Systemen getestet werden können. [SHWK15]

#0374: In diesem Beitrag wird eine Weiterentwicklung von PreScan durch Tideman erläutert. Dies umfasst grundsätzlich die Erweiterung um beispielsweise die Anbindung von veDyna und aktiven Lichtmodellen. Im Wesen erlaubt PreScan jedoch auch weiterhin sowohl Einzelkomponenten als auch Systeme simulativ zu testen. [Tid10]

#0393: Zhang et al. beschreiben eine Test Methode zur Bestimmung der Performance eines Spurwarnassistenten. In Verbindung mit TESIS DYNA4, einer Fahrdynamiksimulation und einem Echtzeitcontroller, wurde eine Kamera inkl. Steuergerät als HiL realisiert. Durch die Closed-Loop Simulation ist das Testbett sowohl für einen Komponententest als auch zu einer vollständigen Systemabbildung geeignet. Eine Parametervariation oder eine Toleranzmodellierung erfolgt hier nicht. [ZCLL14]

#0371: Helmer et al. erläutern in ihrem Ansatz, welchen Benefit die Nutzung von empirischen Unfalldaten und die Monte-Carlo Simulation von Aktiven Sicherheitssystemen für den Fußgängerschutz haben können. Dabei wird das System als Ganzes modelliert, die Modularisierung mit dem Test einzelner Komponenten ist nicht vorgesehen. Eine Variation der Parameter erfolgt hier zufallsbasiert, so dass das Kriterium der Toleranzmodellierung nicht erfüllt ist. [HSS+11]

#0110: Der Beitrag von Tideman et al. befasst sich mit dem Entwicklungsprozess eines Fahrerunterstützungssystems und damit wie und von wem die Anforderungen dafür zusammengetragen werden. Ein Prototyp des Systems wird dann in einem Fahrsimulator durch entsprechende Probanden bewertet. Daher ist das System als Ganzes im Fokus, lässt sich jedoch theoretisch auch auf eine einzelne Komponente anwenden. Wie allerdings ein simulativer Test methodisch aufgebaut ist, bleibt hier offen. [TVA10]

#0169: Gruyper et al. illustrieren in ihrem Beitrag eine Simulationsarchitektur für Kooperative Systeme, welche über eine Sensor Simulationsplattform SiViC und der Prototyping Plattform RTMaps realisiert werden. Der Fokus liegt dabei auf der Simulation von mehreren Fahrzeugen und deren Kommunikation. Daher stehen hier Gesamtsysteme deutlicher im Fokus als Einzelkomponenten. Ein methodischer Aufbau wird beschrieben, wobei eine Parametervariation oder

eine Toleranzmodellierung noch nicht berücksichtigt ist. [GDN+12]

#0368: In der methodischen Beschreibung zur integrierten Bewertung von Aktiven Fußgängerschutzsystemen erläutern Raudszus et al. die Kombination von passiven und aktiven Anteilen an einem Fußgängerschutz und wie diese in ihrer Zusammenwirkung einen Effekt auf den Unfallschutz haben. Dabei wird ein ganzheitlicher Ansatz gewählt, so dass die Aktiven Systeme in dem Beitrag als ein System modelliert werden, was im Wesentlichen einer Geschwindigkeitsreduktion entspricht und dadurch einen Einfluss auf die Anprallkinematik hat. Daher sind Einzelkomponenten nicht im Vordergrund der Simulationsbestrebungen. [RHZE15]

#0083: Der Beitrag von Lee et al. erläutert im wesentlichen die Entwicklung eines Algorithmus' zur Spurhaltung eines Fahrzeugs. Dabei werden während der Evaluation mehrere Tests und unter anderem auch simulative Tests durchgeführt. Speziell in diesem Beitrag wird eine methodische Beschreibung dieser Simulationen nicht gegeben, auch nicht bezogen auf die verwendete Architektur. Daher liegt der Fokus dieses Beitrags deutlich auf einer Einzelkomponente. [LCY+14]

4.6.3 RQ-3: Methodische Entwicklung von Simulationsinfrastruktur

Die inhaltliche Zusammenfassung der Beiträge zur Forschungsfrage RQ-3 erfolgt hier auf der Grundlage der erreichten Punktzahl in absteigender Reihenfolge. Bei Punktgleichheit wird in der aufsteigenden Reihenfolge der Study-ID fortgesetzt.

#0330: Der Ansatz zum Honda-DRI Safety Impact Methodology umfasst mehrere Subsysteme. So verfügt das Framework über eine Szenariengenerierung, welche aus hinterlegten Unfalldaten eine Crashsituation für eine Wiederholung der Szene in einer Simulation rekonstruiert. Im zweiten Modul werden die Daten um weitere Technik relevante Spezifikationen ergänzt und nach repräsentativen Maßstäben ausgesucht. Dort wird entschieden, welche Art von Tests, z.B. Realtests mit Fahrer oder Full Scale Simulator Tests ohne Fahrer vorgenommen werden sollen. Daraus ergibt sich dann die Parametrierung der konkreten Testszenarien. Die Modellierung des Aktiven Sicherheitssystems erfolgt im Simulationsmodul, welches als "Head-on Crash Avoidance Assist System (H-CAAS)" illustriert wurde. In Modul vier werden daraufhin die Ergebnisse aus der Simulation der Unfälle mit und ohne Gegenmaßnahmen bestimmt. Trotz der detaillierten Beschreibung der Methodik und des Frameworks wird kein Bezug zu einer übergeordneten Methode beschrieben, wie bei der Konzeption des Frameworks vorgegangen wurde. Dieser Umstand verhindert eine Bestbewertung in dieser Studie. [ZAS+12]

#0332: Dieser Beitrag ist inhaltlich ähnlich wie **#0330** aufgebaut. Hier wird jedoch detaillierter auf das Submodul "Objective Tests" eingegangen, welches die Testumgebung mit den aus-

gewählten Testszenarien zusammenbringt und erläutert, wann welche Art ausgewählt werden sollte. [AZS+11]

#0333: In diesem Beitrag wird von einer Fallstudie zu einem prototypischen Notbremssystem berichtet. Er ging den anderen beiden Beiträgen zeitlich voraus und behandelt neben der Beschreibung zum Aktiven Sicherheitssystems auch die Auswahl geeigneter Testkriterien bezogen auf die unterschiedlichen Testarten. Auch hier wird die gleiche Punktzahl wie in **#0330** und **#332** erreicht. [ZAC+09]

#0172: In der Vorstellung des Frameworks von Noth et al. wird zunächst kein konkretes Aktives Sicherheitssystem modelliert. Vielmehr werden die einzelnen Softwarekomponenten der Gesamtarchitektur wie z.B. das Traffic Modul, die weiteren Kernelemente mit Straßen- Fahrzeug- und Rendering-Elementen und andere erläutert. Zudem werden auch weitere Plug-Ins beschrieben, wie beispielsweise eines für CAN Bus, ein Umgebungs-Plug-In oder auch das Random Test Plug-In. Letzteres erlaubt die zufällige Variation von Testbedingungen. Die Regeln der Variation können dann ebenfalls festgelegt werden; hier stehen ein Triangle, ein Step und ein Toggle Variator zur Verfügung. Durch das Write und das Plot-Plug-In können die aufgezeichneten Daten dann auch ausgewertet werden, wobei die inhaltliche Bewertung der Daten dort mangels modellierten Sicherheitssystems nicht berührt wird. Die kurze Nennung von Anforderungen an die Software Architektur wird hier als Meta-Methodik gewertet, da es eine im Vorfeld der Erstellung grundsätzliche Überlegung zur Konzeption darstellt. Dabei wird auf den Aufbau des Frameworks eingegangen, um die Software zuverlässig im Sinne der Zweckerfüllung zu konzipieren. Abgesehen von den üblichen Motivatoren zur Nutzung von Simulationen wie Reproduzierbarkeit, Kostenersparnis, Testtiefe und Gefahrenreduktion ist jedoch nicht erkennbar, welches (Simulations-)ergebnis am Ende erzielt werden soll. [NEI12]

#0174: In der statistikbasierenden Simulationsmethode von McMillan et al. wird ein Aktives Sicherheitssystem zum Spurhalten verwendet, welches als ein abgeschlossenes System dargestellt wird. Die Szenariengenerierung erfolgt über die Erzeugung von Inputvektoren, welche mittels des Latin hypercube Samplings als eine Monte-Carlo-Simulation ermittelt werden. Die Auswertung erfolgt dann über die Berechnung der Wahrscheinlichkeit zur Verhinderung eines Spurverlassens und der Optimierung des Sicherheitssystems. [MPH+97]

#0245: Nentwig et al. befassen sich in ihrem Beitrag in erster Linie mit dem simulativen Test von Kameras, welche eine Komponente für Aktive Sicherheitssysteme darstellen können. Daher fehlt zunächst der Teil eines Sicherheitssystems, der die verarbeitenden Daten aus der Kamera für weitere Funktionen wie z.B. zur Bewertung oder Fahrzeugbeeinflussung nutzt. Dennoch wird in diesem Ansatz eine Szenariogenerierung vorgestellt, welche aus vorhandenen realen

Videoaufnahmen eine virtuelle Szene in der Simulationsumgebung erzeugt. Der Fokus liegt hier jedoch auf optische Effekte, um die Bildverarbeitung der Kamera entsprechend zu testen. Die Auswertung ist sehr ausführlich beschrieben und umfasst letztlich die Validierung, da auf reale Daten und Ergebnisse vergleichsweise zurückgegriffen wird. Eine grundlegende Meta-Methodik wird hier nicht aufgeführt. [NS11]

#0345: Der Beitrag von Tideman et al. zeigt die Integration eines Spurhalteassistenten in die PreScan-dSPACE-Matlab/Simulink Testumgebung. Dabei werden eine Reihe von Anforderungen allgemeiner Natur an eine Simulationsumgebung formuliert, so dass auch hier als eine übergeordnete Herangehensweise zur Realisierung und damit als eine Meta-Methodik betrachtet wird. Die Szenariengenerierung wird auch hier eine Parametervariation adressiert, jedoch wird nicht ersichtlich, ob die Variation der Parameter zufallsbasiert oder systematisch erfolgt. Eine Auswertemethodik oder ein Validierungsprozess werden nicht weiter eingebracht. [TBL+12]

#0368: Bei Raudszus et al. wird im Rahmen der Methodik ein Aktives Sicherheitssystem zur Geschwindigkeitsreduktion verwendet. In diesem Beitrag ist nicht ersichtlich, inwiefern es sich dabei um ein modelliertes System, z.B. durch eine Funktion oder eine Matrix, handelt. Eine Szenarienerstellung ist Bestandteil der Methodik, jedoch wird hier nicht klar formuliert, ob es sich um eine automatisierte Szenariengenerierung oder um eine manuell erstellte Variante handelt. Die Auswertemethodik ist umfangreich und erstreckt sich sowohl auf die aktive als auch auf die passive Phase. Damit soll eine integrierte Bewertung von möglichen Unfällen mit Fußgängern ermöglicht, die Effektivität verschiedener Maßnahmen am Fahrzeug bewertet und die Auswirkungen auf die potentiellen Verletzungen des Fußgängers ermittelt werden können. [RHZE15]

#0061: Im Beitrag von Gietelink et al. wird sowohl ein Aktives Sicherheitssystem modelliert, welches die sichere Geschwindigkeit und den Abstand zu einem vorausfahrenden Fahrzeug dem Fahrer anzeigen soll, als auch ein Validierungsprozess angedeutet, welcher repräsentativ für den VeHIL beschrieben wird. Außerdem wird auch auf die Validierung von Fahrzeugfunktionen mit Hilfe des VeHIL eingegangen. Die anderen Kriterien zu dieser Forschungsfrage werden nicht erfüllt. [GPSV09]

#0083: Der Beitrag von Lee et al. verwendet zur Entwicklung eines Spurhalteassistenten einen Fahrsimulator, welcher dann in weiteren aufgeführten Beiträgen genannt wird. Neben der ausführlichen Darstellung der Algorithmenentwicklung wird auf den Aufbau des Fahrsimulators VTT selbst nur knapp eingegangen. Eine Fallstudie zwischen Simulation und Realität wird im weiteren Verlauf des Beitrags detailliert dargestellt. Jedoch wird eine spezielle Auswertungsmethodik oder auch eine übergeordnete Methodik nicht beschrieben. [LCY+14]

#0110: Der Beitrag zum szenariobasierten Ansatz zur Entwicklung eines Fahrerinformationssys-

tems von Tideman et al. gibt einen Überblick über das Vorgehen der Entwicklung mit Hilfe von mehreren Probanden und einem Fahrsimulator. Dabei steht im Fokus, die verschiedenen Meinungen der Probanden einzubeziehen und ihnen so plausibel zu machen, welche Konsequenzen verschiedene Änderungen auf das System haben. Gleichzeitig erhalten die Entwickler die Möglichkeit, auf die Kundenanforderungen noch direkter eine Rückmeldung zu erhalten und zudem die Reaktion auf die Umsetzung der Anforderungen zu bekommen. Dabei wurde ein Szenariengenerator verwendet, der in diesem Beitrag jedoch nicht näher technisch spezifiziert wurde. Als Fallstudie wurde ein Informationssystem für den Spurwechsel entwickelt. [Tid10]

#0141: Die Verknüpfung von PreSCAN und VeHIL wird hier von Hendriks et al. vorgestellt. Darin wird ein konkretes Aktives Sicherheitssystem modelliert und auf eine Art Szenariengenerierung eingegangen und die Wichtigkeit der Modellierung hervorgehoben. Allerdings wird keine konkrete Lösung zur verbesserten Generierung vorgeschlagen. Eine Auswertungsmethodik, ein konkreter Validierungsprozess oder eine Meta-Methodik finden keine Erwähnung in diesem Beitrag. [HTP+10]

#0145: Han et al. modellieren in ihrem Beitrag ein ACC System mit einer integrierten Kollisionsvermeidung, um simulativ eine Bewertung solcher Systeme vorzunehmen. Gleichzeitig stellen sie den Virtual Test Track vor, dessen Fahrzeugmodell einer Validierung unterzogen wurde. Hinweise oder Beschreibungen einer Szenariengenerierung oder einer konkreten Auswertungsmethodik konnten nicht entnommen werden. Eine Meta-Methodik wurde nicht konkret erläutert. [HyY06]

#0169: In dem Beitrag von Gruyer et al. wird neben einer Modellierung des Aktiven Sicherheitssystems, hier in Form eines kooperierenden Kollisionswarnsystems, auch auf die mehrfache Ausführung von Szenarien mit unterschiedlichen Parametern wie z.B. der Ausstattungsrate von Kommunikationskanälen der Fahrzeuge eingegangen. Weitere Kriterien werden hingegen nicht erfüllt. [GDN+12]

#0170: Nentwig et al. präsentieren in ihrem Artikel neben einer Szenariengenerierung aus realen Fahrten auch einen Validierungsprozess zum Vergleich zwischen realen und simulierten Kameraaufnahmen. Der Fokus bei der Szenariengenerierung liegt jedoch vorwiegend in der Reproduktion der Aufnahmen und anschließend Abgleiche für die Bildverarbeitungsalgorithmen durchführen zu können. Weitere Kriterien werden nicht behandelt. [NS10]

#0171: In dem weiteren Beitrag von Donghoon et al. wird erneut ein ACC modelliert und auch das Bewegungsmodell wird detailliert validiert. Durch die Schwerpunktbildung auf diese Aspekte zur Simulation werden keine weiteren Kriterien erfüllt. Außerdem orientiert sich das Papier sehr nah an **#0145**. [HY06]

#0173: Bock et al. stellen im Rahmen ihres Ansatzes zur VIL Methode mit Augmented Reality eine Auswertungsmethodik vor, wie mit Hilfe eines realen Fahrzeugs und realen Fahrten zusätzlich ein Aktives Sicherheitssystem durch virtuelle Einblendung und Datenergänzung analysiert werden kann. Zudem gehen sie auf eine Validierung der Methode VIL ein und beschreiben en detail, wie diese für den Ansatz vollzogen wird. Dazu werden Hypothesen aufgestellt und zwischen Realität und Simulation auf Signifikanz überprüft. [BMF07]

#0262: Im Beitrag von Chien et al. wird der Fokus auf eine Auswertungsmethodik gelegt, zu welcher eine Szenarienauswahl und -zusammenstellung als eine Art der Generierung zählt; allerdings ist diese hier nicht automatisiert. Aufgrund des Bezugs zu realen Fahrten auf Prüfgeländen erfolgt keine Modellierung eines virtuellen Systems direkt. Die weiteren Kriterien zur Meta-Methodik und zum Validierungsprozess können hier nicht registriert werden. [CLA+12]

#0291: Siehe zuvor **#0262**. [CLC13]

#0344: Dieser Beitrag von Nentwig bezieht sich ebenfalls auf die HIL Instrastruktur, die bereits hier genannt worden sind. Aufgrund des allgemeineren Charakters des Papiers wird eine etwas niedrigere Punktzahl erzielt. So wird beispielsweise ein Aktives Sicherheitssystem modelliert und zudem eine Auswertemethodik mit Automatisierung vorgestellt. Allerdings fehlt der konkrete Anwendungsfall in diesem Beitrag. Die übrigen Kritieren werden hier getroffen. [NSM11]

#0371: Helmer et al. beschreiben im Rahmen ihrer Unfallanalyse zwischen Fußgängern und Fahrzeugen eine statistische Auswertungsmethodik. Dabei erläutern sie, auf welche Weise Zielvariablen und beschreibende Variablen definiert und fehlende Daten in der Datengrundlage ausgeglichen werden. Durch die uni- und multivariate Analyse und die ausführliche Diskussion der Ergebnisse resultiert eine hinreichende Validierung, so dass das Kriterium eines Validierungsprozesses erfüllt ist. [HSS+11]

#0393: In dem Beitrag von Zhang et al. wird ein LDW System modelliert, welches in eine Infrastruktur aus PreSCAN und MatLAB mit einer Fahrdynamik veDYNA von TESIS integriert ist. Dabei wird die sort tree Methode zur Entwicklung der Szenarien eingesetzt, um eine effiziente Abdeckung der Anforderungen an das System in möglichst wenigen Testfällen abzudecken. Es wird hier jedoch nicht ersichtlich, inwiefern diese automatisiert ist. Weitere Kriterien werden hier nicht erfüllt. [ZCLL14]

#0187: Der Beitrag zum DRI Driving Simulator und den Autoren um Staplefort et al. erläutert die Verwendungsmöglichkeit und den Aufbau des Simulators und modellieren u.a. ein ACC System und weitere Assistenzsysteme. Dabei wird nicht auf eine Szenariengenerierung oder Auswertemethodik eingegangen. Auch eine Meta-Methodik wird nicht näher erwähnt. [SWB+97]

#0192: Der Beitrag von Garate et al. beschreibt ausführlich eine Methode zur Modellierung eines aktiven Sicherheitssystems zum Schutz von Fußgängern. Um ein proof-of-concept zu realisieren, wurde das System auf der Simulationsebene mit Hilfe von PreScan und Matlab getestet. Inhaltlich konzentriert sich der Beitrag vor allem auf die Erläuterung der einzelnen Schritte im Modell und legt wert auf seine Logik. Weitere Kriterien zur Forschungsfrage werden vom Beitrag nicht erfüllt. [GBK12]

#0197: Im Beitrag von Choi et al. wird das entwickelte Aktive Sicherheitssystem in einem hohen Detailgrad beschrieben. Auf den verwendeten Fahrsimulator zur Evaluation des entwickelten Algorithmus' wird nur knapp eingegangen, so dass nur die wichtigsten Komponenten angerissen werden. Der experimentelle Aufbau des Versuchs führt zum Schluss, dass es sich dabei nicht um ein automatisiert generiertes Szenario handelt. Auch eine spezielle Auswertemethodik oder ein Validierungsprozess wird nicht aufgeführt, gleichwohl wird eine Studie mit entsprechenden Ergebnissen präsentiert, welche die grundsätzliche Funktionsweise des Algorithmen belegen. Eine übergeordnete Methodik wird hier nicht näher erwähnt. [CYSK14]

#0346 Schmidt et al. stellen in ihrem Beitrag virtuelle ECUs und deren Vorteile vor. Durch die Einbettung ins V-Modell als grundlegenden Entwicklungsprozess wird hier eine Meta-Methodik identifiziert. Grundsätzlich wird aber kein konkretes System modelliert, so dass die übrigen Kriterien hier nicht berührt werden. [SHWK15]

#0374: Dieser Beitrag von Tideman et al. beschreibt eine grundsätzliche Modellierungsmöglichkeit von Aktiven Sicherheitssystemen und weiteren Features von PreSCAN, ohne sich jedoch auf ein konkretes Fallbeispiel zu beziehen. Daher werden hier auch keine weiteren Kriterien erfüllt. [TBL+12]

4.7 Validitätsbetrachtungen

Runeson and Höst unterscheiden bei Ihren Ergebnissen unterschiedliche Aspekte zur Bestimmung der Validität. Es wird zwischen construct validity, internal validity, external validity sowie realiability unterschieden. [RH08]

- Construct Validity: Inwiefern dient das entwickelte Vorgehen der Intention des Forscher und was wird untersucht in Bezug auf die Forschungsfragen?
- Internal validity: Inwiefern kann bei kausalen Zusammenhängen ein dritter Faktor das Ergebnis beeinflussen?
- External validity: Inwiefern sind die Ergebnisse für andere Fälle relevant? Inwiefern sind die Ergebnisse verallgemeinerbar?
- Reliability: Inwiefern sind die Ergebnisse abhängig vom durchführenden Forscher. Idealerweise werden die gleichen Ergebnisse erzielt, wenn ein anderer Forscher nach dem gleichen Protokoll handelt. [RH08, S. 153]

Die größte Gefahr hinsichtlich der construct validity bestand darin, dass die Menge an Beiträgen bewusst klein gehalten wird, damit keine potentiellen Beiträge über Consumer Tests in die Studie einbezogen werden können, die einen möglichen Forschungsbedarf identifizieren. Es wurden mehrere Datenbanken herangezogen, um eine möglichst breit angelegte Suchanfrage durchführen zu können. Auch anhand der Gesamtmenge mit 431 Beiträgen, die durch die Eingabe des Suchstring in den Datenbanken gefunden wurden, kann eine inhaltliche Breite der Studie belegt werden.

Dennoch kann an dieser Stelle nicht zu 100% ausgeschlossen werden, dass evtl. vereinzelte Beiträge nicht in der Gesamtmenge aufgeführt sind. Durch die mehrfache Prüfung anhand von Referenzbeiträgen und deren Autoren wurde versucht, das Risiko hierfür zu minimieren.

Das Protokoll zur dieser Literaturrecherche wurde nach der Methode von Kitchenham [KC07] erstellt, so dass das Vorgehen in dieser Studie von der Auswahl der Datenbanken, der Suchstring

Definition, der Kriterien-Definition, der Bewertungskriterien zur Beantwortung der Forschungsfragen transparent ist. Bei der Wahl des Suchstrings wurden einige Iterationen durchlaufen, um einige übliche Quellen zur Thematik als definitive Treffer in der Menge der Ergebnisse aus der Suchanfrage sicherzustellen. Außerdem wurde auch auf die Spreizung hinsichtlich der unterschiedlichen Datenbanken geachtet, so dass die Autoren der Referenzbeiträge auch in anderen Datenbanken mit ihren Beiträgen aufgeführt sind.

Folgende Beiträge und Autoren wurden dabei ausgewählt:

- [NS11]
- [SS12]
- [RBL+09]
- [TBL+12]

Mit Hilfe dieser Beiträge und ihrer Autoren konnte dann ein Suchstring ausgewählt werden, der wiederum zu weiteren Beiträgen und Artikeln führt, die auch von den o.g. Autoren stammen. Gleichzeitig führte der Suchstring auch andere Autoren und Beiträge aus dem gleichen Themenumfeld auf.

Internal Validity: Im Rahmen der Analyse ist gegen mehrere Faktoren (hier: Bewertungskriterien) geprüft worden, ob dort eine Abhängigkeit besteht. Dies konnte im wesentlichen bei allen drei Forschungsfragen anhand von verschiedenen Kriterien widerlegt werden.

External Validity: Die Forschungsfragen dienen letztlich dazu, Forschungsbedarf darüber zu identifizieren, an welcher Stelle noch Lücken im Rahmen von Simulationsmethoden bei Aktiven Sicherheitssystemen existieren. Die Aussagen sind insofern verallgemeinerbar, als dass die weitere Forschung in dem Bereich dort weiter fortgeführt werden kann, um die vorhandenen Lücken in der Methodik zur Simulation von Consumer Tests zu schließen.

Reliability: Durch die ausführliche Darstellung des Studienprotokolls, die detaillierte Definition der Ausschluss und Einschlusskriterien, sowie die Erläuterungen zu den Bewertungskriterien der einzelnen Forschungsfragen ist eine hohe Zuverlässigkeit in der erneuten Durchführung dieser Studie durch Dritte gegeben. Dies unterstreicht u.a. auch die Untersuchung anhand des Kappa-Kriteriums im Rahmen der Ein- und Ausschlusswahl der einzelnen Beiträge.

Hierzu wurden aus der Gesamtmenge (431 Beiträge) eine Stichprobe von 50 Beiträgen ausgewählt, wobei die Beiträge durch einen Zufallsgenerator bestimmt wurden.

Laufende Nr.	1	2	3	4	5	6
Zufallszahl	20	243	213	398	258	332
Reference ID	[IAY13]	[MSCK13]	[DT10a]	[RAB+12]	[JFWA13]	[AZS+11]
Laufende Nr.	7	8	9	10	11	12
Zufallszahl	268	331	378	42	361	154
Reference ID	[DHAT14]	[NWJ+11]	[unk]	[BCDG15]	[HE11]	[AOB12]
Laufende Nr.	13	14	15	16	17	18
Zufallszahl	105	385	359	276	290	321
Reference ID	[SBHK14]	[SPM13]	[JLC+13]	[KZR13]	[CSD+12]	[LW11]
Laufende Nr.	19	20	21	22	23	24
Zufallszahl	184	422	128	87	12	384
Reference ID	[PGN+10]	[Jen09]	[YSLS14]	[MS15]	[CS08]	[KS97]
Laufende Nr.	25	26	27	28	29	30
Zufallszahl	147	343	144	45	121	245
Reference ID	[ABB+10]	[VAK10]	[DT10b]	[BTR+14]	[WPH+14]	[NS11]
Laufende Nr.	31	32	33	34	35	36
Zufallszahl	102	214	249	164	179	318
Reference ID	[SA07]	[TSGZ06]	[CWY13]	[MKS13]	[TFT+12]	[TGW+05]
Laufende Nr.	37	38	39	40	41	42
Zufallszahl	266	40	165	163	170	322
Reference ID	[ACB12]	[BBW09]	[PPA+10]	[SR10]	[NS10]	[ST13]
Laufende Nr.	43	44	45	46	47	48
Zufallszahl	241	254	309	216	27	357
Reference ID	[YYW+10]	[BE06]	[TRPP04]	[XMKS07]	[RSMT14]	[SLL12b]
Laufende Nr.	49	50				
Zufallszahl	16	2				
Reference ID	[GLB+13]	[ABRM13]				

Tabelle 4.14: Zufällige Auswahl von 50 Beiträgen

Insgesamt wurde dann durch eine 2. Person eine gleiche Analyse hinsichtlich der Ein- und Ausschlusskriterien durchgeführt. Die nachfolgende Tabelle mit der jeweiligen Anzahl an Ja und Nein-Antworten beschreibt, ob diese Beiträge zur weiteren Bewertung herangezogen werden oder nicht.

Autor/ Kontrolle	ja	nein	Summe
ja	5	2	7
nein	0	43	43
Summe	5	45	50

Tabelle 4.15: Kappa-Kriterium: Übereinstimmung von Ja und Nein-Antworten.

Danach wurde anhand Cohens Kappa-Kriterium der Grad der Übereinstimmung bestimmt. Nachfolgend sei dieses im Überblick erläutert; für eine detaillierte Beschreibung sei auf [KPS14] verwiesen.

$$\kappa_{Cohen} = \frac{p_0 - p_e}{1 - p_e}, \tag{4.1}$$

wobei p_0 die Wahrscheinlichkeit für die übereinstimmenden Antworten h_{ii} beschreibt:

$$p_0 = \frac{\sum_{i=1}^{z} h_{ii}}{N}, \tag{4.2}$$

wobei z die Anzahl der Antwortmöglichkeiten (hier: Ja bzw. Nein) und N alle Elemente der Stichprobe darstellt. Weiterhin beschreibt p_e die Wahrscheinlichkeit der nicht übereinstimmenden Antworten:

$$p_e = \frac{1}{N^2} * \sum_{i=1}^{z} h_{i.} * h_{.i}, \tag{4.3}$$

wobei $h_{.i}$ die Randhäufigkeit für jede Spalte und $h_{.i}$ die Randhäufigkeit für jede Zeile entspricht. Im vorliegenden Fall konnte ein $\kappa_{Cohen} = 0,811321$ erzielt werden, so dass für die Einordnung anhand der Ein- und Ausschlusskriterien eine hinreichende Übereinstimmung abgeleitet wird.

4.8 Schlussfolgerung

Mit dieser Studie konnte aufgezeigt werden, dass die Simulation von Fahrerassistenzsystemen und die Entwicklung geeigneter Methoden und Techniken einen wichtigen Forschungsbereich interdisziplinär darstellt. Gleichzeitig ist aber durch diese Studie deutlich geworden, dass sich viele Beiträge inhaltlich ähneln oder ähnliche Schwerpunkte haben. So spielt die Modellentwicklung den Hauptteil der Forschungsbestrebungen wieder. Dabei wird überwiegend methodisch vorgegangen, allerdings bleiben andere wichtige Bausteine in der Entwicklung von Simulationsumgebungen auf der Strecke, wie bereits [BZBP13] hervorgehoben hat. Daher wurden hier drei Forschungsfragen entwickelt, um die Forschungsarbeiten im Bereich der Consumer Tests zu identifizieren, die einen wichtigen Aspekt in der Automobilvermarktung einnimmt.

Somit kann als Antwort auf [RQ-1]: ***Welche Ansätze und Konzepte zur Simulation von Consumer Tests, insbesondere bezogen auf EuroNCAP und NHTSA, existieren hinsichtlich Aktiver Sicherheit?*** formuliert werden, dass sich momentan nur eine geringe Zahl von Ansätzen und Konzepten schwerpunktmäßig auf Consumer Tests beziehen. Die wenigen Beiträge, die sich auf Consumer Tests spezialisiert haben, betrachten entweder Systeme abseits von Notbremsassistenten bzw. Warnsystemen oder spezielle Bewertungsverfahren zur Erfassung der Leistungsfähigkeit. Dabei sind diese Verfahren dann universell anwendbar, also unabhängig vom realen oder virtuellen Test. Im Gegenzug zeigt sich jedoch, dass Aktive Sicherheitssysteme selbst im industriellen Kontext eine gewichtige Rolle spielen, gleich wenn sich die Beiträge hier auf Spurhalteassistenten und andere Fahrerassistenzsysteme mit Schwerpunkt Sicherheit erstrecken.

Bei der Frage [RQ-2]: ***Sofern [RQ-1] hinreichend durch entsprechende Beiträge beantwortet: Wie werden in diesen Ansätzen und Konzepten Aktive Sicherheitssysteme oder deren Komponenten bezüglich ihrer zulässigen Toleranzen simulativ bewertet?*** wurde aufgrund der geringen Anzahl an Beiträgen mit Consumer Test Schwerpunkt eine Erweiterung durch heranzuziehende Artikel in Erwägung gezogen und daher einer Mindestpunktzahl von 0,6 Punkten ausgewählt. Es konnte festgestellt werden, dass in den Ansätzen mehrheitlich sowohl ein ganzes System als auch einzelne Komponenten beschrieben wurden. Auch wurde meist ein methodisches Vorgehen bei der Entwicklung dieser identifiziert. Allerdings fehlen in diesen Ansätzen und Konzepten überwiegend Bestandteile zu einer Toleranzen-Modellierung, wie z.B. eine Fahrspurmodellierung oder eine automatisierte Variation von Testparametern; stattdessen sind sie auf den Einzelfall als Funktionstest konzentriert. Somit bleibt hier eine methodische Lücke, welche für die Betrachtung von Consumer Test Szenarien jedoch wichtig ist zu schließen. Dadurch wird eine detaillierte Einschätzung am Ende der Entwicklung ermöglicht, die darlegt, wie sich die Software in den vermeintlich marginal unterschiedlichen Szenarien verhält.

Mit Frage [RQ-3]: ***Welche Methodik wurde in diesen Ansätzen und Konzepten verwendet, um zielgerichtet eine solche Simulationsumgebung zu entwickeln, die das Ziel verfolgt, ein Aktives Sicherheitssystem in seinen zulässigen Toleranzen zu bewerten?*** sollte erörtert werden, inwieweit bereits während der Entwicklung des jeweiligen Ansatzes zur Simulation übergeordnete, methodische Überlegungen eingeflossen sind. Dazu wurde zunächst betrachtet, auf welchen Aspekt im Rahmen eines Simulationsansatzes sich der Ansatz bezieht. Konkret wird entweder die Modellierung eines Aktiven Sicherheitssystems, die Szenariengenerierung, die Auswertung oder der Validierungsprozess betrachtet. Im Anschluss wurde geprüft, ob dort eine Meta-Methodik entwickelt wurde, um ähnliche Problemstellungen zielgerichteter zu lösen. Auffällig war, dass sich von den bereits eingegrenzten Papieren viele mit Modellierung des Systems befassen und eine Methodik zur Szenariengenerierung sowie eine Auswertungsmethodik zunehmend weniger in die Ansätze eingeflossen ist. Auch die Nennung eines Validierungsprozesses ist, gleich welcher Art realisiert, zunächst in weniger als der Hälfte der Beiträge ein Bestandteil. Die Frage nach der Verwendung einer Meta-Methodik fällt schließlich in den meisten Fällen negativ aus, so dass auch hier eine methodische Lücke besteht, denn bevor zielgerichtet simulativ ein Erkenntnisgewinn über Modellbildung, Szenariengenerierung und -variation, Auswertung und Validierung der Ergebnisse erzielt werden kann, so umfassend muss im Vorfeld das Ziel und das geplante Vorgehen eindeutig formuliert sein. Eine Handlungsanleitung dazu würde somit entscheidend zum Erfolg beisteuern.

4.9 Zusammenfassung

Im Kapitel 4 wurden drei Forschungsfragen formuliert, deren Beantwortung moderne Ansätze zur Entwicklung von Simulationsumgebungen herausarbeiten sollen. Dazu wurde ein Systematic Literature Review durchgeführt, welche sich an den Leitlinien der Methode von [KC07] orientiert. Die erste Frage konzentrierte sich darauf, welche Ansätze und Konzepte zur Simulation von Aktiver Sicherheit, speziell Consumer Tests wie z.B. EuroNCAP oder den USNCAP existieren. Dabei konnte identifiziert werden, dass sich eine geringe Anzahl der Beiträge damit befasst und meist abseits von Notbrems- bzw. Warnsystemen auf andere Systeme der Aktiven Sicherheit beziehen. Die zweite Frage zielte konkret auf die Art und Weise ab, wie in diesen Beiträgen die Systeme in ihren zulässigen Toleranzen innerhalb der Simulation bewertet werden. Dabei konnte keine Toleranzmodellierung in den Beiträgen festgestellt werden, gleich wenn methodisch bei der Abbildung der Systeme und Komponenten vorgegangen wurde. Somit konnte eine methodische Lücke hinsichtlich der Toleranzmodellierung aufgezeigt werden. Mit der dritten Frage zur Verwendung einer Methodik innerhalb der identifizierten Ansätze und Konzepte konnte kei-

ne übergeordnete Herangehensweise zur Entwicklung einer Simulationsumgebung festgestellt werden, so dass auch hier ein Entwicklungspotential gesehen wird. Häufig wurde die Modellierung des Systems selbst beschrieben, jedoch nicht die Komponenten, die zur Einbettung und zur zielgerichteten Entwicklung benötigt wurden. Eine Überprüfung der vorliegenden Ergebnisse wurde unter Berücksichtigung der Richtlinien von [RH08] vorgenommen.

5 *SimAQuAss4ConTest* - Eine Methode zur Entwicklung einer Simulationsumgebung

Dieses Kapitel beschreibt die in der Arbeit entwickelte Methode *SimAQuAss4ConTest* (engl.: Simulative Agile Quality Assurance for Consumer Tests), wie Simulationen im Rahmen der Entwicklung von Aktiven Sicherheitssystemen für Consumer Tests effektiv entwickelt werden können. Hierdurch wird ein systematisches Vorgehen bei der Realisierung der Simulationsumgebung und insbesondere eine Konzentration auf das Wesentliche in den Vordergrund gestellt. Denn leider führt insbesondere der Prozess der Modellbildung häufig zu einer Verschiebung der Prioritäten zu Ungunsten des eigentlichen Zieles: Der Gewinnung von zusätzlichen Informationen über Wirkungszusammenhänge, Systemverhalten oder andere Einflüsse als im Vergleich zu einem rein realen Test bzw. Absicherung. Dies geht einher mit einer vergleichsweise hohen Investition von Arbeit und Zeit zur Schaffung von hoch genauen Modellen, um möglichst ein detailgetreues und "realitätsnahes" Abbild der Wirklichkeit zu schaffen. Dies kann ein Niveau erreichen, sodass das zugrundeliegende aktive Sicherheitssystem oder auch allgemeine Fahrerassistenzsystem mit einem traditionellen Entwicklungsprozess fertiggestellt ist, während sich sein Simulationsmodell noch in der Entwicklung befindet, für dessen Test es eigentlich konzipiert und implementiert wurde.

Gelegentlich schließt sich die Argumentation an, dass dieses Modell im Folgeprojekt oder in einem Ähnlichen eingebunden werden könnte. Je spezieller jedoch ein Modell ist, desto geringer lässt sich eine Allgemeingültigkeit im Sinne einer Übertragung auf ähnliche Konstellationen realisieren. So kann zum Beispiel eine Veränderung der Technologie des realen Systems oder eine Komponente die bisherigen Effekte in der Realität der Art verändern, dass ein Modell davon im schlimmsten Fall unbrauchbar ist, zumindest im Sinne der Abbildungsgüte, die bei steigendem Detailgrad ebenfalls zulegt. Daher kann ein solcher Wechsel der Technologie das Modell unter Umständen derart unpräzise sein, sodass das Modell weitreichenden Anpassungen unterworfen werden muss. Daher sollte sich eine Simulationsumgebung mit seinen Modellen daran orientieren, was im Sinne einer adäquaten Abbildung der Wirklichkeit hinreichend ist. Je komplexer also ein Modell ist, desto unflexibler kann es auf Veränderungen von außen reagieren.

Ein Ansatz, um dieser Herausforderung zu begegnen, besteht u.a. darin, die Modelle auf einen verallgemeinernden Stand zu bringen und mit Hilfe des Prinzips der Modularisierung um konkretisierende Elemente des abzubildenden Kontextes zu erweitern, sodass die Wahrscheinlichkeit einer Wiederverwendbarkeit von Teilen des Modells gesteigert wird. In jedem Falle ist auch bei Wiederverwendung von verallgemeinerten Teilen des Modells eine erneute Prüfung der Angemessenheit der konkretisierenden Anteile notwendig, welche mit einem entsprechenden Aufwand, wie eingangs erwähnt, erfolgen muss.

An dieser Stelle ist zudem auch nicht berücksichtigt, dass die Regel- und Steuersoftware, welche beurteilt, ob eine Aktion erforderlich ist, noch nicht in dieser Argumentation einbezogen ist. Vielmehr geht es aktuell darum, das System selbst abzubilden.

Wie schon in Kapitel 2 deutlich geworden ist, besteht die Simulation nicht nur aus der Modellbildung, sondern auch aus einer Reihe von weiteren Aktivitäten wie z.B. Entwicklung eines geeigneten Frameworks zur Einbindung der Modelle, Schaffung einer Infrastruktur zur Auswertung sowie der Entwurf von Bewertungskriterien bzw. -methoden.

Im Folgenden wird daher eine Methode vorgestellt, die sich explizit mit der Identifikation einer Zielgröße befasst, auf die letztlich alle nachfolgenden Aktivitäten ausgerichtet sind. Die nachfolgende Grafik fasst diese einzelnen Prozessschritte überblicksartig zusammen. Die kommenden Abschnitte erläutern die einzelnen Prozessschritte dann näher im Detail, um sie in darauf folgenden Kapiteln in einer Fallstudie anzuwenden und damit ein Proof of Concept anzubieten.

5.1 Vorangestellte Überlegungen und Argumentationen

Um eine Methode entwickeln zu können, stellt sich zunächst die Frage nach der Definition des Begriffs “Methode”. Welche Semantik umfasst dieser Begriff und welche Konsequenz ergibt sich daraus für die Beantwortung der Forschungsfragen. Speziell liegt hier der Fokus darauf, was eine Methode letztlich bei der Lösung von Problemstellungen leisten kann und was ihr nicht möglich ist bzw. wo die Grenzen sind. Eine Abgrenzung zum Begriff hat die Gesellschaft für Informatik (GI) bereits geleistet, der in diesem Zusammenhang aufgegriffen wird.

5.1.1 Zum Begriff der Methode - eine Erläuterung

Die nachfolgenden Definitionen sind angelehnt an die Übersicht, die durch [AIB19] und auf Basis von [FBML98, BFH+07] zusammengestellt und zitiert wurden. Neben der Methode sind

noch eine Reihe weiterer Begriffe genannt, die in Beziehung zur Methode stehen. Diese werden ebenfalls für eine semantische Abgrenzung benötigt.

Der Begriff der Methode ist in der Informatik u.a. durch Hesse et al. geprägt worden. So definieren sie wie folgt:

> *Methoden sind planmäßig angewandte, begründete Vorgehensweisen zur Erreichung von festgelegten Zielen (im allgemeinen im Rahmen festgelegter Prinzipien). Methoden können fachspezifisch sein.* [HMF92] nach [AIB19]

Als Bestandteil einer Methode werden eine Notation, mehrere systematische Handlungsanweisungen und Regeln zur Kontrolle und Bewertung der Ergebnisse gesehen. Weiterhin ergänzen Balzert bzw. Schumann et al. nach Chrousts Auffassung:

> *Eine Methode ist eine systematische Handlungsvorschrift [oder Vorgehensweise], um Aufgaben einer bestimmten Klasse zu lösen. Sie beruht auf einem oder mehrerer Prinzipien. Die Handlungsvorschrift beschreibt [ausgehend von gegebenen Bedingungen], [wie] ein Ziel mit festgelegter Schrittfolge erreicht wird. Methoden sollen anwendungsneutral sein, (...).* [Bal82, Sch88, Cro92], zitiert nach [AIB19]

Der Begriff der Methodik ist definiert in Anlehnung zur Methode als

> *(...) eine Anweisung zur methodischen, folgerichtigen und zweckmäßigen Lösung einer wissenschaftlichen Aufgabe (...).* [Cro92] nach [AIB19]

Damit ergibt sich jedoch kein klarer Unterschied zum Begriff der Methode, so dass eine Abgrenzung an dieser Stelle nicht erfolgen kann. Vielmehr ergibt sich daraus ein Synonym für Methode. Als Methodenbündel werden Sammlungen von zusammenpassenden Methoden bezeichnet. Eine Methodenfamilie sind Methoden, die dem gleichen Prinzip unterliegen. [AIB19]

Da eine Methode in einen Rahmen von Prinzipien eingebettet ist, sind diese definiert als

> *(...) Grundsätze, die man seinem Handeln zugrunde liegt. Solche Grundsätze sind im allgemeinen nicht nur für einen bestimmtes Teilgebiet, sondern für das gesamte Fachgebiet oder zumindest wesentliche Teile davon womöglich auch über das Fachgebiet hinaus im wissenschaftlich-technischen Bereich gültig.* [Cro92] zitiert aus [AIB19]

Alternativ ist auch die folgende Definition aufgeführt:

> *Unter einem Prinzip wird ein allgemeingültiger Grundsatz verstanden, der aus der Verallgemeinerung von Gesetzen und wesentlichen Eigenschaften der objektiven Realität abgeleitet ist und als Leitfaden dient.* [Cro92, Sch88] zitiert nach [AIB19]

Dem gegenüber wird ein Verfahren wie folgt definiert:

> *Verfahren sind ausführbare Vorschriften oder Anweisungen zum gezielten Einsatz von **Methoden**. Eine Methode kann durch mehrere (alternative oder sich gegenseitig ergänzende) Verfahren unterstützt werden.* [HMF92] zitiert nach [AIB19]

Ein Verfahren konkretisiert somit eine Methode für bestimmte Problemstellungen oder -klassen. Dabei dient in der Softwareentwicklung ein Werkzeug als

> *(...) programmtechnische[s] Mittel zum automatisierten Bearbeiten von Informationsmengen. [Sch88] Sie automatisieren ein **Verfahren**. Es gibt Werkzeuge, die eine **Methode** (wenn auch manchmal nur implizit) implementieren.* [Cro92]

Im Zuge der Prinzipien taucht auch der Begriff der **Technik** auf:

> *Techniken operationalisieren **Prinzipien**. Sie werden eingesetzt, um vorgegebene Ziele leichter, schneller, sicherer, präziser oder in sonstiger Hinsicht günstiger erreichen zu können. Wir unterscheiden nicht-automatisierte Techniken in **Methoden** und **Verfahren** und durch **Werkzeuge** (teil-)automatisierte Techniken. (...) **Methoden** und **Verfahren** werden von der Wissenschaft erarbeitet, die Praxis liegt hauptsächlich in den Händen der Ingenieure.* [HMF92] zitiert nach [AIB19]

Wenn man also den Begriff Methode definieren möchte, so ist diese im Kontext der anderen hier aufgeführten Begriffe zu betrachten. Daraus ergibt sich ein System von Begriffen, welche untereinander über eine Beziehung verfügen. Die nachfolgende Darstellung 5.1 veranschaulicht diese Beziehungen untereinander.

Somit kann folgende Definition heraus gestellt werden:

> ***Eine Methode dient zur Erreichung eines bestimmten Zieles.** Dies erfolgt auf Basis von allgemeinen und fachspezifischen **Prinzipien**, manifestiert als systematische Handlungsanweisungen und unter Verwendung von bereits existierenden **Verfahren**. Dabei können **Werkzeuge zur (Teil-)Automatisierung** eingesetzt werden, jedoch ist eine Methode **anwendungsneutral** zu halten. **Eine Methode** zeichnet sich*

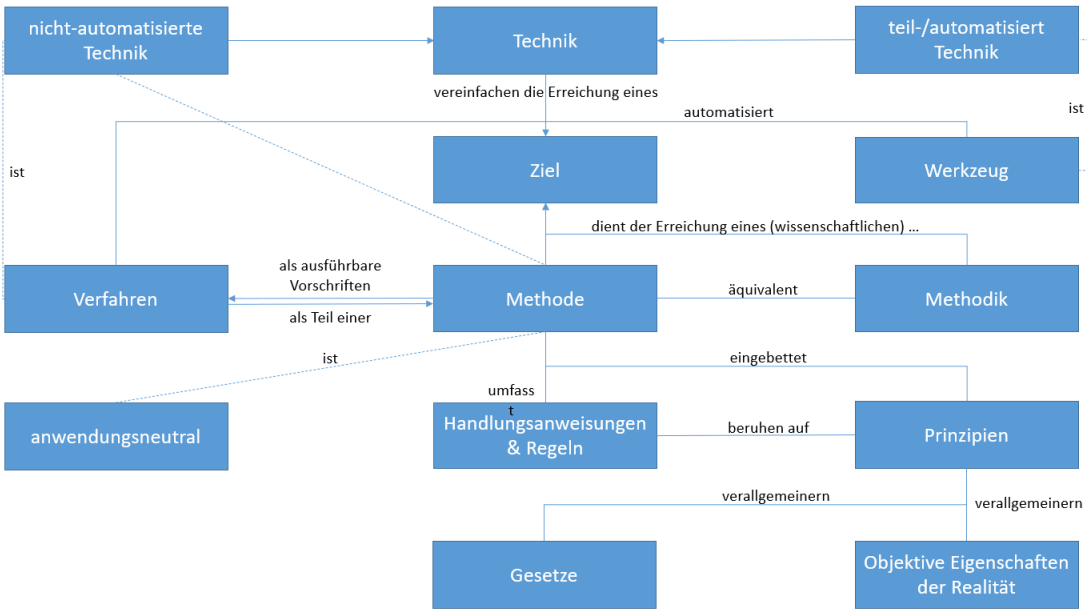


Abbildung 5.1: Der Begriff der Methode und seine Semantik

> *auch dadurch aus, dass* ***ein Teil nicht-automatisiert*** *ist. Dies bietet* ***die Möglichkeit****, vereinfachend den Fokus auf das Einbringen von* ***neuen Aspekten in bekannte Verfahren und Prinzipien*** *zu lenken und so forschungsrelevante Fragestellungen zu beantworten.*

Diese Definition bildet den Kontext für die Entwicklung und Anwendung der hier vorgestellten Methode.

5.2 Die Methode im Überblick

In einem ersten Prozessschritt wird die Erwartung an die Simulationsumgebung definiert. Dabei müssen wichtige Fragen im Vorfeld beantwortet werden, da andernfalls die Gefahr besteht, den späteren Verwendungszweck der Simulationsumgebung zu verfehlen oder die notwendigen Ressourcen ineffektiv zuzuordnen, also womöglich an einer Stelle einzusetzen, wo der Mehrwert nur begrenzt oder vergebens sein könnte. Der zweite Prozessschritt befasst sich übergeordnet mit der Schaffung einer geeigneten Infrastruktur, um die Ergebnisse aus dem ersten Schritt technisch umzusetzen. Während des dritten Prozessschrittes wird ein hinreichendes Aus- und Bewertungsverfahren entwickelt bzw. ausgewählt, um die Ergebnisse aus der Berechnung, deren Grundlage

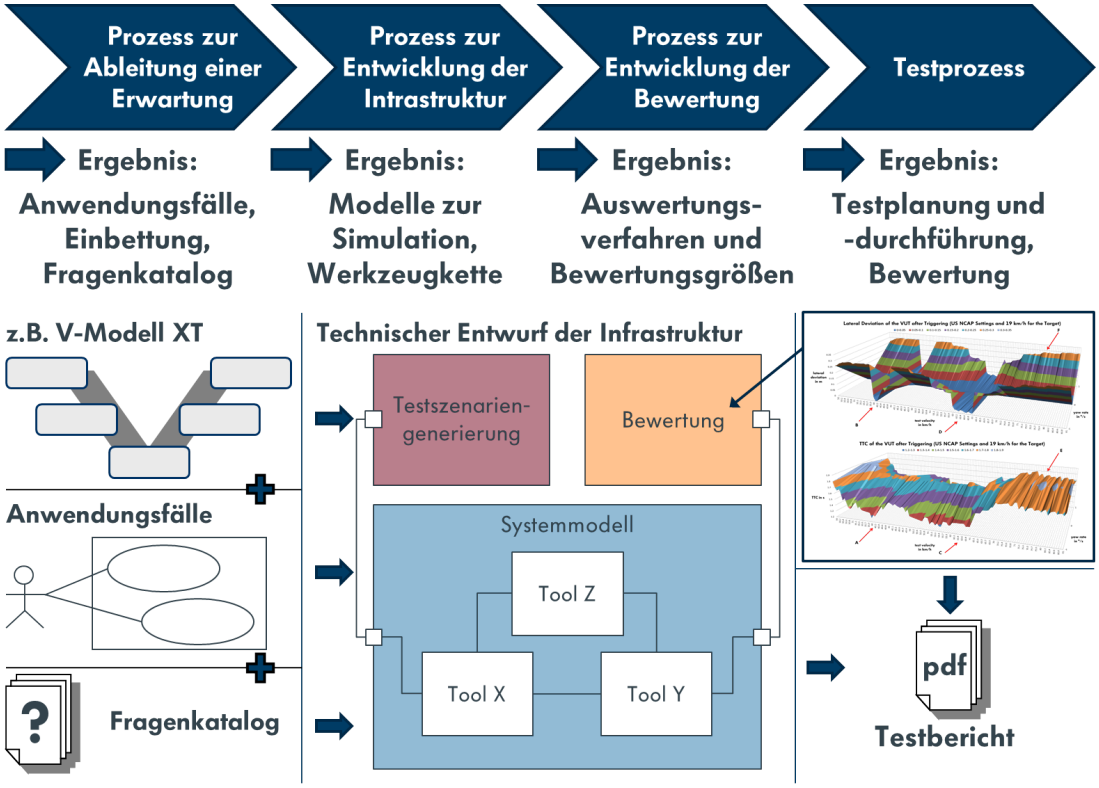


Abbildung 5.2: Die Methode zur effektiven Entwicklung einer Simulationsumgebung

die vorher designierte und umgesetzte Infrastruktur bildet. Im letzten Schritt erfolgt dann die Anwendung der Simulationsumgebung und die Ergebnisproduktion im Sinne des Fragenkatalogs aus dem ersten Prozessschritt.

Die nachfolgenden Abschnitte beschreiben die hier zusammengefassten Aspekte näher im Detail. Dabei wird eine Perspektive eingenommen, die es ermöglichen soll, eine Vielzahl an Problemstellungen im Kontext der Entwicklung von Fahrzeugfunktionen zu bearbeiten und eine geeignete und effektive Simulationsentwicklung zu ermöglichen.

5.3 Prozess zur Analyse und Modellierung des Untersuchungsraumes

Bevor auf die zwei zentralen Fragen dieses Abschnittes eingegangen wird, soll zunächst noch einmal explizit die Argumentation geschärft werden, aus welchen Gründen es wichtig ist, sich

vor allen Anfängen zu überlegen, was mit der Simulation bezweckt werden soll. Die Modellbildung, auf die traditionellerweise und die überwiegende Aufmerksamkeit während der Simulation gelegt wird, bildet nur einen wichtigen und dennoch nur **einen** Baustein für eine zielgerichtete Simulation zu Testzwecken.

Zwei zentrale Fragen stehen im Mittelpunkt dieses Prozessschrittes:

1. Welche(s) Problem/Herausforderung soll durch die Simulationsumgebung adressiert werden?
2. Welche Fragestellung während der Entwicklung soll am Ende beantwortet werden?

Die Wichtigkeit der Beantwortung dieser beiden Fragen kann nicht hoch genug eingestuft werden, da sie das konkrete Ziel zum Projekt definiert und damit Fehlentwicklungen vorbeugt, so dass die Simulation nicht zum Selbstzweck degradiert wird; ein leider nicht selten auftretendes Phänomen.

Zur Beantwortung dieser Fragen muss einerseits der Entwicklungsprozess der jeweiligen Fahrzeugfunktion genau analysiert werden, damit deutlich wird, wo, wann, was und wie entwickelt wird. Nur auf dieser Grundlage kann überlegt werden, wo eine Simulationsumgebung sinnvoll eingesetzt werden kann, damit sie einen Mehrwert generieren kann. In einem zweiten Schritt muss daraufhin geklärt werden, welches Ergebnis voraussichtlich erwartet wird und durch die Simulation erzielt werden soll. Durch Definition speziell dieser Frage lässt sich dann der Detailgrad eines entsprechenden Modells ableiten, ohne sich dabei in Nebensächlichkeiten zu verlieren.

5.3.1 Analyse des Entwicklungsprozesses für eine Simulation

In der Regel ist eine Simulationsumgebung innerhalb eines Entwicklungsprojektes zu integrieren. Grundsätzlich kann eine solche Umgebung aber auch im Rahmen eines Forschungsprojektes zur Grundlagenforschung eingesetzt werden. Idealerweise würde eine Entwicklungsaufgabe inkl. der vorangegangenen Forschungs- und Vorentwicklungsarbeit durch entsprechende Simulationsmaßnahmen begleitet werden. Die Praxis und insbesondere die Erfahrungen aus den Projekten in Unternehmen zeigen jedoch, dass die Simulation häufig aus anfänglich kostenintensiven Gründen keine oder nur unzureichend Berücksichtigung finden. Dies ist in Teilen auch darauf zurückzuführen, dass kein zusätzlicher Mehrwert erreicht wird, der zusätzlich vermarktet werden kann. Auf der anderen Seite führt eine Zurückhaltung bei Investitionen in Test und Simulation womöglich zu höheren Kosten aufgrund von Fehlerwirkungen im Feld.

Daher ist zu Beginn eines Simulationsprojektes der eigentliche Entwicklungsablauf bzw. der dazugehörige Prozess im Detail zu klären. Wichtigste Informationsquellen sind hier Prozessunterlagen, sofern die Prozesse auch in den jeweiligen Abteilungen dokumentiert wurden. Die Prozessunterlagen umfassen in der Regel, welches Ziel erreicht werden soll, welche Rollen vorhanden sind, welche (Teil-)Ergebnisse erreicht werden, usw. Für den Fall, dass eine solche Beschreibung des Entwicklungsprozesses nur unzureichend vorliegt, müssen weitere Analysemethoden angewandt werden, wobei der Schwerpunkt auf der Beschreibung der einzelnen Arbeitsschritte liegt. Die darin enthaltende Bewertung der Prozesse mit Schwachstellen- und Ursachenanalyse liegt nicht im Fokus.

Darüber hinaus können verschiedene Aspekte (z.B. Technologiewechsel, Personalunterdeckung) dazu führen, dass zuvor definierte Standardprozesse nicht mehr wie gewohnt durchlaufen und eingehalten werden können, sondern an der ein oder anderen Stelle adaptiert werden, woraus sich “Best Practices” für den operativen Einsatz entwickeln. Diese gilt es ebenfalls zu berücksichtigen.

In der UML [OMG12] werden dazu zwei Arten von Diagrammen genutzt:

- Aktivitätsdiagramme
- Use-Case Diagramme

Das Aktivitätsdiagramm gliedert die Prozessschritte und setzt sie in eine aufeinander bauende Reihenfolge. Zudem bietet diese Diagrammart die Möglichkeit, Kontroll- und Datenflüsse zu modellieren. Ein Use-Case-Diagramm beschreibt dann gruppiert oder auch detailliert die Anwendungsfälle des späteren Nutzers bzw. zeigt auf, welche weiteren Akteure dort noch involviert sind. Weiterhin können verschiedene Anforderungen als Anwendungsfälle innerhalb eines Systems abgebildet werden.

5.3.2 Definition der Fragestellung

Aus der Analyse des Entwicklungsprozesses geht hervor, an welcher Stelle bestimmte (Teil-)Ergebnisse ohne Simulation vorliegen und wo Simulationstechniken möglicherweise ihren Einsatz finden könnten. Diese wurden als Use-Cases im dazugehörigen Diagramm dargestellt.

Für diese grundsätzlichen Potentiale sind sodann (Forschungs-)Fragen so zu formulieren, dass die später geplanten Ergebnisse der Simulation diese expliziten Fragen beantworten. Der Detailgrad der Fragen ist hier so zu wählen, dass ein sinnvolles Modell entwickelt werden kann. In diesem Zusammenhang sollte das Prinzip **Teile und Herrsche** Anwendung finden, um die

Komplexität der Problemstellung beherrschbar zu gestalten. Die (Forschungs-)Fragen sollten dabei auch überprüfbar sein, bedeutet somit, dass auf ihrer Grundlage später eine überprüfbare Größe als Metrik herausgearbeitet werden kann. In diesem ersten Analyseschritt geht es schwerpunktmäßig darum, dass die Forschungsfrage eine Orientierung bietet, welche Zusammenhänge im Verhalten des Systems genauer untersucht werden sollen. Daher wird an dieser Stelle zunächst auf eine verbale Formulierung zurückgegriffen. Eine Verknüpfung mit einer Auswahl an Eingangs- und Ausgangsgrößen sowie Zielgrößen erfolgt dann im Rahmen des Infrastrukturaufbaus der Simulation.

Auch die Ausrichtung der Frage ist entscheidend, denn es können auf der einen Seite Fragestellungen aufgeworfen werden, die später als wiederkehrendes Ergebnis auch in nachfolgenden Projekten verwendet werden können, wobei die Änderungen im Systemkontext Berücksichtigung finden müssen. Auf der anderen Seite können hingegen einmalige explorative Fragestellungen entsprechend untersucht werden.

5.4 Prozess zur Entwicklung einer adäquaten Simulationsinfrastruktur

Wenn die zu untersuchenden Fragestellungen klar herausgearbeitet sind, beginnt der Prozess zur Entwicklung der Simulationsumgebung. Dabei müssen direkt im Anschluss an die Entwicklung der (Forschungs-)Fragen die Zielgrößen identifiziert werden. Darauf aufbauend wird dann der Untersuchungsgegenstand (hier: das System und der Systemkontext) je nach Fragestellung und Zielgröße modelliert.

5.4.1 Identifikation der Zielgrößen

Die Auswahl der Zielgrößen muss bereits zu einer frühen Phase erfolgen, da sie für die Modellierung von entscheidender Bedeutung sind. Dabei ist die Betrachtung des zugrundeliegenden Systems und der Systemkontext entscheidend. Üblicherweise verfügt das System schon über eine Teststrategie und einem Katalog von Tests, die bereits ohne simulative Methoden durchgeführt werden. Diese können als Ausgangsbasis dienen, um geeignete Zielgrößen zu identifizieren. Auch standardisierte Bewertungsverfahren von Dritten oder Gesetze können die Grundlage bilden, anhand derer sich die Zielgrößen identifizieren lassen.

Die Fahrzeugentwicklung baut in der Regel auf physikalischen Gesetzmäßigkeiten wie z.B. der Elektrotechnik, Bewegungslehre, Kinetik, usw. auf, so dass sich aus den jeweiligen mathemati-

schen Formeln bereits erste Wirkungszusammenhänge der einzelnen Größen identifizieren lassen. Ein Schema zur qualitativen Bewertung mit Hilfe von Metriken wird zu einem späteren Zeitpunkt in Kapitel 8 erfolgen.

5.4.2 Konzeptionelle Vorüberlegungen und Technikauswahl

Bei der Entscheidung von Technikauswahl im Rahmen von konzeptionellen Vorüberlegungen sind einige Rahmenbedingungen zu berücksichtigen bzw. festzulegen, falls nicht vorgegeben. Dazu gehören u.a. auch Budgetfragen, Fremdvergaben oder Eigenentwicklungen sowie bestehende Softwareprodukte mit individualisierten Anteilen oder Neuentwicklungen. Auch der Grad der Hybridisierung muss festgelegt werden und ergibt sich letztlich aus den bis hierhin vorangegangenen Schritten. Dies ist abhängig auch von der Modellierung und des Detailgrades, der abgebildet werden soll.

Auch eine entsprechende Recherche wie das SLR in Kapitel 4 ist durchzuführen, um neben den üblichen Prinzipien und Techniken aus der Standardliteratur neuere Ansätze in die Erstellung der Infrastruktur einfließen zu lassen. Welche Technik hier jedoch ausgewählt werden soll, ist von der Fragestellung der Untersuchung abhängig, so dass mehr als ein Rahmen mit den hier genannten Eckpunkten nicht weiter ausgeführt werden kann, weil die umgebenden Bedingungen der konkreten Untersuchung zu individuell sind.

5.4.3 Modellierung des Untersuchungsgegenstandes

Daran anschließend wird dann die genaue Modellierung vorgenommen. Dabei ist zwischen eigentlichem System und Systemkontext zu unterscheiden. Nun ist das weitere Vorgehen davon abhängig, welche Zielgröße ausgewählt wurde und welche Komponenten des Systems im Fokus liegen. Daran anknüpfend werden dann die umgebenden Komponenten aus dem Systemkontext modelliert, um die damit relevanten Beziehungen zwischen System und Umgebung abzubilden und die Eingangsdaten für das System-Under-Test zu liefern.

5.4.4 Design der Simulationsarchitektur

Hier wird dann adressiert, auf welche softwaretechnischen Komponenten und auf welche Architektur das Modellsystem abgebildet werden soll. Neben der architektonischen Abbildung des Modells müssen auch entsprechende weitere Systemteile berücksichtigt und in der Architektur festgehalten werden. Dazu zählen:

- Pre-Processing Komponenten
- Simulations- und Kalkulationskomponenten
- Post-Processing Komponenten

Die Pre-Processing Komponenten sorgen für die Konfiguration der Simulationsumgebung und stellen die für die Durchführung der Simulation notwendigen Daten bereit. Dies kann z.B. die Parametrierung der SUT-Komponenten, des Systemkontextes wie Witterungsbedingungen, Tageszeit, Fahrbahnbeschaffenheit, etc. oder zusätzlichen Informationen wie Navigationsdaten enthalten.

Die Simulations- und Kalkulationskomponenten bilden den Rahmen, in welche die Modelle integriert werden. Häufig werden dafür ganze Plattformen bereitgestellt, welche schon in Teilen entsprechende Modelle z.B. für den Systemkontext umfassen, also für Komponenten, die häufig wiederverwendet bzw. deren Eigenschaften als ausreichend identifiziert werden. In diesem Zusammenhang muss jedoch das Ziel der Simulation immer klar vor Augen bleiben, damit hinterher nicht die Zielgröße dem Modell folgt, sondern umgekehrt. Andernfalls reduziert man den Nutzen des Modells auf das bereits Vorhandene und erreicht unter Umständen nicht das anvisierte Ziel.

Das Post-Processing erfordert Komponenten, die bei der Auswertung und Bewertung der erzeugten Ergebnisse unterstützen und zweckgebunden z.B. als Report aufbereiten. Dabei muss die Auswertung nicht zwangsläufig in die Simulationsumgebung integriert werden. Sie kann auch als Stand-Alone Variante konzipiert sein und auf bereits vorhandene Software zurückgreifen. Sie sollte allerdings als Teil der Architektur aufgeführt werden, damit klar wird, auf welche Weise die Auswertung später erfolgen wird.

Insgesamt kann es zwei Ausrichtungen geben: Zum einen können alle Komponenten der Architektur in Eigenregie entwickelt und umgesetzt werden. Dies sichert auf der einen Seite, dass der jeweilige Zweck der Simulationsumgebung exakt der Vorgabe entspricht. Auch ist die Nachvollziehbarkeit aufgrund der Offenheit der Umsetzung zu jedem Zeitpunkt gewährleistet. Auf der anderen Seite führt dies dazu, dass ein umfangreiches Wissen über alle physikalischen Zusammenhänge erforderlich ist und die Abstraktionsfähigkeit dadurch erheblich eingeschränkt wird. Auch entsprechende Anpassungen und Weiterentwicklungen machen einen hohen Einsatz an Ressourcen notwendig, der nur begrenzt auf mehrere Stellen oder Fachbereiche im Unternehmen finanziell aufgeteilt werden kann.

Zum Anderen kann auf vertriebene Software gesetzt werden. In der Regel kann diese Variante finanziell günstiger ausfallen als bei der Eigenregie. Allerdings können die verwendeten

Modellkomponenten nicht dem gewünschten Einsatzzweck entsprechen, so dass umfangreiche Anpassungen durchgeführt werden müssen und weitere Kosten verursachen.

5.5 Prozess zur Entwicklung eines Bewertungsverfahrens

Dieser Prozess sorgt für die Informationsaufbereitung und sichert die Wissensgewinnung. Es gibt dabei drei wesentliche Auswahlmöglichkeiten. Dabei müssen zunächst die Signale oder Messgrößen ausgewählt werden, welche zur Bewertung herangezogen werden. Aufgrund der Individualität der Simulationsziele werden in diesem Rahmen Anhaltspunkte geliefert, wie eine Identifikation von Größen erleichtert und wie geeignete Metriken entwickelt werden können. Auch eine Betrachtung des Entwicklungsfortschritts über die Zeit kann hilfreich zu sein, zu erkennen, wo möglicherweise noch Handlungsbedarf im Sinne der Fehlerfindung und -beseitigung vorliegt oder genauere Erkenntnisse über das Verhalten des SUT zu erlangen sind.

5.5.1 Identifikation von Signalen zur Bewertung

Bei einem Signal handelt es sich um eine Messgröße bzw. um einen Parameter, der zwischen Systemkomponenten ausgetauscht wird. Bei einem Fahrzeug-CAN erhalten mehrere Steuergeräte als Systemkomponenten die Signale. Anhand des Headers kann ein Steuergerät identifizieren, ob das jeweilige Signal relevant ist. Die Auswahl der für die spätere Auswertung relevanten Signale kann zunächst über die Schnittstelle der realen Systemkomponente identifiziert werden. Es ist dabei sicherzustellen, dass im Rahmen der Modellbildung dieses Signal nicht aufgrund der Abstraktion aus dem Modell entfernt bzw. herausgenommen wurde. Aber auch konkrete, veröffentlichte Testverfahren können einen Anhaltspunkt liefern, welche Signale des Systems relevant sind. (vgl. z.B. beim EuroNCAP Tests: Restgeschwindigkeit).

Ein weiterer Anhaltspunkt für die Identifikation von Signalen können die Realtests bieten, da dort üblicherweise solche Fragestellungen aufgeworfen und Bewertungen vorgenommen werden. Der Vorteil der Simulation kann dabei sein, dass man gerade Zusammenhänge ausfindig machen kann, die so in der Realität nur mit hohem Aufwand möglich wären.

5.5.2 Ableitung von Metriken

Nach der Identifikation schließt sich die Entwicklung eigener Metriken an. Eine (Software-)Metrik wird nach IEEE

> *(...) allgemein [als] eine Funktion, die eine Softwareeinheit in einen Zahlenwert abbildet. Dieser Wert ist interpretiert als der Erfüllungsgrad eines Qualitätsziels für die Softwareeinheit (IEEE Standard 1061)*

Dabei ist zu unterscheiden, dass eine Darstellung von Signalverläufen über die Zeit noch keine Metrik bildet. Für eine Metrik muss klar definiert sein, in welchem Wertebereich sich das Signal aufhalten darf, damit der Test als Erfolg oder Misserfolg gewertet werden kann.

Es ist zu berücksichtigen, dass ein individuelles Simulationssetup neben Standardmetriken ein individuelles Metrikensystem umfasst, welche auf die Zielgröße(n) ausgerichtet sind. Neue Metriken lassen sich finden, indem ausgehend von einer Zielgröße jeweilige Signale miteinander z.B. arithmetisch verknüpft werden. Diese werden hier als Hauptmetriken bezeichnet. Nebenmetriken unterstützen den Entwicklungs- und Testprozess und erlauben die Beurteilung von möglichen Nebenzielen zur Effizienzsteigerung.

5.5.3 Design von Meta-Metriken

Die zuvor genannten Metriken ermöglichen die Beurteilung von Ergebnisdaten innerhalb einer Simulationsiteration. Entscheidend ist jedoch, wie sich Metriken über die Zeit entwickeln. Für diesen Fall haben Berger, Kühnel et al. den Ansatz der “Meta-Metrics” entwickelt, um den Entwicklern ein Tool zu liefern, mit dem der Ressourceneinsatz gezielter eingesetzt werden kann, wodurch insgesamt die Qualität von entsprechenden Entwicklungsartefakten gesteigert werden kann. [BBH+13]

Sie definieren Meta-Metriken wie folgt:

> *(...) The continuous determination of quantitative figures, which are defined over a set of results of simulation runs carried out for specific aspects, to steer and optimize the development process for an increased quality of the resulting product. [BBH+13]*

Dabei wird das Hauptaugenmerk darauf gelegt, auch bereits durchgeführte Simulationsläufe zu berücksichtigen und in einen zeitlichen Bezug mit dem aktuellen Simulationslauf und dessen Metriken zu setzen. Dadurch gelingt es, die Qualität über den Entwicklungsprozess hinaus zu bewerten und ggf. daraus Maßnahmen abzuleiten, sollte nicht die geforderte Qualität erreicht werden.

Die detaillierte Beschreibung erfolgt dann im Kapitel 8.

5.6 Prozess zur Durchführung der Simulation und Auswertung

In den vorangegangenen drei Abschnitten wurden die Voraussetzung für die Durchführung von Simulationen geschaffen. Der nachfolgende Abschnitt befasst sich damit, wie die Simulationen durchzuführen sind. Dabei ist zunächst entscheidend, wie ein Experiment aufgebaut wird. Auch bei der Durchführungen sind einige wichtige Aspekte zu beachten, auf die unten noch näher eingegangen wird. Abgeschlossen wird der Abschnitt mit der Diskussion der Ergebnisse und der Ergebnisdarstellung. Als Maßstab wird dabei auf die Arbeit von Runeson, Höst et al. [RH08, WRH+12] Bezug genommen, da hier bereits eine grundlegende Methodik speziell im Software Engineering hinsichtlich empirischer Studien geschaffen wurden. Ihre Adaption auf den vorliegenden Kontext und ihre kritische Prüfung ist Teil dieser Arbeit.

5.6.1 Entwurf einer Untersuchung

Nach Runeson et al. [WRH+12] gibt es im Rahmen des Software Engineering mehrere Möglichkeiten, empirische Untersuchungen durchzuführen. Eine Untersuchung per Simulation kann in diesem Zusammenhang als solche gewertet werden. Sie stellen zu Anfang zwei Paradigmen voran, anhand derer sich empirische Studien unterscheiden lassen:

- Explorative Untersuchungen (engl.: explorative research)
- Erklärende Untersuchungen (engl.: explanatory research)

Explorative Studien betreffen nach Runeson et al. Objekte, die in ihrer natürlichen Umgebung untersucht werden, um diverse Phänomene dabei aufzudecken und daher einen flexiblen Aufbau benötigen. Sie werden auch unter qualitativ bewertende Untersuchungen zusammengefasst. Explanative bzw. erklärende Untersuchungen haben einen quantitativen Fokus und betrachten die Beziehung zwischen mehreren Objekten, um daraus Ursache-Wirkung-Vergleiche zu extrahieren. Daher benötigen solche Untersuchungen in der Regel einen gleichen Aufbau und werden meist im Rahmen eines kontrollierten Experiments mit gleichen Bedingungen durchgeführt. [WRH+12]

Dabei gibt es drei Arten von empirischen Untersuchungen, die sich anhand einzelner Kriterien unterscheiden lassen:

- Umfragen
- Fallstudien
- Experimente

Als Umfragen werden Informationsbeschaffungen von Personen oder über Personen verstanden, welche Wissen, Einstellungen und Verhalten beschreiben, vergleichen oder erklären. [WRH+12, S. 58] Eine Fallstudie befasst sich im Software Engineering mit der Untersuchung eines aktuellen Phänomens auf Basis mehrerer Beweisquellen bei einer oder mehreren Instanzen in ihrem realen Kontext, insbesondere wenn die Grenzen zwischen Phänomen und Kontext nicht klar benannt werden können. [WRH+12, S. 146] Beim (kontrollierten) Experiment wird ein Faktor oder eine Variable des Versuchsaufbaus einzeln geändert. Dabei werden je nach Subjekt unterschiedliche Einflüsse vorgenommen, während andere Variablen konstant bleiben, um den Effekt auf die Ausgangsvariablen zu messen. Bei human-orientierten Experimenten gehen diese Einflüsse von Menschen aus und bei technologisch-orientierten Experimenten liegen die technischen Einflüsse im Fokus. Ein Quasi-Experiment ist ähnlich zu einem regulären Experiment, wobei die Übertragung der Einflüsse auf Subjekte nicht zufallsbasiert ist, sondern sich aus Charakteristika der Subjekte und Objekte ergibt. [WRH+12]

Strategie	Design Typ	Qualitativ/quantitativ
Umfrage	Fest	Beides
Fallstudie	Flexibel	Beides
Experiment	Fest	Quantitativ

Tabelle 5.1: Design Typen und qualitative und quantative Daten in empirischen Studien [WRH+12]

Es ist also notwendig, im Vorfeld die richtige Entscheidung dahingehend zu treffen, für welche Art einer empirischen Untersuchung gewählt wird. Zweifelsfrei wird sich eine Umfrage in diesem Kontext kaum ergeben. Das Simulationsergebnis kann allerdings davon abhängen, ob eine Fallstudie ODER ein Experiment gewählt wird. Der Vorteil der Simulation ist hier, dass die meisten Systemgrößen beeinflussbar sind und unter den gleichen Startbedingungen durchgeführt werden können. Dies spricht nach Runeson et al. eher einem experimentellen Aufbau. Gleichwohl wird hier die Vermeidung der Zufälligkeit angestrebt, um einen möglichst deutlichen Zusammenhang zwischen dem Phänomen und dessen Ursprungs herzustellen.

Speziell der letzte Aspekt ist für die hier gestellte Methode entscheidend. Ihre Übertragung auf den vorliegenden Fall zur Untersuchung von Notbremssystemen in Consumer Tests wird Anwendung finden. Dabei wird unter anderem ein Schwerpunkt auf die Szenariengenerierung gelegt, da die jeweilige Reaktion des Assistenzsystems mit unterschiedlichen Trajektorien in den sehr streng vorgegebenen Grenzen der Consumer-Tests unterschiedliche Auswirkungen auf die einzelnen Systemkomponenten haben kann.

Definition von Untersuchungsraum und -tiefe

Auch für das einzelne Experiment ist die Festlegung, die Beschreibung des Untersuchungsraumes sowie die geplante Tiefe der Untersuchung zu definieren. Folgende Punkte können dabei als Rahmen für diese Definition angesehen werden:

- Gegenstand der Untersuchung (was wird untersucht?),
- Zweck der Untersuchung (aus welchem Grund wird untersucht?),
- qualitativer Schwerpunkt (welcher Effekt wird untersucht?),
- Perspektive (aus welcher Sicht wird untersucht?) und
- Kontext (wo wird die Untersuchung durchgeführt?).

Die Abgrenzung zu Kapitel 9 besteht darin, dass zuvor der Fokus auf die Infrastruktur der Simulationsumgebung gelegt wurde. Hier wird nun das jeweilige Einzelexperiment in den Vordergrund gerückt, anhand derer eine überprüfbare und nachvollziehbare Aussage zur Simulationsaufgabe gemacht werden kann. [WRH+12]

Planung des Experiments

Runeson et al. beschreiben sieben einzelne Prozessschritte, die am Ende zu einem Design führen.

1. Kontext Auswahl
    - Off-line vs. on-line
    - Studenten vs. Experten
    - Planspiel vs. reale Problemstellung
    - spezifisch vs. generell
2. Hypothesen Formulierung
3. Variablenauswahl
4. Prüfling bzw. Subjekt Auswahl
5. Entwurfstyp des Experiments auswählen
    - Generelle Entwurfstypen
        - Randomization

  - blocking
  - balancing
- Standard Entwurfstypen
  - One factor with two treatments.
  - One factor with more than two treatments.
  - Two factors with two treatments.
  - More than two factors each with two treatments.

6. Instrumentation
7. Validitätsabschätzung

Bei der Kontextauswahl müssen weitere Rahmenbedingungen bzw. Dimensionen festgelegt werden, ob z.B. ein neues Verfahren direkt oder begleitend zum alten angewendet wird. Auch die Auswahl möglicher Personen mit Bezug auf ihr Erfahrungswissen kann das Ergebnis ebenso wie bei einem Planspiel oder einer realen Problemstellung beeinflussen.

Grundsätzlich sehen Runeson et Höst eine Hypothesenformulierung vor. Diese dient als Grundlage für die Akzeptanz einer Untersuchung oder auch ihrer Zurückweisung. Dabei werden sowohl eine Null-Hypothese als auch eine widerlegbare Hypothese herausgearbeitet. Danach werden die jeweils unabhängigen und abhängigen Variablen ausgewählt, wobei Runeson et al. anführen, dass idealerweise nur eine abhängige Variable in der Hypothese Berücksichtigung findet, damit hinterher bei der Validierung der Schlussfolgerung die Komplexität beherrschbar bleibt. Die generellen und die Standard Entwurfstypen befassen sich jeweils mit der Strukturierung des Ablaufs, also bspw. ob die Zuordnung der Faktorausprägung zufällig, geblockt oder ausgeglichen sein soll bzw. wie viele Faktoren bei der Untersuchung berücksichtigt werden sollen. Daraus ergeben sich nach Runeson et al. die jeweiligen Analysemöglichkeiten während der Bewertung der Ergebnisse. [WRH+12]

Eine solche Hypothesenformulierung kann jedoch aus vereinfachenden Gründen zunächst auf eine formulierte Ausgangs- bzw. Forschungsfrage beschränkt werden, die dann wiederum argumentativ und auf Grundlage der Simulationsergebnisse beantwortet werden. Dabei müssen die Argumente nachvollziehbar und im Kontext allgemeiner Erkenntnisse eingebettet sein und nicht einer “gefühlten” Wahrheit unterliegen, damit sie auch entsprechend widerlegbar sind.

Ein weiterer wichtiger Aspekt bei der Planung ist die Validitätsbetrachtung (engl.: threats to validity). Die vier Varianten

- interne Validitätsbetrachtungen
- externe Validitätsbetrachtungen
- konstruktive Validitätsbetrachtungen
- abschließende Validitätsbetrachtungen

stehen zudem in wechselseitiger Beziehung zu einander und beeinflussen sich je nach Ausprägung gegenseitig. Das heißt, dass bei einer starken Berücksichtigung der Validität in einem Aspekt ein anderer schwächer bewertet wird. Dies gilt es bei Sicherstellung der Validität zu berücksichtigen und entsprechend eine hinreichende Balance zu finden. [WRH+12]

Nach Abschluss der Planung wird das Experiment durchgeführt.

5.6.2 Durchführung des Experiments

Bei der Durchführung ist darauf zu achten, dass die zuvor festgehaltenen Randbedingungen eingehalten werden. Zu diesen Randbedingungen gehören die festgelegten Anfangszustände der Simulationskomponenten und Modelle sowie die definierten Zustandsänderungen, die durch die Eingangsdaten beschrieben sein können. Unter Beobachtung sollte dabei auch das Zeitverhalten innerhalb der Simulation sein, damit eine spätere Abschätzung darüber gemacht werden kann, inwiefern eine mögliche Schwankung als Fehler deklariert werden kann.

Durch den Aufbau der Simulation und entsprechender Funktionalität ist bei der Durchführung auf einen immer gleichen Aufbau abgesehen von den eingehenden und sich ändernden Parametern zu achten, damit hinterher mögliche Anomalien identifizierbar sind. Auch muss der Ablauf der Durchführung in diesem Zusammenhang gleich gestaltet sein. Bei der Variation der Parameter sollte die Nachvollziehbarkeit der Variation im Vordergrund stehen, damit keine wechselseitige Beeinflussung der Änderungen ausgeschlossen werden kann. Das führt dazu, dass eine entsprechend große Zahl an Eingangsdateien generiert werden muss. Dieser Input liefert verschiedene Werte im Zeitverlauf und nur jeweils eine Variable hat eine Variation in ihren Werten, während die anderen Variablen in den unterschiedlichen Dateien gleich sind.

Da es sich bei der Simulation in der Regel um immer wiederkehrende Abläufe handelt, kann für die Durchführung eine Parallelisierung der Infrastruktur angedacht werden. Die limitierenden Faktoren sind dabei die Kosten für Aufbau und Einrichtung der parallelen Systeme, die dafür notwendige Arbeitskraft und die Betriebskosten. Hier darf ich eine Amortisationsrechnung nicht fehlen, damit eine nachhaltige Lösung etabliert werden kann. Weiteres Entscheidungskriterium ist hier, inwiefern die zu untersuchende Fragestellung auch mit der Simulationsarchitektur

untersucht werden kann und welche Umbaumaßnahmen müssen für die Untersuchungen daran vorgenommen werden. Auch entscheidet der einzubringende Aufwand darüber, zu welchem Zeitpunkt eine Amortisation erreicht wird.

5.6.3 Analyse und Diskussion der Ergebnisse

Die mit der Simulationsumgebung erzeugten Daten müssen nun aufbereitet, ausgewertet, analysiert und im Kontext erörtert werden. Dabei hilft die intensive Vorbereitung der Experimente in diesem Arbeitsschritt. Neben den deskriptiv-statistischen Methoden und der späteren analytischen Methoden müssen die bereits im Vorfeld getroffenen Annahmen zu den “Threats to Validity” überprüft und für die Einordnung der Ergebnisse aufgeführt werden.

Mögliche Einschränkungen bei den gemachten Annahmen zur Simulation hinsichtlich der getroffenen Abstraktionen und Vereinfachungen bei der Modellbildung müssen in diesem Abschnitt kenntlich gemacht werden, um eine Vergleichbarkeit zu einem immer im Fokus stehenden realen Test zu gewährleisten. Dabei ist speziell auf die Abstraktionen und die Vereinfachungen der Modellierung einzugehen. Diese Überlegungen müssen für die Vergleichbarkeit gewürdigt werden.

5.6.4 Testberichterstellung

Im letzten Schritt müssen die gewonnenen Ergebnisse und Erkenntnisse entsprechend dokumentiert werden. Der Aufbau des Dokuments hängt vom jeweiligen Simulationskontext und der untersuchten Fragestellung ab. Wichtig an der Stelle ist, die genauen Randbedingungen und möglichen Einschränkungen der gewonnenen Ergebnisse ebenfalls hinreichend zu dokumentieren, um systematische Fehler bei der weiteren Nutzung der Erkenntnisse und eine Übertragung auf nachfolgende Untersuchungen zu vermeiden.

Eine mögliche Gliederung eines solchen Dokuments geben Runeson et al. innerhalb ihrer Arbeit als Leitfaden heraus. Dabei wird neben der im weiteren Verlauf darzulegenden Ergebnisse und Threats to Validity auch eine Empfehlung ausgesprochen, wie man die zu untersuchende Fragestellung, den Systemkontext, die verwendete Methode und die später gemachten Schlussfolgerungen aufbereitet, um nachfolgenden Lesern einen umfassenden Überblick zu gewähren. Die bereits gemachten Veröffentlichungen zu dieser Arbeit wurden nach gleichem Schema aufgebaut. [BBH+14b, BBH+15b, BBH+15a]

5.7 Zusammenfassung

Aufgrund des Ergebnisses aus Kapitel 4 wurde abgeleitet, dass weiterer Bedarf existiert, Methodiken zum Aufbau von Simulationsumgebungen zu entwickeln und gleichzeitig auch den Kontext der Consumer Test als Grundlage zu verwenden, da in diesem Zusammenhang kaum Beiträge ausgewiesen sind.

Dafür wurde zunächst der Begriff der Methode bzw. der Methodik semantisch eingeordnet. Danach wurde eine Methodik eingeführt, welche sich insbesondere mit zwei Ausgangsfragen zu Beginn auseinander setzt. Im ersten Schritt muss geklärt werden, welches Entwicklungsproblem durch die Simulation adressiert werden soll. Sie rückt den Entwicklungsprozess selbst in den Vordergrund und soll definieren, wie und wo eine Umgebung zielgerichtet eingesetzt werden kann. Im zweiten Schritt stellt sich die Frage, welche Fragestellung die Simulation beantworten soll. Hier geht es um den spezifischen Erkenntnisgewinn während der Entwicklung des Systems oder der Funktion an sich.

Die vorgestellte Methodik ist in vier Prozessbausteine aufgeteilt. Im ersten Baustein wird mit Hilfe von agilen Modellierungstechniken der UML der Entwicklungsprozess innerhalb der Organisationseinheit analysiert und das Einsatzfeld von Simulation definiert. Aufbauend auf dieser Erkenntnis wird dann das entwicklungsspezifische Ziel formuliert.

Im zweiten Baustein werden anschließend für die Fragestellungen geeignete (Mess-)Größen abgeleitet, die als Anhaltspunkt für die Simulationsarchitektur dienen. Insbesondere hier wird ein Ansatz gewählt, der zum Ziel hat die für die Simulation notwendigen Modelle so weit wie möglich in ihrer Komplexität gering zu halten. Sie fügen sich in eine Gesamtarchitektur mit entsprechenden Komponenten für Pre- und Postprocessing, wie beispielsweise Werkzeuge zur Testfallgenerierung.

Im dritten Baustein werden die Bewertungsverfahren herausgearbeitet, die zur Beurteilung der Ziel- und Messgrößen dienen. Hier stehen einerseits Metriken zur Verfügung, welche dabei unterstützen zu einem jeweiligen Zeitpunkt eines Tests oder Simulationslaufes zu bestimmen, inwiefern jener erfolgreich war. Die eigens entwickelten Meta-Metriken erlauben andererseits, solche Beurteilung über mehrere Simulationsläufe in einem selektierten Zeitraum durchzuführen und damit über eine Entwicklungsperiode die Veränderung der Qualität einer Testkomponente zu ermitteln.

Im letzten Prozessbaustein wurde beschrieben, wie aus den gewonnenen Daten und Informationen eine Erkenntnis im Sinne der zu Anfang ausgearbeiteten Fragen erzielt werden können. Dabei wurde die Methodik zu empirischen Studien von Runeson und Höst herangezogen und für

den vorliegenden Kontext adaptiert. Neben dem Aufbau des Experiments ist auch entscheidend, welche Aspekte der Validität für den konkreten Simulationslauf gelten, bevor ein möglicher Erkenntnisgewinn abgeleitet werden kann.

Mit Vorstellung dieser Methode werden in den folgenden Kapiteln die verschiedenen Prozessbausteine weiter vertieft und weitere Hilfestellungen in der Entwicklung von Simulationsumgebungen gegeben.

6 Analyse und Modellierung des Untersuchungsraumes

Im Folgenden wird die Methodik aus Kapitel 5 angewandt und durch explizite Modellierung der Vorgehensweisen konkretisiert. Zunächst wird der Entwicklungsprozess in der vorliegenden Projektsituation, hier in der Funktionsentwicklung "Consumer Test", dahingehend analysiert, an welcher Stelle eine Simulationsumgebung sinnvoll eingesetzt werden kann. In einem weiteren Schritt werden dann konkrete Fragen formuliert, die für den jeweiligen Anwendungsfall interessant und im Sinne der Funktionsentwicklung als hilfreich anzusehen sind.

6.1 Identifikation von Aufgaben im Entwicklungsprozess

Wie bereits in Kapitel 3 deutlich wurde, gilt es zwei Entwicklungsziele in Einklang zu bringen: Zum einen muss die jeweilige Funktion dem Kundenwunsch entsprechen und darf nicht zu unlogischem oder fehlerhaftem Verhalten für den Kunden führen. Gleichzeitig gilt es in diesem Zusammenhang aber auch die Consumer-Test-Anforderungen zu erfüllen, die nicht immer deckungsgleich sein müssen. So kann beispielsweise eine frühzeitige Auslösung einer Notbremsfunktion hier im Sinne der Consumer-Test Anforderungen sein, um eine hohe Punktzahl zu erreichen, jedoch aus Kundensicht einen zu frühen Eingriff darstellen. Daher ist eine robuste Auslegung der Funktionen ein wesentliches Ziel, so dass beide Teilziele, möglichst hohe Punktzahl in den Consumer Tests bei gleichzeitiger Verringerung von False-Positive bzw. True-Negative Fällen, in Einklang zu bringen sind.

Bevor auf den konkreten Systemkontext eingegangen wird, seien noch ein paar allgemeine Aspekte zur Identifikation von Informationsquellen genannt, aus denen die notwendigen Informationen zum Entwicklungsprozess gewonnen werden können. Die hier aufgeführte Liste ist dabei als eine Auswahl zu verstehen und lässt sich nur schwer allgemeingültig formulieren, da die Unterschiedlichkeit der Entwicklungsprozesse in den Unternehmen eine individuelle und kontextabhängige Betrachtung erforderlich macht.

- Gesamtheitlicher und in der Regel vertraulicher bzw. geheimer Produktentstehungsprozess über alle Fach- und Geschäftsbereiche
- Dokumente zum Jobsplit oder Verantwortlichkeiten-Matrix
- Arbeitsanweisungen
- Liste der Dokumente und Aufzeichnungen einer Organisationseinheit
- Eigene Dokumentationen zu Arbeitsprozessen
- Arbeitsergebnisse aus vorherigen Projekten
- Best-Practices
- individueller Testanforderungskatalog

Die Dokumentation zum gesamtheitlichen Produktentstehungsprozess eines Unternehmens, der in der Regel ein wohl gehütetes Geheimnis ist und der Kernprozess des wirtschaftlichen Handelns darstellt, kann als erster Einstieg dafür dienen. Üblicherweise wird die eigene Organisationseinheit innerhalb dieses Prozesses ein oder mehrere Prozessschritte ausführen und entsprechende Ergebnisse beisteuern. Die Dokumente zum Jobsplitt oder die Verantwortlichkeiten-Matrix sind weitere Informationsquellen, welche Aufschluss über den Entwicklungsprozess geben.

Da die Darstellung dieser Prozesse sich wegen der Reichweite über mehrere Organisationseinheiten erstreckt, ergibt sich die Detaillierung und Konkretisierung der jeweiligen Prozessschritte aus den Arbeitsanweisungen, die für die Organisationseinheit Gültigkeit besitzen. Idealerweise verfügt die Organisationseinheit, in der die Simulation angesiedelt werden soll, auch über eine Liste der Dokumente und Aufzeichnungen, so dass daraus weitere Umfänge von Arbeitsergebnissen identifiziert bzw. abgeleitet werden können.

Sollten etwaige Informationsquellen so nicht auszuwerten sein, können grundsätzlich die vorangegangenen Projekte und deren Arbeitsergebnisse als Anhaltspunkt dienen. Es gilt dabei jedoch zu prüfen, inwiefern diese bereits einer Standardisierung unterzogen wurden. Evtl. wurden dazu Best-Practices Methoden dokumentiert, welche zusätzliche Details zum Entwicklungsprozess preisgeben. Neben all diesen Quellen kann ein individueller Testanforderungskatalog als Informationsquelle dienen.

6.2 Modellierung und Visualisierung von Entwicklungsschritten und -prozessen

Die individualisierten und gesamtheitlichen Produktentstehungsprozesse bilden das Herzstück einer Unternehmung, da hier in der Regel das über die Jahrzehnte gesammelte Erfahrungswissen berücksichtigt ist, sodass mit den Informationen zu den Prozessen sehr sensibel umgegangen werden muss und wird. Eine Abstraktion insbesondere im externen Umgang mit den Prozessen ist dabei nicht zu vermeiden, soll der aus dem Produktentstehungsprozess abgeleitete Wettbewerbsvorteil weiter bestehen bleiben.

Dieser wichtige Punkt führt dazu, dass im Rahmen dieser Arbeit eine detaillierte Betrachtung des zugrundeliegenden Entwicklungsprozesses im vorgegebenen Rahmen von Volkswagen nicht möglich ist. Dennoch wird grundsätzlich auf eine Methodik zur Darstellung von Prozessen und die Erweiterung von bestehenden Prozessen eingegangen, um prinzipiell die Möglichkeit zu bieten, z.B. noch undokumentierte bzw. implizite Vorgehensweisen explizit zu machen.

6.2.1 Ziel und Notation von Aktivitätsdiagrammen

Aktivitätsdiagramme sind Teil der UML 2.0 [OMG12, OMG15] und haben ein umfangreiches Portfolio an Elementen, um Kontroll- und Datenflüsse zu visualisieren. Dieses ursprüngliche Ziel, innerhalb eines zu entwickelnden oder bereits entwickelten Systems die Daten- und Kontrollflüsse darzustellen, wird hier genutzt, um die einzelnen Aktivitäten innerhalb einer Entwicklungsphase zu beschreiben. Alle Elemente én detail darzustellen, würde den Rahmen an dieser Stelle überschreiten, daher wird sich zunächst nur auf die für die Prozessbeschreibung wichtigen Elemente eingegangen. Grundsätzlich lässt sich dann der Prozess auch mit den erweiterten Elementen vervollständigen. Die Aussagekraft wird durch die vereinfachende Darstellung nicht geschmälert. Für einen vertiefenden Einblick sei auf die [BHW15, KS15, OMG15] Literatur verweisen.

Knoten wie in Abbildung 6.1, zu denen die Kontroll- und Objektknoten, Aktionen und weitere Aktivitäten, aber auch die Kanten zur Abbildung von Kontroll- und Objektflüssen zählen, bilden zusammen eine Aktivität. Graphisch wird eine Aktivität durch ein abgerundetes Rechteck illustriert. Rechtecke an den Seiten beschreiben die Eingangs- und Ausgangsparameter. Aktionen als Basiselemente beschreiben Verhalten und bilden Funktionen ab, die nicht weiter zerlegbar sind. Sie wandeln Eingaben in Ausgaben um. [BHW15, KS15, OMG15]

Durch diese Elemente lassen sich wie in Abbildung 6.2 eingehende und ausgehende Signale

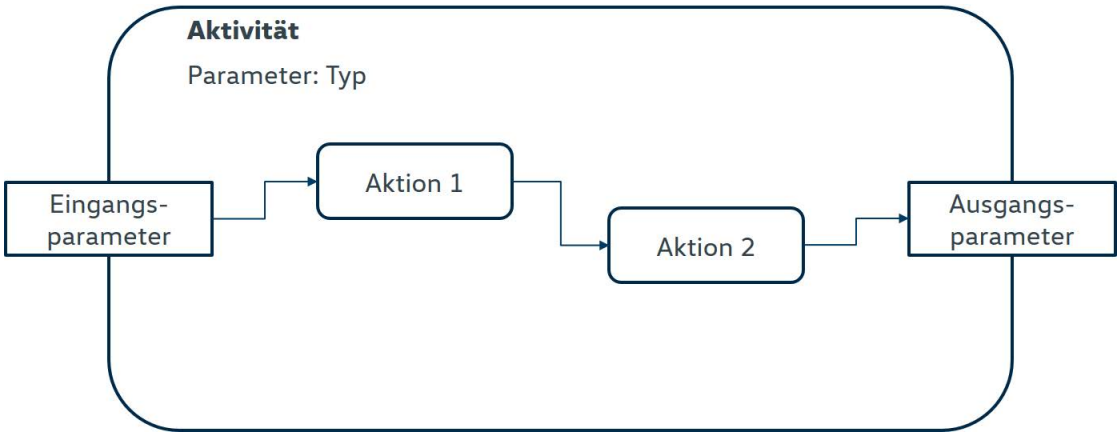


Abbildung 6.1: Beispiel für ein Aktivitätsdiagramm

Abbildung 6.2: Notation für Signale

darstellen, die an einer anderen Stelle im System oder einem externen System empfangen werden bzw. von dort aus gesendet worden sind. [BHW15, KS15, OMG15]

Abbildung 6.3: Notation für Zeitereignis

Mit diesem Element in Abbildung 6.3 können zeitgesteuerte Ergebnisse visualisiert werden, die Aktionen auslösen.

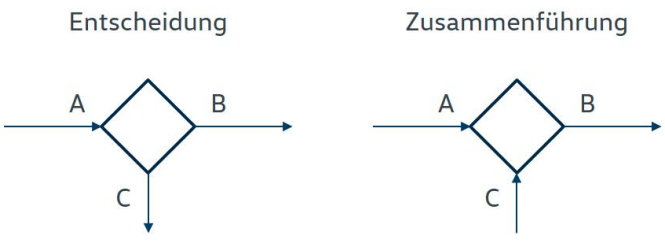


Abbildung 6.4: Notation für Kontrollstrukturen

Mit Hilfe von Kontrollknoten wie in Abbildung 6.4 können Kontrollflüsse strukturiert werden. Bei einer Entscheidung wird in Abhängigkeit einer Bedingung auf die eine oder andere Kante verzweigt. Bei der Zusammenführung werden zwei Kanten zu einer verschmolzen, wobei dies auch in Abhängigkeit einer Bedingung geschehen kann.

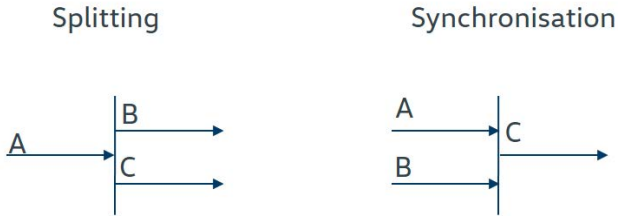


Abbildung 6.5: Notation für Splitting und Synchronisation

Der Splitting-Konten aus Abbildung 6.5 nimmt eine Kante auf und teilt sie parallel auf zwei neue Kanten. Bei der Synchronisation werden zwei Kanten zu einer wieder zusammengeführt.

Abbildung 6.6: Notation für Start und Endeknoten

Durch einen Startknoten beginnt eine Aktivität, bei einem Endknoten wird sie komplett beendet. Bei einem Ablaufende wird ein Zweig einer Aktivität beendet, die Aktivität selbst läuft aber weiter.

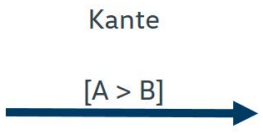


Abbildung 6.7: Notation für Splitting und Synchronisation

Ein Pfeil wie in Abbildung 6.7 verbindet zwei Elemente. Eine Bedingung für das Durchlaufen wird in eckigen Klammern angegeben.

Durch Objektknoten können innerhalb von Aktivitätsdiagrammen Objekte oder Daten repräsentiert werden. Sie können entweder als Rechtecke zwischen zwei Kanten oder als zwei Pins an zwei Aktionen repräsentiert werden, die durch eine Kante miteinander verbunden

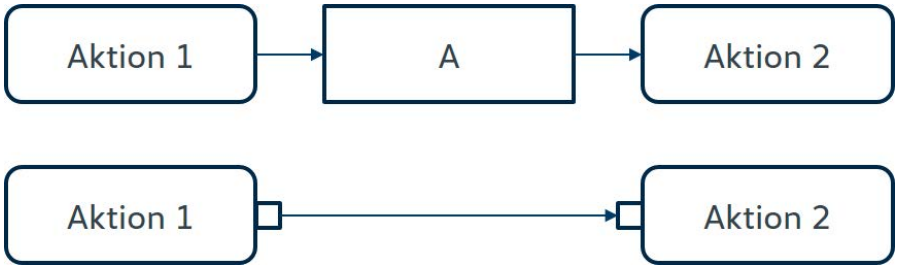


Abbildung 6.8: Notation für Datenübergänge

sind. In einem solchen Fall handelt es sich dann um einen Datenfluss; sofern keine Elemente zur Objektbeschreibung verwendet wurden, handelt es sich um einen Kontrollfluss. [BHW15, KS15, OMG15]

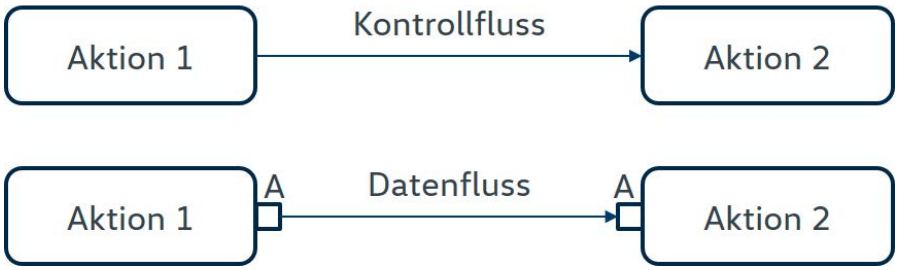


Abbildung 6.9: Notation für Kontroll- und Datenfluss

Nachdem nun die wichtigsten Elemente eines Aktivitätsdiagramms präsentiert wurden, wird mit der Vorstellung der Methodik zur Beschreibung von Prozessen fortgesetzt.

6.2.2 Methodik zur Beschreibung von Prozessen

Diese Methodik ist als Ergänzung zu betrachten, da grundsätzlich eine Unternehmung ohnehin schon über eine Prozessvisualisierung verfügen sollte. Wenn allerdings ein exploratives Umfeld die Rahmenbedingungen skizziert, können Teile des Prozesses noch nicht genau oder auch aufgrund der Neuartigkeit spezifiziert werden.

Im Folgenden wird eine Methodik beschrieben, die dabei helfen soll, zielgerichtet ein Aktivitätsdiagramm herauszuarbeiten. Dabei wird ergänzend zu bereits beschriebenen Prozessen vorgegangen.

1. Wenn bereits ein ausgearbeitetes Aktivitätsdiagramm existiert, fahre mit Schritt 11 fort.
2. Identifiziere aus den Informationsquellen die Aktionen, die von beteiligten Personen oder Systemen durchgeführt werden.
3. Zeichne sie gemäß Notation in das Diagramm ein.
4. Identifiziere aus den Informationsquellen die Objekte, die entweder als Voraussetzung für eine Aktion benötigt wird oder als Ergebnis einer Aktion entsteht.
5. Zeichne an die betreffenden Aktionen die Pins ein.
6. Sofern vorhanden, lege die Eingangs- und Ausgangsparameter fest und zeichne sie als Rechteck auf die Seitenumrandung der Aktivität.
7. Ordne die Aktionen nun nach Ihrer zeitlichen Abfolge an.
8. Dort, wo Aktionen parallel verlaufen, zeichne Splittingknoten ein und wo Aktionen wieder zusammengeführt werden, zeichne Synchronisierungsknoten ein.
9. Sofern Bedingungen für das Splitting oder die Synchronisierung erfüllt werden müssen, beschrifte die entsprechenden Knoten.
10. Zeichne anschließend Verzweigungsknoten dort ein, wo Alternativzweige entstehen. Beschrifte auch hier die Knoten mit den Bedingungen, die zu ihrem Durchlaufen führen.
11. Optional: Prüfe, ob externe Signale empfangen oder Signale nach extern gesendet werden müssen und setze die Signal-Elemente an die notwendige Stelle ein.
12. Optional: Markiere Aktionen, die zeitgesteuert sind.
13. Wenn noch keine Initial- und Endknoten im Aktivitätsdiagramm vorhanden sind, füge vor der ersten ausgeführten Aktion den Initialknoten und an der letzten Aktion den Endknoten ein; Setze ein Ablaufende bei jeder Verzweigung ein, sofern damit die Aktivität nicht beendet wird.
14. Prüfe, ob noch Verfeinerungen erforderlich sind. Wenn ja, fahre mit dem nächsten Schritt fort, wenn nein, springe an das Ende der Methodik.
15. Führe eine Verfeinerung durch
16. Beende die Erstellung des Aktivitätsdiagramms, wenn keine weiteren Informationen mehr in das Diagramm überführt werden können oder der gewünschte Detaillierungsgrad erreicht ist.

Die Eingangsbedingung existiert, damit nicht mehrere gleiche Aktivitätsdiagramme erstellt werden, die denselben Inhalt haben. Die optionalen Schritte zu den Signalen und den zeitgesteuerten Aktionen sind deshalb optional, da sie auch ohne ihre Angabe die Aktivität aus sich selbst heraus durchführen können und man nicht zwingend auf sie angewiesen ist. Die Integration von Initial- und Endknoten ist ein notwendiger Schritt, damit innerhalb einer Aktivität ein Anfangszustand und ein Endzustand vorliegt. Diese werden jedoch bei jeder Verfeinerung von Aktionen eine Ebene tiefer mitgeführt, so dass ihre Initialisierung nur einmal erfolgen muss.

Weitere Verfeinerungen sind für jede Aktion denkbar und auch durchführbar. Üblicherweise würde man aus Übersichtsgründen alle Verfeinerungen nicht innerhalb eines Aktivitätsdiagramms darstellen, sondern als eigene Diagramme aufführen.

6.3 Ableitung von Anwendungsfällen

Im vorherigen Kapitel 3 ist aufgezeigt worden, wie das V-Modell und der fundamentale Testprozess ineinander greifen können. Üblicherweise unterliegt die Entwicklung dem Prozess vom V-Modell und wird durch diverse Prozessschritte verfeinert. Der Vorteil des V-Modells liegt u.a. in der Abstraktion auf eine funktionale Anforderung und der schrittweisen Verfeinerung von einem Gesamtsystem hin zu einem konkreten Bauteil oder einer Komponente, wahlweise auch ein Artefakt, wenn es sich um softwaretechnische Umfänge handelt.

Aufgrund dieser Eigenschaften lassen sich die Anwendungsfälle und insbesondere ihre Reichweite innerhalb des Entwicklungsprozesses zuordnen. Daraus ergibt sich dann, um welches “subject under test” es sich handelt: Gesamtsystem, mehrere vernetzte Komponenten oder Einzelkomponenten. Dies hat dann auch Auswirkung auf die Reichweite der Simulationsumgebung. Das UML Use-Case Diagramm stellt dabei ein geeignetes Hilfsmittel dar, wie man entsprechende Schwerpunkte visualisieren kann. Mit Ihrer Hilfe lassen sich dann auch die Anwendungsfälle analysieren und auf eine Simulationsumgebung übertragen.

6.3.1 Ziel und Notation von Use-Case-Diagrammen

Das Use-Case-Diagramm ist Teil der UML 2.5 und hat zum Ziel, die funktionalen Aspekte eines Systems abzubilden und diese auch in der Interaktion mit seiner Umwelt darzustellen, welches insbesondere durch die Einfachheit der Notationselemente gelingt. Nach Jeckle et al. geht es vor allem darum, die Beziehungen und die Abhängigkeiten zwischen den Akteuren und den eigentlichen Anwendungsfällen aufzuzeigen. Dabei steht zunächst nicht die interne Struktur

des Systems im Vordergrund, sondern vielmehr die Schärfung der Funktionalität, damit eine zielgerichtete Umsetzung erfolgen kann. Daher wurde diese Diagrammart mittlerweile auch den Verhaltensdiagrammen zugeordnet. [BHW15, KS15, OMG15]

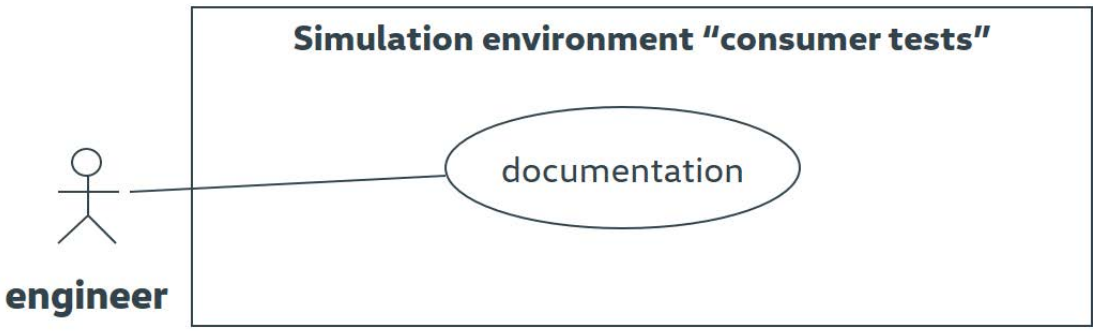


Abbildung 6.10: Beispiel für einen Use-Case

In der Abbildung 6.10 können exemplarisch einige Notationselemente identifiziert werden. Die ovalen Elemente beschreiben den eigentlichen Use-Case, also die Handlung, das Verhalten oder die Funktionalität, welche eine Eigenschaft des Systems beschreibt. Das sich darum befindliche Rechteck kennzeichnet das zugrundeliegende System und grenzt es zu seiner Umwelt, die nicht näher beschrieben ist, ab. Einzig die möglichen Akteure, hier als "Strich-Männchen" visualisiert, werden als Teil der Umwelt beschrieben. Dabei kann ein Akteur sowohl eine reale Person als auch ein weiteres, externes System darstellen. [BHW15, KS15, OMG15]

Abbildung 6.11: Akteure mit einer Ableitungsbeziehung

Die Verbindungslinien zwischen Akteur und Use-Case bilden eine Assoziation und beschreiben, dass der Akteur an dem Verhalten bzw. der Funktionalität beteiligt ist. Es gibt auch eine Ableitungsregel, die wie im Klassendiagramm durch einen nicht ausgefüllten Pfeil wie in Abbildung 6.11 illustriert wird. Durch diese Verknüpfung wird beschrieben, dass die beiden Arten von Akteuren eine Teilmenge gleicher Eigenschaften haben, sich jedoch in einigen Aspekten unterscheiden. [BHW15, KS15, OMG15]

Weitere Elemente zur Beschreibung von Anwendungsfällen sind die "include" und die "extends"

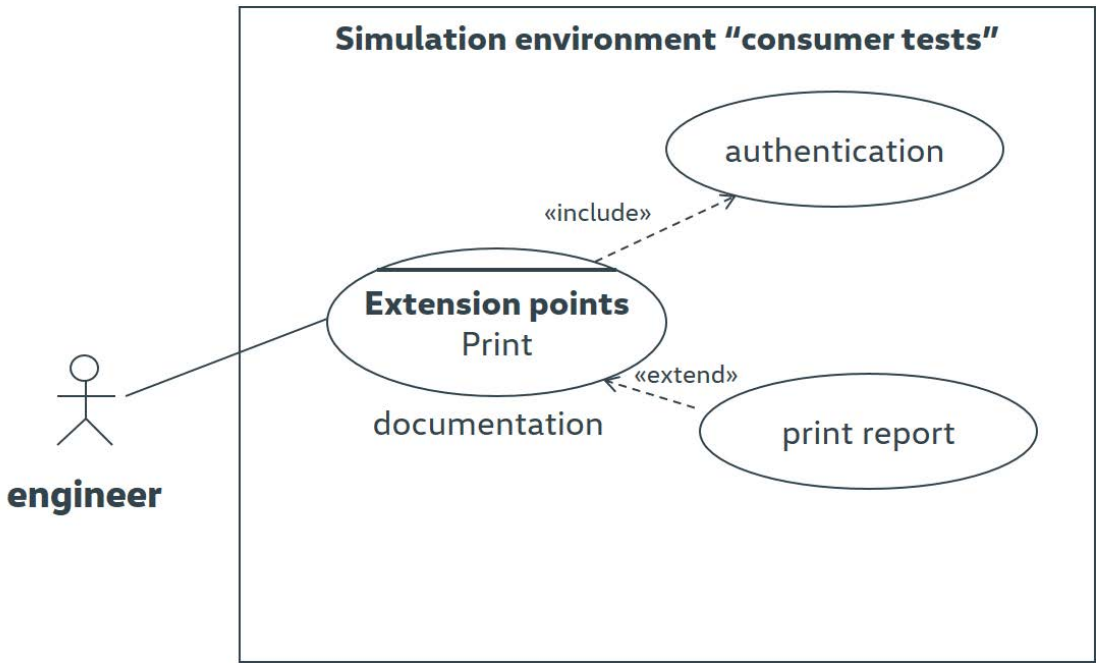


Abbildung 6.12: include und extend in einem Use-Case-Diagramm

Befehle aus Abbildung 6.12. Durch ersteren wird beschrieben, dass der Use-Case UC01 noch einen weiteren Use-Case UC02 umfasst, der für die Durchführung der Funktionalität erforderlich ist. Durch "extends" wird auch eine Verhaltenserweiterung beschrieben, welche jedoch nicht zwingend, sondern optional um die Funktionalität erweitert werden kann.

In der UML 2.5 lassen sich auch weitere Diagrammtypen wie Klassen, Komponenten- oder Zustandsdiagramme in Use-Case Diagramme integrieren. Dies soll hier aber nicht Gegenstand der Betrachtung sein. [BHW15, KS15, OMG15]

6.3.2 Die Methodik zu Use-Case Diagrammen

Zusammenfassend lässt sich die Methode zur Entwicklung eines Use-Case Diagramms für die Klärung der Aufgaben einer Simulationsumgebung innerhalb eines vorgegebenen Entwicklungsprozesses folgendermaßen beschreiben:

1. Identifiziere bisher beschreibende Quellen und Dokumentationen des Entwicklungsprozesses
2. Analysiere sie auf Stakeholder, beteiligte Systeme, (Zwischen-)Ergebnisse.

3. Wenn bei den Elementen die Möglichkeit existiert, gleiche Teilmengen an Eigenschaften zusammenzufassen, so füge eine Ableitung von Element A zu Element B ein. (Vererbung)
4. Füge die identifizierten Elemente hinsichtlich Stakeholder und Systeme entsprechend der Notation ein.
5. Beschreibe die Arbeitsergebnisse als Handlungen, so dass daraus ein Use-Case entsteht.
6. Verbinde die Elemente mit den Use-Cases über Assoziationen in der Art, dass ihre Beteiligung am Use-Case deutlich wird.
7. Wenn Abhängigkeiten zwischen Use-Cases in der Weise bestehen, dass Use-Case A vor oder nach Use-Case B durchgeführt wird, dann füge die Abhängigkeitsbeziehung ein.
8. Füge bei Bedarf Bedingungen an diese Abhängigkeitsbeziehungen an, wenn der Eintrittsfall beschränkt ist.
9. Wenn der Bedarf nach Detaillierung der Use-Cases als extension point besteht, erweitere um zusätzliche Use-Cases und verbinde sie mit dem extension point
10. Wiederhole Schritt 9, bis der gewünschte Detaillierungsgrad erreicht ist.
11. Prüfe, ob eine Zuordnung der Use-Cases innerhalb des Entwicklungsprozesses erforderlich ist. Falls ja, führe die Methode “Zuordnung im Entwicklungsprozess” aus.
12. Wiederhole ab Schritt 1 in regelmäßigen Zeitintervallen zur Überprüfung der Erweiterbarkeit des Simulationssystems.

Schritt 1 umfasst die Sammlung der Informationsquellen, wobei die oben genannte Aufzählung einen Überblick über die Dokumentarten verschafft. Die Analyse der Dokumente auf beteiligte Stakeholder und die bisherige Einbindung weiterer Systeme erfolgt in Schritt 2. Dazu gehört auch die Bestimmung der Ergebnisse, die bisher entweder vom jeweiligen Stakeholder bzw. vom beteiligten System erwartet werden. In Schritt 3 kann dann optional nach übereinstimmenden Merkmalen und Eigenschaften der identifizierten Elemente wie Akteur oder System gesucht werden, um Standardisierungen durch Ableitungen analog zu Klassendiagrammen zu erreichen. Schritt 4 überführt die identifizierten Objekte dann in der entsprechenden Notation in das Use-Case Diagramm.

Schritt 5 ist der entscheidende Punkt: Hier müssen die Ergebnisse aus dem eigentlichen Entwicklungsprozess bewertet und als Handlung zu ihrer Erreichung verallgemeinert werden. Dafür kann es auch von Bedeutung sein, welche Eingangsgrößen zum Ergebnis führen werden. Dabei ist die Transformation von Eingangsgrößen zu Ausgangsgrößen zunächst nur abstrakt denkbar, also nicht im Detail bekannt. Auch die genaue Anzahl der Eingangsgrößen muss noch

nicht genau definiert sein. Vielmehr muss ersichtlich werden, was als Handlung vollzogen werden soll. Denn die Simulation kann zunächst auch als Mittel dazu dienen, diese Unklarheiten in Form eines explorativen Ansatzes herauszuarbeiten. Gleichwohl kann die Simulation eine Abbildung des Entwicklungsprozesses sein, in dem alle Schritte und Berechnungen bekannt sind. Aber auch hier lebt das Diagramm von seiner Schlichtheit, damit der spätere Einsatzzweck der Simulationsumgebung im Blick behalten wird. Schritt 6, 7 und 8 verbinden die Handlungen mit konkreten Personen, beteiligten Systemen und weiteren Anwendungsfällen, wobei ergänzende Bedingungen die möglichen Aktionen zeitlich oder kausal strukturieren. Durch die Schritte 9 und 10 können weitere Strukturierungen vorangetrieben werden, so dass eine verbesserte Sicht auf das Gesamtsystem ermöglicht wird. Der letzte Schritt erlaubt über den Lebenszyklus des Systems ein Review der Zielsetzung und eine Überprüfung der Erweiterbarkeit, um evtl. neu auftretende Fragestellungen, die sich über die bisherigen Anwendungsfälle nicht abbilden lassen, zu integrieren. Ein Beispiel dazu folgt im nachfolgenden Abschnitt im Rahmen der Fallstudie.

6.3.3 Zusammenführung und Zuordnung der Aufgaben innerhalb des Entwicklungsprozess

Wenn mit Hilfe der Use-Case-Diagramme insbesondere die Systemfunktionalität in Form der Anwendungsfälle und ihre Interaktion mit der Umwelt sowie ihren Akteuren beschrieben werden kann, ohne tief in die innere Struktur des Systems einzutauchen, dann lässt sich aus diesem Artefakt eine jeweilige Zuordnung innerhalb des V-Modells erzielen. Da jeder Use-Case einen bestimmten Kontext innerhalb des Entwicklungsprozesses umfasst, kann auch eine konkrete Zuordnung auf die jeweilige Abstraktionsebene vorgenommen werden.

Als ausschlaggebende Kriterien für die Zuordnung sind somit wie folgt heranzuziehen:

- menschlicher Akteur in Abhängigkeit seiner Projektrolle, der an einem bestimmten Ergebnis interessiert ist.
- Integrationsstufe des “subject under test” (SUT)
- SUT: Hardware oder Software

Üblicherweise ist innerhalb eines Projektes definiert, welcher Projektteilnehmer welche Rolle ausfüllt. Beispielsweise kann dies der verantwortliche Entwicklungsingenieur oder aber auch der davon losgelöste Tester. Darüber hinaus können hier jedoch weitere Kriterien unterschieden werden, die z.B. die Abstraktion nach Funktion, nach System oder nach Bauteil erlaubt. Auch eine Aufteilung nach Hard- und Software kann eine weitere Einordnungshilfe darstellen.

Daraus kann folgende Methodik für die Zuordnung im Entwicklungsprozess formuliert werden:

1. Erstelle eine Matrix, die auf der einen Achse die Ausprägungen der einzelnen Integrationsstufen des SUT aufweist.
2. Ergänze auf der zweiten Achse die verschiedenen Rollen der Akteure, für welche das Ergebnis aus dem Use-Case relevant ist.
3. Auf der dritten Achse kann die Unterscheidung nach Hard- oder Software erfolgen.
4. Indexiere alle identifizierten Use-Cases und trage in das Feld den Index des Use-Cases ein, zu dessen Kriterien die Zuordnung erfolgen kann.
5. Wähle eine Visualisierung für das V-Modell aus und übertrage die Matrix in diese Grafik. (vgl. Abb. 6.15)

Am Ende dieses Teilprozesses ergeben sich nun zwei Artefakte: Einmal das Use-Case Diagramm mit den Aufgaben der Simulationsumgebung in seiner Gesamtheit für eine Problemstellung und zum anderen eine Visualisierung des V-Modells mit entsprechender Zuordnung der Aufgaben im Entwicklungsprozess. Insbesondere die Zuordnung der Anwendungsfälle innerhalb des V-Modells liefert eine Orientierung für eine zielgerichtete Anpassung der Simulationsumgebung an den Entwicklungsprozess.

6.4 Entwicklung von Fragestellungen als Teil der Aufgabenbewältigung

Nachdem aufgezeigt wurde, wie die Anforderungen in Aufgaben für die Simulationsumgebung und gleichzeitig innerhalb eines Entwicklungsprozesses adressiert werden können, wird auf die Aufgaben selbst der Fokus gelegt. Dabei interessiert speziell, welche konkreten Fragestellungen innerhalb der Aufgaben untersucht werden soll. Das Ziel dieses Schrittes ist es, entweder das bisherige implizite Wissen über das System oder die Funktion explizit zu transformieren oder noch nicht verfügbares Wissen darüber zu gewinnen. Die Formulierung der Fragen in Textform bietet zunächst den Vorteil einer pragmatischen Herangehensweise insbesondere bei noch (in Teilen) unbekanntem Systemverhalten.

Es stellt sich aber die Herausforderung, wie solche Fragen effektiv herausgearbeitet werden können, also ob es eine Möglichkeit gibt, eine Methodik zu entwickeln, mit deren Hilfe die Ableitung von Fragestellungen standardisiert werden kann, insbesondere wenn der zugrundeliegende Kontext für die Fragestellung variiert.

An dieser Stelle der Gesamtmethodik wird zunächst der Schwerpunkt auf die Eingrenzung auf ein bestimmtes Verhalten der Funktion oder des Systems gelegt. Dadurch ist es möglich, zunächst ohne internes Wissen über die Funktionsweise offene Fragen über vorangegangene reale Experimente bzw. Fahrversuche zu entwickeln; dies ist als Annäherung zu verstehen. In Kapitel 7 wird dann die Ausarbeitung von Zielgrößen erläutert, welche die Antwortmöglichkeiten definiert und konkretisiert. Die Zielgrößen setzen jedoch ein gewisses Maß an internem Wissen über Funktion oder System voraus.

Vor diesem Hintergrund lässt sich folgende Methodik formulieren, die an dieser Stelle vergleichsweise abstrakt gehalten werden kann, da aufgrund der fehlenden konkreten Rahmenbedingungen Entscheidungskriterien nur eingeschränkt verallgemeinernd berücksichtigt werden können:

1. Sofern das Systemverhalten schon bekannt ist, fahre mit Schritt 5 fort, sonst
2. Identifiziere das Systemverhalten $S(g)$ anhand von zugehörigen Lastenheften oder Testfallkatalogen.
3. Wenn das Systemverhalten $S(g)$ für die Regelfälle bekannt ist, fahre mit Schritt 5 fort, sonst
4. Führe eine Kampagne mit Experimenten durch, sodass das Regelverhalten des Systems oder Funktion bestimmt werden kann.
5. Beschreibe das Systemverhalten oder das Verhalten der Funktion in einfacher Form (z.B. in Textform, grafisch als Szenarien oder als Mindmap).
6. Wähle ein (Teil-)Verhalten $T(r)$ als Referenzverhalten für die weitere Betrachtung aus, das noch nicht ausgewählt wurde.
7. Prüfe, ob weitere Verhaltensweisen unter den Randbedingungen des Referenzverhaltens $T(r)$ denkbar und relevant sind.
8. Formuliere diese Verhaltensweisen als offene Fragen.
9. Ordne Sie $T(r)$ zu
10. Wiederhole ab Schritt 6, bis alle zu betrachtenden Verhaltensweisen adressiert wurden.
11. Wiederhole ab Schritt 5, bis alle zu untersuchenden Teilverhalten Fragen erhalten haben.

Zunächst ist es wichtig zu prüfen, ob es bereits vorhandenes Wissen über das System oder die Funktion existiert. Dabei ist es unabhängig davon, ob es explizit in Form von Lastenheften oder

Testfallkatalogen oder implizit durch Fachwissen im Entwicklungsteam vorliegt. Explizites Wissen liegt meist dann vor, wenn der Entwicklungsprozess eines Systems oder Funktion auf die Serienentwicklung ausgerichtet ist. Das implizite Wissen wird in einer viel früheren Phase als während der Phase der Serienentwicklung vorhanden sein, insbesondere wenn noch nach neuen Funktionen oder Systemen geforscht wird. Ist eine konkrete Dokumentation zunächst nicht vorhanden, so können zunächst Fragen formuliert werden, die das Regelverhalten zum Gegenstand haben. Wenn bereits das Regelverhalten bekannt und dokumentiert ist, können Randerscheinungen und Grenzfälle näher unter die Lupe genommen werden, so dass Fragen für deren Untersuchung herausgearbeitet werden. Insbesondere in solchen Fällen, die nicht häufig auftreten, sind gerade Qualitätsverbesserungen erzielbar, da der Aufwand Fehlerwirkungen ausfindig zu machen, wesentlich höher ist als bei den Regelfällen. Ihr Auftreten im Feld führt häufig zur Unzufriedenheit beim Kunden, denn ein fehlerhaftes Regelverhalten würde ohnehin nicht zu einer Serienfreigabe führen.

Für den unwahrscheinlichen Fall, dass noch kein Verständnis vom Systemverhalten vorliegt, kann eine Kampagne mit realen Experimenten durchgeführt werden. Dies ist deswegen unwahrscheinlich, da es im industriellen Kontext zumindest bei Entwicklungsvorhaben mit anderen Projektpartnern eine Vorstellung davon gibt, was das zu entwickelnde System später leisten soll. Dennoch kann der Fall eintreten, dass z.B. ein Wettbewerber über eine Funktion verfügt, über die man Erkenntnisse gewinnen möchte. Aus Gründen der Einfachheit kann das Verhalten beispielsweise in Textform, grafisch als Bilderstory oder auch als Mindmap beschrieben werden.

Da ein System in der Regel nicht nur auf eine Weise reagiert, verhält es sich in Folge unterschiedlicher Randbedingungen auch in unterschiedlicher Weise. Daher ist es sinnvoll, sich zunächst nur einen Teil des Verhaltens genauer anzuschauen und dort nach Randerscheinungen zu suchen. Diese Potentiale werden dann als offen formulierte Frage aufgeführt.

Die Fragen, die also aufgrund der Analyse der Systemränder identifiziert wurden, werden dann dem jeweiligen Teilverhalten zugeordnet. Anschließend wiederholt man die Vorgehensweise für jedes spezifische Teilverhalten des Systems oder der Funktion, bis das Gesamtverhalten erschlossen wurde.

Als letztes Element zur Analyse hat man den Fragenkatalog in Form des drittes Artefakt erhalten, so dass der zweite Prozessschritt zur Methodik für die Qualitätssicherung durchgeführt werden kann: Die Entwicklung der Infrastruktur. Das nachfolgende Fallbeispiel zeigt die Anwendung der vorgestellten Methodik.

6.5 Beispiel Consumer Tests

Im Nachfolgenden soll nun die Methode anhand eines konkreten Fallbeispiels in der Praxis hinsichtlich des ersten Prozessschrittes angewendet werden. Dabei wird das Fallbeispiel aus der Funktionsentwicklung für Aktive Sicherheitssysteme herausgearbeitet. Wie eingangs Kapitel 6 erwähnt, muss allerdings bzgl. des vollständigen Entwicklungsprozesses innerhalb der Volkswagen AG aus Gründen der Vertraulichkeit abstrahiert werden.

6.5.1 Aktivitätsdiagramm zu Consumer Tests

In Kapitel 2 wurde u.a. die funktionsorientierte Entwicklung mit einer entsprechenden Verfeinerung der Beschreibung von Funktion über System und letztlich Bauteil in der Abbildung 2.5 illustriert.

Daraus lässt sich nun ein Prozess generieren, der im vorliegenden Fall in abstrakter Variante dargestellt wird und grafisch in Abbildung 6.13 zusammengefasst ist. Dieser zeigt den Weg der Entwicklung einer aktiven Sicherheitsfunktion über die verschiedenen Ebenen entlang am V-Modell.

Auf den vier Ebenen der V-Modells wurden dazu die Aktivitäten aufgeschlüsselt, die üblicherweise durchgeführt werden. Dabei wurde auch berücksichtigt, dass eine testgetriebene Entwicklung angestrebt wird. Dies wurde bereits in Abbildung 2.3 veranschaulicht, indem der fundamentale Testprozess in das V-Modell integriert wurde. Hier wird nun auf jeder Abstraktionsebene neben den üblichen Prozessschritten bereits mit der Testfallerstellung und den vorbereitenden Tätigkeiten mit dem Ziel begonnen, erste Absicherungen der Implementierung vorzunehmen und die Anforderungen zu erfüllen.

Im folgenden sei die Abbildung 6.13 kurz beschrieben: Für eine Funktion wie z.B. eine Aktive Sicherheitsfunktion werden zunächst die Anforderungen definiert. Dazu zählt zum einen die Funktionsbeschreibung aber auch zum anderen das Funktionslastenheft. Bei wiederkehrenden Funktionen, die in mehreren Fahrzeugmodellen zum Einsatz kommen, werden die Artefakte aus der Vorgängergeneration des Fahrzeugs durchaus wiederverwendet. Daraus abgeleitet wird anhand der Anforderung idealerweise gleich eine Testspezifikation parallel entwickelt, um die Anforderung sogleich abgesichert zu haben. Dies ist ein paralleler Pfad zur absteigenden Flanke des V, der in den gegenüberliegenden Teil mündet.

Danach wird mit dem funktionalen Systementwurf begonnen, damit eine erste Zuordnung der Funktionen zu (Teil-)Systemen möglich wird. Gerade im Fahrzeug übernehmen Steuergeräte

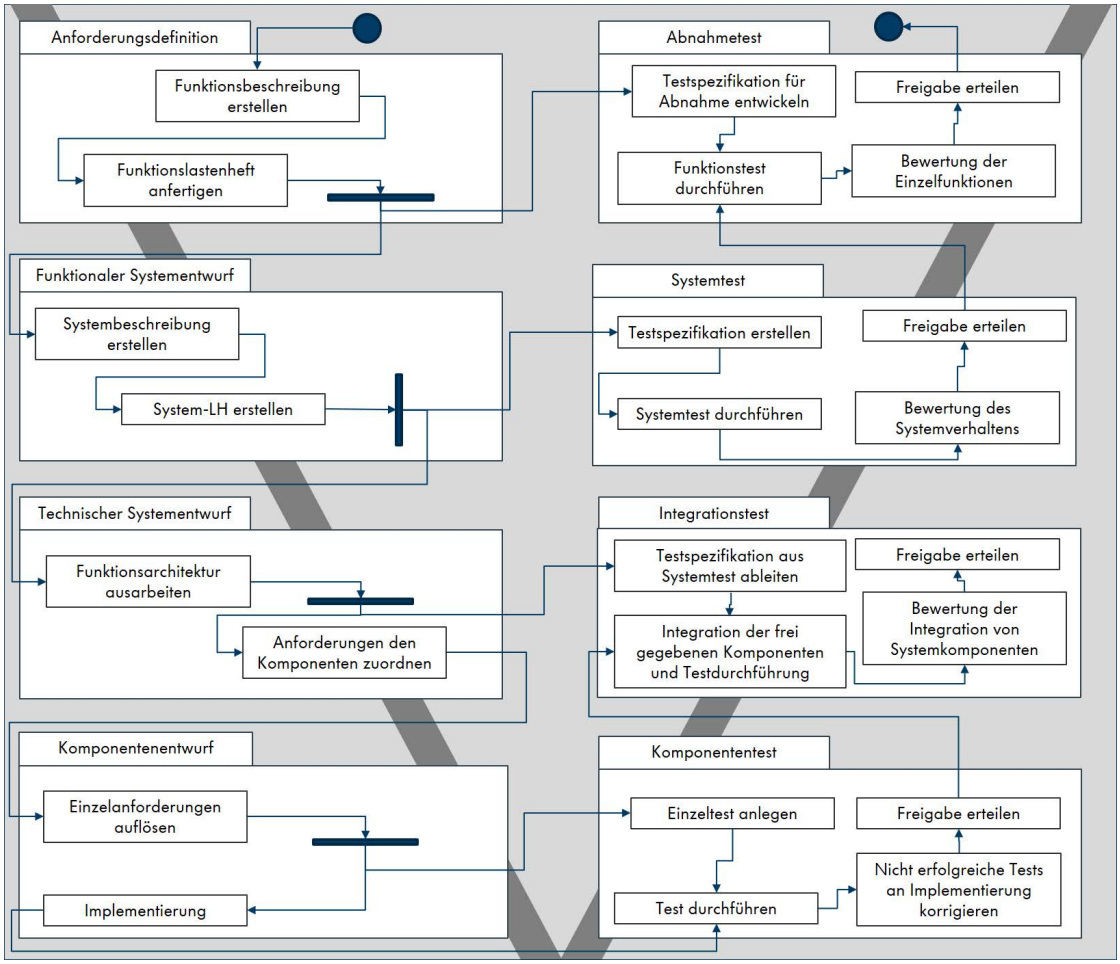


Abbildung 6.13: Aktivitätsdiagramm zum Entwicklungsprozess einer Aktiven Sicherheitsfunktion

durchaus mehrere Funktionen, die in ihrer Art ähnlich sind bzw. deren Daten für andere Funktionen wieder notwendig sind. Dies wird dann in einer Funktionsarchitektur zusammengefasst, welches dem technischen Systementwurf entspricht. Beide Ebenen weisen jeweils einen Abzweig zur Erstellung einer Testspezifikation auf. Es ist zielführend, diese losgelöst von der Funktionsebene zu erstellen, da zum einen die Entkopplung die Wahrscheinlichkeit erhöht, die Anforderungen anders abzusichern, so dass unter Umständen evtl. weitere Fehler bei jenen Anforderungen gefunden werden und zum anderen ein anderer Fokus aufgrund des unterschiedlichen Abstraktionscharakters ins Blickfeld rückt.

Auf der untersten Ebene werden dann die vorherigen Anforderungen und Detaillierungen auf die einzelnen Systemkomponenten verteilt und entsprechend implementiert.

Darauffolgend wird dann anhand der aufsteigenden Flanke im V-Modell die Testdurchführung und Auswertung vollzogen. Ausgehend vom Einzeltest einer Anforderung werden dann diese gebündelt durchgeführt und anschließend bewertet. Die nicht erfolgreichen Tests werden wieder zurück an die Implementierung gespiegelt. Da Test und Implementierung in unterschiedlichen Händen und Rollen liegen, ist die Wahrscheinlichkeit hoch, dass Fehlerwirkungen nicht durch ein detailliertes Wissen über den Quellcode verdeckt werden. Anhand projektspezifischer Kriterien können dann entsprechende Freigaben erteilt werden, so dass die bis dahin auf Komponenten-Ebene entwickelten Artefakte in die technische Systemebene zum Integrationstest übergeben werden können.

Beim Integrationstest werden nun die Komponenten auf ihr Zusammenspiel getestet und anschließend bewertet. Beim jeweiligen Testerfolg wird für die nächste Ebene der Systemtest freigegeben. Hier sind dann alle Komponenten vorliegend und fügen sich bereits in die Gesamtarchitektur ein. Am Ende wird bei den Funktionstest das fertige System im Kundenkontext, dem späteren Anwender, betrachtet und getestet sowie mit dem Schwerpunkt der Abnahme versehen. Dabei spielen technische Details zur Implementierung eine untergeordnete Rolle, so dass alleinig das Funktionsverhalten des Systems entscheidend ist.

Dieser noch vorherrschende, klassische Ansatz der Entwicklung von Fahrzeugfunktionen wie z.B. auch Aktiven Sicherheitsfunktionen liegt ein Prozess zu Grunde, der die Betrachtung von mehreren, parallel verlaufenden Fahrzeugprojekten und einer Wiederholung von Arbeitsschritten und Tätigkeiten berücksichtigt. Daher sind agile Ansätze der Softwareentwicklung wie SCRUM oder andere durchaus denkbar, jedoch noch nicht etabliert. Wegen der hohen Komplexität der Fahrzeugentwicklung ist auch eine Einführung ein recht großes Wagnis, welches bisher noch nicht eingegangen wurde. Zudem ist eine agile Entwicklung wie SCRUM erst geeignet, wenn es sich um nicht standardisierte Entwicklungen handelt und man sich auf iterativer Basis mit kurzen Zeiträumen für Entwicklungsergebnisse an die spätere Vision annähern muss.

Auf dieser Basis eines Aktivitätsdiagramms können nun weitere Überlegungen zur Identifikation von Anwendungsfällen gemacht werden. Hier ist entscheidend, dass für eine geeignete Simulation entsprechende Einsatzszenarien identifiziert werden, um eine geeignete Zielsetzung für die Simulationsumgebung zu erhalten und damit einen Anhaltspunkt für einen späteren Abstraktionsgrad bei der Modellierung zu bekommen. Im nachfolgenden Abschnitt wird daher auf die Anwendungsfälle konkret eingegangen.

6.5.2 Anwendungsfälle im Rahmen der Entwicklung von Aktiven Sicherheitssystemen

Das unter Abbildung 6.13 dargestellte Diagramm beschreibt auf eine abstrakte Weise, wie eine aktive Sicherheitsfunktion entwickelt werden kann. Dabei erfolgen weitere Schritte innerhalb dieser übergeordneten Handlungen, die nachfolgend zu einem Aktivitätsdiagramm (vgl. Abb. 6.14) geführt haben.

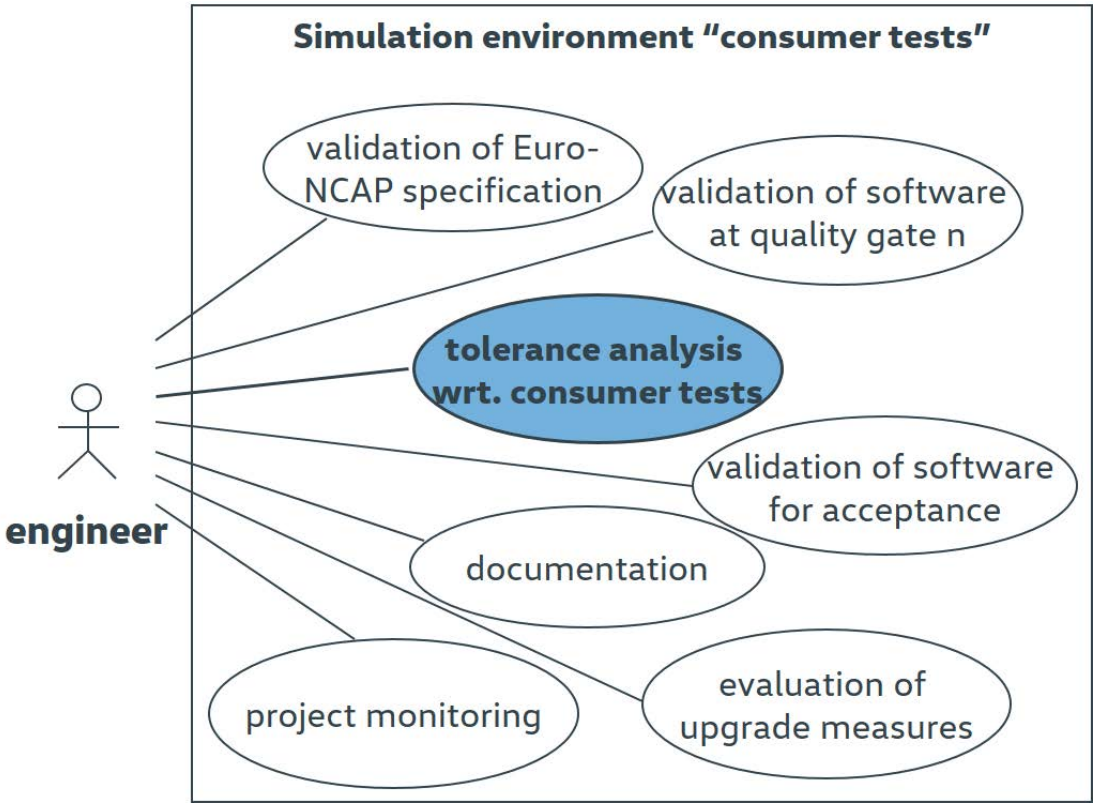


Abbildung 6.14: Uses Cases für Consumer Test Simulation

- Validierung von Consumer Test Spezifikationen
- Toleranzanalyse im Rahmen von Consumer Tests
- Validierung der Software zur Qualitätsstufe n
- Validierung der Software für die Akzeptanz
- Projektstatusverfolgung

- Evaluation von Funktionserweiterungen
- Dokumentation

Die Validierung von Consumer Tests Spezifikation ist insbesondere während ihrer Entstehung auf ihre Zweckmäßigkeit zu prüfen. Die Toleranzanalyse der konkreten Fahrszenarien eines Systems, wie es sich unter der Ausnutzung der gesamten Toleranzbereiche verhält, ist ein zweiter Anwendungsfall, der für die Entwicklung identifiziert werden kann. Die Validierung der Software zu einer Qualitätsstufe n bzw. zur späteren Akzeptanz bilden weitere Anwendungsfälle. Dabei liegt im Fokus, dass eine (Software-)Komponente zu einem bestimmten Meilenstein oder einem anderen vereinbarten Zeitpunkt eine gewisse Qualität erreicht haben sollte oder muss. Unter Qualität ist hier zu verstehen, dass neben der implementierten Funktionalität auch die Zweckmäßigkeit der Funktion erfüllt wird. Die Akzeptanz ist dann am Ende des Entwicklungsprozesses nochmal ein gesonderter Anwendungsfall, da 100% der Umfänge vorhanden und funktional zweckmäßig sein müssen. Innerhalb zweier offizieller Qualitätsstufen können weitere Untersuchungen zur (Software-)Komponente vorgenommen werden, wobei dann neue und über die offiziell vereinbarten Testfälle und -kriterien hinaus weitere Untersuchungen stattfinden können, um Aufschluss über die Qualität zu erhalten. Dies erlaubt dann eine intensivere und gezielte Projektverfolgung mit einem frühzeitigem Gegensteuern beim Lieferanten. Die Simulation soll aber auch die Möglichkeit bieten, dass neue Funktionen bzw. Funktionserweiterungen simulativ evaluiert werden können, wenn beispielsweise ein neuer Algorithmus vom Lieferanten ohne eine vorheriges Lastenheft oder konkrete Spezifikation dem Auftraggeber zur Verfügung gestellt wird. Als letzten Anwendungsfall soll die Simulationsumgebung auch die erzielten Ergebnisse hinreichend dokumentieren.

Im Folgenden wird der Use-Case für die Toleranzanalyse in Consumer-Test Szenarien für den weiteren Verlauf der Arbeit ausgewählt. Dies ist dadurch begründet, dass speziell die Toleranzbetrachtungen in Szenarien bisher keine Rolle in der Entwicklung von Aktiven Sicherheitssystemen gespielt hat. Auch erlaubt die tiefere Betrachtung dieses Anwendungsfalls eine umfassende Erprobung der hier vorgestellten Methodik zur Qualitätssicherung.

6.5.3 Integration der Anwendungsfälle in das V-Modell

Es ist nun aufgezeigt worden, dass aus einem V-Modell als allgemeine Grundannahme für eine Entwicklung im Automobilbau die jeweiligen Aktivitäten auf den einzelnen Ebenen im Rahmen eines Aktivitätsdiagramms in eine Beziehung bzw. Abfolge gebracht werden können, um eine Vergegenwärtigung des aktuellen Prozesses zu erreichen. Dieses Ergebnis dient dann wiederum

als Grundlage für die Identifikation von Anwendungsfällen, die für die Zielsetzung einer Simulationsumgebung notwendig ist. Nun kann im Umkehrschluss eine Zuordnung dieser Use-Cases in das V-Modell erfolgen, so dass sich eine Gesamtdarstellung ergibt, welche als Ausgangslage bzw. als Artefakt für weitere softwaretechnische Methoden dienen kann. Dabei ist zu beachten, dass Anwendungsfälle auf mehreren Abstraktionsebenen angesiedelt werden können, da sie sich entweder nur auf ein einzelnes Bauteil oder Komponente beziehen oder auf ganze (Teil-)Systeme erstrecken können. An dieser Stelle wird dann ein Index angeführt: F für Funktion, S für System und B für Bauteil.

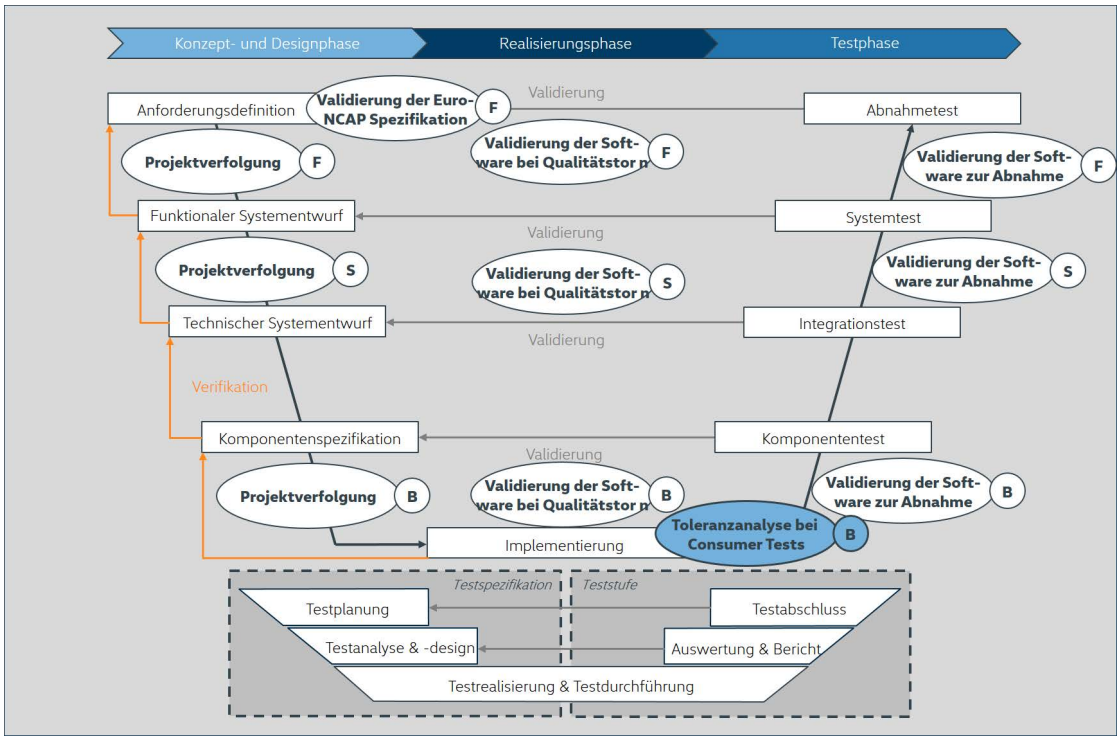


Abbildung 6.15: Uses Cases innerhalb des V-Modells

In der vorliegenden Fallstudie sind in der Abbildung 6.14 mögliche Anwendungsfälle dargestellt. In der Abbildung 6.15 wurden nun jene in das V-Modell integriert. Es gibt einige Anwendungsfälle, die auf allen Abstraktionsebenen des V-Modells relevant sind und dort angewendet werden können. Dazu gehören bspw. “Projektverfolgung” oder “Validierung der Software bei Qualitätstor n”. Ihre Konkretisierung ergibt sich aus der jeweiligen Spezifikation auf der zugehörigen Ebene; sie werden über den Index F, S oder B innerhalb des V-Modells unterschieden. Einige Fälle jedoch bleiben nur auf einer konkreten Ebene verortet, da das gewünschte Ergebnis nur in dem jeweiligen Kontext von Bedeutung ist wie z.B. bei “Validierung der EuroNCAP

Spezifikation", bei dem es sich um die grundlegende Überprüfung der Spezifikation bzw. die Feststellung einer Änderung sowie deren Auswirkungen auf eine Funktion handelt. Die Besonderheit dieser Spezifikation besteht darin, dass sie von einer externen Stelle definiert wird und daher einer internen Bewertung unabhängig vom eigentlichen Entwicklungsprozess unterliegt. Das Ergebnis hat dabei zwar grundsätzlich die Reichweite auch auf die unteren Ebenen im V-Modell, allerdings dient es primär auf der obersten Ebene für weiterführende Arbeitsschritte wie z.B. bei der Auslegung der Funktion oder auch einer möglichen Einflussnahme auf eine Änderung der Spezifikation.

Der Use-Case "Toleranzanalyse bei Consumer Tests" kann hier als eine Instanz des Anwendungsfalls "Validierung der Software bei Qualitätstor n" angesehen werden. Aus welchen Gründen eine solche Toleranzanalyse im Umfeld von Consumer Tests durchgeführt werden sollte, kann im letzten Abschnitt dieses Kapitels nachvollzogen werden.

6.5.4 Der Fragenkatalog zur Toleranzanalyse von Consumer Tests

Bereits in [BHK+14] wurde die Motivation für eine Toleranzanalyse aufgezeigt. An dieser Stelle sei in detaillierterer Art und Weise auf den Sachverhalt eingegangen.

Hintergrund zur Thematik besteht in den zulässigen Abweichungen von einer idealen Fahrt eines Fahrzeugs in den jeweiligen Testszenarien der Consumer Tests. Diese Toleranzen sind in den jeweiligen Spezifikationen der Consumer Test Behörden definiert und beziehen sich in der Regel auf die Anfahrtsgeschwindigkeit des Fahrzeugs, der Abweichung von der idealen Anfahrt auf ein Ziel, auf den Lenkwinkel über die Dauer der Anfahrt aber auch auf das Verhalten des Zielfahrzeugs und weitere Parameter.

Je nach Position und Zustand des Vehilce-Under-Test (VUT) bzw. auch des Zielfahrzeugs können sich je nach Implementierung des Ausweichalgorithmus ein unterschiedliches Verhalten ergeben. Die Abbildung 6.16 verdeutlicht grundsätzlich diesen Zusammenhang bei einem Testszenario mit zwei Fahrzeugen.

In der Literatur wird in der Regel bei klassischen Methoden ein Äquivalenzklassentest als essentiell betrachtet. [SL07] Dieser sieht vor, dass ein Repräsentant für eine Menge an möglichen Eingangs- bzw. Testdaten ausgewählt wird und dieser Vektor stellvertretend für alle anderen möglichen Testvektoren ist, die ein gleiches Verhalten einer zu testenden Komponente oder Funktion erwarten lassen. Das Ziel dabei ist es, die Testanzahl auf das Notwendige zu reduzieren.

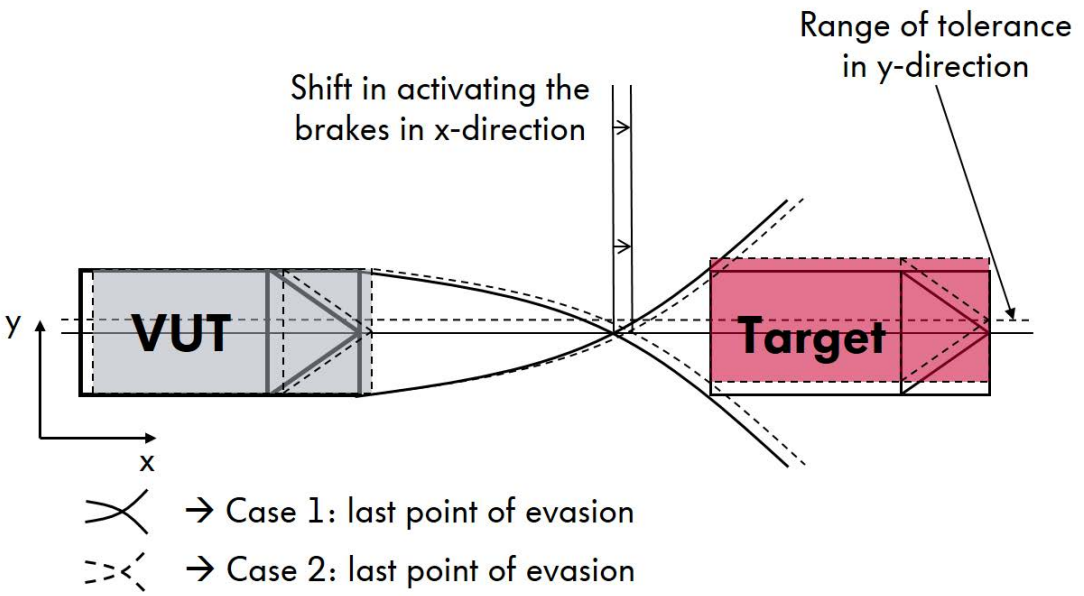


Abbildung 6.16: Schema zur Auswirkungen von Toleranzen auf ein EuroNCAP Testszenario (siehe auch [BHK+14])

Für den vorliegenden Fall würde es zum Beispiel bedeuten, dass man einen Parametervektor wählt, der eine Anfahrt auf das Testfahrzeug beschreibt. Allerdings kann man in der Abbildung 6.16 erkennen, dass bereits eine leichte Versetzung des Zielfahrzeugs von der Ideallinie eine direkte Auswirkung auf einen Auslösealgorithmus hat, der u.a. die Ausweichmöglichkeiten eines Fahrers auf ein Hindernis berücksichtigt. So führt diese Änderung dazu, dass die Ausweichmöglichkeit für den Algorithmus auf einer Seite länger bestehen bleibt und sich der Abstand zwischen VUT und Zielfahrzeug verringert. Für einen EuroNCAP Test bedeutet dies, dass sich damit die Restgeschwindigkeit erhöht, mit der das VUT auf das Zielfahrzeug trifft. Eine höhere Restgeschwindigkeit führt damit auch potentiell zu einer geringeren Punktzahl im konkreten Testfall. Um nun speziell bei Consumer Tests ein solches Verhalten untersuchen und absichern zu können, ist eine Toleranzanalyse erforderlich, die mit Hilfe der Äquivalenzklassenbildung als Qualitätssicherungsmethode nur unzureichend möglich ist.

Für eine Toleranzanalyse schließen sich dann eine Reihe von möglichen Untersuchungsfragen an, wie z.B.

- Wie wirkt sich eine max. Abweichung des VUT auf die Restgeschwindigkeit aus?
- Welche Auswirkungen hat eine Varianz von zulässigen Lenkwinkeln in der Anfahrt auf ein Hindernis?

- Welche Konsequenzen für das Testergebnis hat eine Varianz des Zielfahrzeugs innerhalb der Toleranzen.

Weitere Forschungsfragen wurden auch im Zusammenhang mit den Beiträgen [BBH+14a, BBH+14b, BBH+15b, BBH+15a] beantwortet.

6.6 Zusammenfassung

Die vorgestellte Methodik zur Entwicklung von Simulationsumgebungen für aktive Sicherheitssystemen im Kontext von Consumer Tests beginnt in diesem Kapitel mit einem umfangreichen Programm an Untersuchungen zum Entwicklungsprozess von Aktiven Sicherheitssystemen im Fahrzeug. Da für eine Simulationsumgebung der Einsatzzweck eindeutig bekannt sein muss, um zielgerichtet später die Umgebung verwenden zu können, ist ein Portfolio an weiteren Methodiken zusammengestellt worden, die der Erfassung und Strukturierung von Entwicklungsschritten dient. Das beginnt mit einer Erfassung der Aufgaben im Entwicklungsprozess, die hier aus internen Dokumenten und Aufzeichnungen identifiziert werden können. Danach wird mit der Visualisierung von Prozessen begonnen, wobei hier Aktivitätsdiagramme aus der UML als geeignetes Mittel zur Strukturierung der Prozesse angesehen wird. Daraus lassen sich wiederum Anwendungsfälle ableiten, ebenfalls ein Instrument der UML, so dass eine Aussage über die Aufgaben der späteren Simulationsumgebung getroffen werden kann. Hier ist es möglich, eine Zuordnung jener Aufgaben in das V-Modell vorzunehmen. Abschließend werden die jeweiligen (Forschungs-)Fragen als Teil der Aufgabenstellung der Simulationsumgebung abgeleitet. Diese ergeben sich aus der konkreten Situation des Systemkontextes und beschreiben, was am Ende für ein Ergebnis vorliegen soll.

7 Entwicklung einer adäquaten Simulationsinfrastruktur

In Kapitel 6 wurde aufgezeigt, wie die zu Anfang wichtige Analyse zielgerichtet durchgeführt werden kann, um die notwendigen Informationen für die Ausrichtung der Simulation zu erhalten. In diesem Kapitel liegt der Fokus darauf, eine geeignete Infrastruktur für die Simulation zu entwerfen und zu implementieren.

Dafür werden zunächst im ersten Schritt die Fragestellungen aus Kapitel 6 herangezogen und die Zielgrößen für die einzelnen Fragen herausgearbeitet. Anschließend erfolgt die Entwicklung des grundlegenden Konzeptes zum Aufbau der Simulationsumgebung: Welchen Detailgrad erhalten die Modelle, in welcher Umgebung werden sie integriert, wie werden die Simulationsdaten erzeugt, wie werden sie ausgewertet. Das Kapitel schließt mit der Fortsetzung der Fallstudie aus Kapitel 6.

7.1 Identifikation von Zielgrößen

Das Ziel der in dieser Arbeit vorgestellten Methodik in seiner Gesamtheit zielt auf die Steigerung der Softwarequalität durch die Identifikation von Fehlerwirkungen und damit die Möglichkeit der Beseitigung von Fehlern. Daraus ergibt sich, dass Fehlerwirkungen meist im Zusammenhang mit der Verarbeitung von Eingangsgrößen und deren Verdichtung auf wenige Ausgangsgrößen auftreten. Daher können Fehler zum einen durch falsche Berechnung entstehen, aber zum anderen durch eine falsche Parametrierung, so dass die Berechnung nur für einen Teil des Wertebereichs korrekt erfolgt.

Auf Grundlage dieses Kontextes lässt sich ableiten, dass zur Beantwortung der Fragestellungen aus Kapitel 6 eine Beurteilung von Zielgrößen dahingehend erfolgen muss, dass sie auf ihr Verhalten bzw. ihre Sensitivität untersucht werden, welches durch eine Variation einzelner Eingangsgrößen erreicht werden kann. Vor diesem Hintergrund lässt sich folgende Methodik formulieren:

1. Wähle eine Frage aus dem Fragenkatalog aus.
2. Identifiziere die beteiligte(n) (Software-)Komponente(n) sowie deren Eingangs- und Ausgangsgrößen.
3. Wähle aus der Menge der Ausgangsgrößen ein Element für die Definition als Zielgröße aus.
4. Wenn die Berechnung der Ausgangsgröße offengelegt ist, fahre mit Schritt 6 fort; sonst
5. Überprüfe die Eingangsgrößen auf ihren Einfluss auf die Ausgangsgröße mit Hilfe von (realen) Experimenten.
6. Wähle die Eingangsgrößen aus, die für die Berechnung der Ausgangsgröße erforderlich sind.
7. Grenze für die Eingangsgrößen den Definitionsbereich ein, den sie innerhalb des gegebenen Systemkontextes annehmen sollen.
8. Lege fest, welcher Wertebereich von der Ausgangsgröße erwartet wird.
9. Definiere im Vorfeld, welcher Werteintervall wie die Fragestellung beantwortet.
10. Wenn dies nicht eindeutig möglich ist, dann entwickle eine eigene Zielgröße als Zusammensetzung von Ausgangsgrößen und weiteren Messgrößen.
11. Wiederhole Schritt 1-8 für alle Fragestellungen bis die Auswahl an Fragen aus dem Katalog abgearbeitet ist.

Die Identifikation der Zielgrößen ist wesentlich davon abhängig, welche Fragestellungen im Fragenkatalog entwickelt wurden. Mit dieser Methodik werden die Antwortmöglichkeiten definiert und somit eine direkte Zuordnung erarbeitet. Dafür wird zunächst eine entsprechende Fragestellung ausgewählt. Dann werden diejenigen (System-)Komponenten ausgewählt, die an dem Systemverhalten beteiligt sind. Dies kann vergleichsweise auf software-architektonisch simpel sein, wenn es sich um eine einzelne Komponente handelt, wie z.B. ein Algorithmus. Wenn hingegen sich die Fragestellung auf mehrere Komponenten erstreckt, kann eine Einzelbetrachtung zunächst für die Komponenten erwogen werden, um anschließend daraus wieder ein Teilsystem zusammenzuführen. Von diesen Komponenten werden dann alle Eingangs- und Ausgangsgrößen zur weiteren Betrachtung ausgewählt.

In einem ersten Anlauf wird ein Element aus der Menge der Ausgangsgrößen selektiert, um sie für die Definition als Zielgröße vorzubereiten. Danach ist entscheidend, ob die Berechnung dieser Ausgangsgröße offengelegt ist, es sich also um eine “White-Box” oder eine “Black-Box”

handelt. Sofern Ersteres zutrifft, kann die Berechnungsvorschrift auf die relevanten Eingangsgrößen Rückschlüsse erlauben. Sollte Letzteres zutreffen, so sind weitere Experimente erforderlich, so dass zumindest die Abhängigkeit von Eingangs- und Ausgangsgrößen geklärt werden kann.

Nach Auswahl der relevanten Eingangsgrößen wird der Definitionsbereich festgelegt, der für die jeweilige Fragestellung voraussichtlich maßgeblich sein wird. Für die Ausgangsgröße legt man den Wertebereich fest, den die Ausgangsgröße annehmen kann, wobei idealerweise die Schnittstelle bekannt ist. Anschließend wird der Wertebereich in Intervalle unterteilt, so dass damit die Antwortmöglichkeiten für die Fragestellung definiert sind.

Sollte bis dahin keine zufriedenstellenden Antwortmöglichkeiten entwickelt werden können, so können weitere Messgrößen erfasst werden und durch die arithmetische oder logische Verknüpfung mit einer Ausgangsgröße als eigenständige Zielgröße zur Lösung der Fragestellung verwendet werden.

Diese Prozedur wird solange wiederholt, bis alle Fragestellungen, die zunächst für die weitere Entwicklung herangezogen werden sollen, mit einer Zielgröße ausgestattet sind. Die Auswahl der zu bearbeitenden Fragestellungen bleibt letztlich dem Entwickler überlassen, da aufgrund von z.B. Projektprioritäten, Ressourcenbeschränkungen oder anderen Einflüssen eine umfassende Bearbeitung nicht zweckmäßig erscheint.

7.2 Konzeptionelle Vorüberlegungen und Technikauswahl

Bis zu diesem Punkt stand bei der Methodik das “Was?” im Vordergrund. Nachdem sich sehr detailliert diese Grundsatzfrage gewidmet wurde, rückt das “Wie?” in den Fokus. Zuvor müssen jedoch noch einige grundlegende Entscheidungen getroffen werden, welche einen Einfluss auf das Simulationskonzept an sich haben werden, ohne dass man sich dadurch bereits auf bestimmte Komponenten oder Technologien festlegt. Bereits in Kapitel 3 wurden verschiedene Entwicklungsansätze aufgezeigt und auf welche Weise sie die Funktionsentwicklung beeinflussen können.

Dazu zählen u.a. folgende Punkte

- Budget
- verfügbarer Zeitrahmen
- Eigenentwicklung vs. Fremdentwicklung

- Art der Leistungserbringung
- Kaufsoftware vs. Software-from-Scratch
- Hybridisierungsgrad (Welcher Anteil wird an Hard- und Software vom originären System verwendet)

Der erste Punkt ist trivial und ein notwendiges Entscheidungskriterium: Die Frage nach dem Budget. Da es in dieser Arbeit mit deutlichem Schwerpunkt um die technologische Gesichtspunkte geht, wird auf diesen Punkt nicht detailliert eingegangen.

Ein weitere wichtige Entscheidung im Vorfeld ist die Vergabe der Entwicklungsleistung und die Art ihrer Erbringung. Das bedeutet konkret, ob die Simulation als Eigenentwicklung innerhalb des eigenen Hauses realisiert wird oder ob ein externer Auftragnehmer dafür beauftragt wird. Dies ist letztlich davon abhängig, wie die Unternehmensstrategie hinsichtlich Software-Engineering-Kompetenz ausgerichtet ist, ob das Know-How bereits für ein solches Softwareprojekt vorhanden ist oder ob die Kompetenz von außen herangezogen werden muss.

Auch die Art der Leistungserbringung kann einen Einfluss auf die Fremdvergabe haben. Denn bisher wurden Aufträge dieser Art standardmäßig über die Erstellung von Lastenheften mit einer detaillierten Beschreibung der Anforderungen und der daran anschließenden Ausschreibung an Fremdfirmen vergeben. Diese antworteten meist mit einem Angebot darauf, welches hin auf technische Konformität überprüft wird, inwiefern alle Anforderungen entsprechend berücksichtigt worden sind. Abschließend erfolgt der Angebotsvergleich der Anbieter für die Umsetzung der Entwicklungsleistungen. Dieses traditionelle Vorgehen, welches prinzipiell einem Wasserfallmodell entspricht, wird meist bei software-fernen Projekten und in großen Unternehmen eingesetzt, da dies die Möglichkeit z. B. der Vergleichbarkeit unter mehreren Anbietern ermöglicht. Aber auch die spätere formelle Abnahme ist aufgrund des höheren Dokumentationsgrades am Ende einfacher möglich.

Bei software-intensiven Projekten kann solch ein Vorgehen durchaus ein Hindernis sein, weil hier im Vorfeld das System bereits am Anfang fertig erdacht sein muss und man eine lange Zeit mit wenig Feedback bis zum nächsten Meilenstein warten muss, bis das theoretisch konzipierte Stück Software dem Praxistest unterzogen werden kann. Eine agile Vorgehensweise mit kurzen Sprints wie z.B. beim SCRUM Ansatz, können hier Abhilfe schaffen, da sie kürzere Iterationen von Lieferungen umfassen und dadurch eine schnelle Validierung über die erreichten Ziele/Anforderungen bietet. Aufgrund von Gesetzesanforderungen hinsichtlich Arbeitnehmerüberlassung kann eine solche Vorgehensweise jedoch die In-House-Entwicklung bevorzugen, da dann die Interdisziplinarität der Entwicklungsteams einfacher umgesetzt und gesteuert werden.

Hintergrund ist, dass klare Kommunikationsrichtlinien bei mehreren beteiligten Unternehmen eingehalten werden müssen wie bspw. mittels zentraler Projektpartner, welche die Aufgaben und Informationen innerhalb des Teams der Fremdfirma weiter delegieren und damit eine direkte Kommunikation zwischen Stakeholder und Entwickler erschwert wird.

Als dritter Punkt muss im Vorfeld die Entscheidung getroffen werden, ob und in welchem Umfang auf Standard-Softwareprodukte zugegriffen oder auf selbstgeschriebene Software gesetzt wird. Hier spielen Fragestellungen über spätere Wartung und Weiterentwicklung genauso eine Rolle wie das Lizenzmodell, mit dem die Software eingekauft oder weiterbetrieben werden darf. Die jeweiligen Entscheidungen haben somit auch wieder direkte Auswirkungen auf den ersten Punkt zur Budgetsituation.

Letzter wichtiger Aspekt ist der Grad der Hybridisierung, also inwiefern auf fahrzeugspezifische Hardware im Rahmen und in welchem Umfang der Simulationsumgebung gesetzt wird. In Kapitel 2 wurden bereits die verschiedenen In-The-Loop-Varianten beschrieben. Welche Variante und zu welchem Grad hängt letztlich mit der Fragestellung zusammen, die durch die Simulation beantwortet werden soll.

7.3 Pre-Processing: Die Testszenario-Erstellung mit MontiCore

Die Testszenario-Erstellung ist ein zentraler Bestandteil im Rahmen der Modellierung. Hier werden Systemelemente innerhalb ihres Systemkontextes angeordnet und über ihre Beziehung untereinander und über ihr Zeitverhalten hinaus beschrieben. Ein Szenario wird dahingehend meist aufgebaut, dass kongruente Situationen auf einen Repräsentanten reduziert werden, der stellvertretend für eine Vielzahl an Parametersätzen als Eingangsgrößen für den Test dient. Diese Parametersätze weisen dann funktional das gleiche oder ein ähnliches Verhalten auf, so dass nicht alle Parametersätze dafür angewendet werden müssen, sondern nur ein Testfall dafür erstellt wird. Dieser klassische Ansatz wird auch als Äquivalenzklassenbildung für Testverfahren bezeichnet.

Wie im späteren Verlauf jedoch noch aufgezeigt wird, ist ein solcher Testansatz im Umfeld von Fahrerassistenzsystemen nicht zielführend, da die unterschiedlichen Parametersätze zu einem anderen Funktionsverhalten führen können.

Testszenarien bilden die Grundlage für alle weiteren Arbeitsschritte im Rahmen der simulativen Qualitätssicherung. Methodisch gesehen ist es daher äußerst sinnvoll, Generierungsmechanismen für ihre Erstellung zu verwenden, da die Effizienzvorteile und insbesondere die Wiederverwendbarkeit sowie die Erweiterungsfähigkeit einer manuellen Erstellung vorzuziehen sind.

Dafür wird das MontiCore-Framework verwendet, welches im Nachfolgenden genauer beschrieben wird.

7.3.1 Das Framework MontiCore

MonitCore ist eine Language Workbench und wurde vom seit 2004 vom Software Engineering Lehrstuhl von Prof. Dr. Bernhard Rumpe an der TU Braunschweig und später an der RWTH Aachen entwickelt [HR17]. Es dient dazu, weitere Werkzeuge im Rahmen der Software-Entwicklung zu erzeugen, daher wird MontiCore auch als Meta-Werkzeug bezeichnet. Dabei werden zwei Ebenen unterschieden: auf der einen Seite werden Werkzeuge direkt generiert, wobei es nicht nur um neu Entworfene handelt, sondern insbesondere um die Wiederverwendung von bereits unabhängig von einander geschaffenen Werkzeugen. Auf der anderen Seite geht es um die Verarbeitung von Modellen, sei es zur Transformation, zur einfachen oder komplexen Analyse der Modelle oder auch zur Simulation jener zugrundeliegenden Laufzeitdaten. Dabei umfasst MontiCore speziell eine Menge an Techniken zur systematischen Wiederverwendung von Sprachkomponenten, welche sowohl Standard als auch domänenspezifische Sprachen umfassen, und erlaubt ihre Entwicklung, Erweiterung und Vererbung. Für eine detaillierte Einführung in MontiCore sei auf das Standardwerk von Rumpe et Hölldobler verwiesen. [HR17]

In Kapitel 2 wurde bereits auf die Vorteile der generativen Software-Entwicklung eingegangen. Die Nutzung von MontiCore und des generativen Ansatzes zur Testszenario Erstellung ergibt sich aus den jeweiligen Effizienzvorteilen in Form der Wiederverwendung und Wartbarkeit des zugrundeliegenden Modells. Insbesondere in dem hier vorliegenden Fall wird die Modellierung der Szenarien tool-unabhängiger gestaltet, wie später noch aufgezeigt wird.

7.3.2 Entwurf einer Szenario-DSL

Der Entwurf einer Scenario-DSL hat zum Ziel, eine werkzeug-unabhängige, domänenspezifische Sprache (DSL) zu definieren und dabei folgende Aspekte zu berücksichtigen:

1. Die Scenario-DSL muss Szenarien in der Automotive-Segment und insbesondere zu Fahrsituationen zwischen Fahrzeugen beschreiben können. Davon losgelöst sind mögliche Optimierungsverfahren wie bei Variationen des Simulationsraumes z.B. bei Parametervariationen von Trajektorien.
2. die Scenario-DSL soll für die Wartung und Pflege Abstraktionen ermöglichen, damit bspw. werkzeugunabhängig die Szenarien generiert werden können.

3. Darüber hinaus soll die Szenario-DSL Konzepte für eine verbesserte Variabilität bieten, um neue Szenarien mit zusätzlichen Systemelementen oder auch ergänzenden Verfahren wie der Trajektorienvariation zu verwenden. [KHP+15]

Reverse-Engineering als Methodik für eine Scenario-DSL

Wenn auf eine Spezial-Software zur Simulation des Systemkontextes zurückgegriffen wird, wird meist das Szenario in einem bestimmten Format abgespeichert, wodurch das Szenario beschrieben ist. Dies kann ein offenes Format wie beispielsweise ein XML-Schema sein; es können aber auch geschlossene Formate, die nur innerhalb der Software bekannt sind, verwendet werden. Die Herausstellung der Vorteile eines offenen oder geschlossenen Formates ist hier nicht Gegenstand einer Erörterung. Vielmehr ist Grundvoraussetzung für die Nutzung einer DSL eine offene Beschreibung von Szenarien, um geeignete Code-Generatoren bauen zu können, welche die Dateien im entsprechenden Format erzeugen.

Innerhalb der Softwarelösungen zur Simulation des Systemkontextes werden häufig graphische Editoren bereitgestellt, mit welchen dann die Testszenarienerstellung erfolgt. Eine rein textbasierte Variante zur Erstellung ist dabei in der Regel nicht möglich, zumindest bietet der Softwarelieferant keine hinreichende Unterstützung an.

Um nicht nur diese Lücke zu schließen, sondern auch ein softwareunabhängiges Beschreibungsformat von Testszenarien zu erhalten, wird im folgenden beschrieben, wie man aus einer Szenarienbeschreibung mittels DSL in MontiCore eine Szenariengenerierung etablieren kann, die eine reine textbasierte Bearbeitung vorsieht. Zu den Vorteilen einer solchen Entwurfsform sei auf [HR17] verwiesen.

Als Ausgangsbasis dient für diese Arbeit die Software Virtual Test Drive von der Fa. Vires, welche eine XML-basierte Erstellung von Szenarien vorsieht. Dazu existiert innerhalb des Software-Pakets ein entsprechender Editor, der anhand von graphischen Elementen eine Anordnung der Simulationsobjekte erlaubt. Aus dieser Beschreibung heraus wird eine Scenario-DSL entwickelt, welche nicht nur den Detaillierungsgrad der VTD Szenarien umfassen wird, sondern darauf aufbauend weitere Abstraktionen wie z.B. als Spracherweiterung eingebaut werden können.

Die XML Schema Definition (XSD) der VTD Testszenarien bildet die Basis für die Grammatik, der hier entwickelten Scenario-DSL. Alle im XSD abgebildeten Eingabeinstanzen sind wichtige Informationen für die durchgeführte Simulation in VTD, so dass die Scenario-DSL zumindest die Menge an Instanzen umfassen muss. Darum wird das gesamte XSD in eine Grammatik überführt, so dass ein Testentwickler hinterher alle Funktionen der VTD Simulationsumgebung

nutzen kann. Die Grammatik ist das Grundgerüst für alle später entwickelten Anpassungen. [KHP+15]

Um die zweite Anforderung entsprechend umzusetzen wurde, eine Reihe von Szenarien aus der Fallstudie zu den Consumer Tests analysiert. Als Ergebnis konnte herausgearbeitet werden, dass sich die Szenarien nur in einigen Positionen unterscheiden. Die nachfolgende Gegenüberstellung verdeutlicht exemplarisch diese Unterscheidung.

Abbildung 7.1: Vergleich von Szenarien hinsichtlich gleicher Codeanteile [KHP+15]

In dieser Abbildung 7.1 können die Unterschiede zweier VTD Szenarien betrachtet werden, die aus einem Projekt zum vorausgehenden Fußgängerschutz entlehnt worden sind. Da durch den hohen Anteil an gleichem Code recht viel Information redundant bereitgestellt wird, wurde folgendes Konzept zur effizienten Repräsentation herangezogen: Weil sich die jeweiligen Testszenarien in einigen Parametern nur inhaltlich unterscheiden, können die gleichen Codeanteile in einer Basisgruppe zusammengefasst werden, während die konkretisierenden Informationen in einer eigenen Einheit festgehalten sind. Abbildung 7.2 verdeutlicht schematisch diesen Zusammenhang. [KHP+15]

Das Szenario vFGS-basis bildet die abstrakte Einheit ab, in welcher der überwiegende Teil mit dem gleichen Code vorhanden ist. Die Konkretisierungen der Szenarien wird dann über die Unterszenarien vFGS-CP1-10kmh usw. realisiert. [KHP+15]

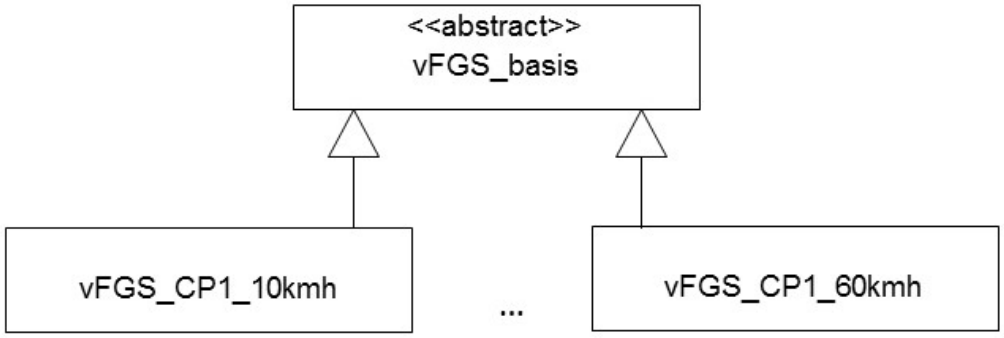


Abbildung 7.2: Schema für das Konzept der Abstraktion von Szenarien

7.3.3 Das Meta-Modell der Scenario-DSL

Die Abbildungen 7.3, 7.4, 7.5 und 7.6 zeigen das Metamodell zur textuellen Repräsentation der VTD Szenarien, wodurch es möglich ist, neue Instanzen zu erzeugen und automatisch zu verarbeiten. Dabei ist das Meta-Modell ähnlich im Aufbau wie die VTD Szenario Definition. Die Klasse ASTScenarioDSL bildet den Grundbaustein des Meta-Modells, da sie die gesamte ScenarioDSL Instanz darstellt und alle weiteren Klassen miteinander verknüpft. Die weiteren Klassen ASTCharakterListElement, ASTVehicleElement, ASTDriverListElement ASTObject-Element und ASTLayoutElement beschreiben die Informationen zur Konfiguration des Umfeldes. Hingegen werden Informationen für die VTD-Szenario Elemente PulkDef, MovingObjects-Control und TrafficControl Elemente durch die Klassen ASTPulkTraffic, ASTPulkDef, AST-MovingObjectsControl and ASTTrafficControl repräsentiert. Es werden auch noch zusätzliche Klassen verwendet, die z.B. die Koordinatenrepräsentation innerhalb der VTD Szenarien übernehmen, wobei die Klasse ASTCoordinate diese Funktion übernimmt.

Die meisten der VTD-Szenario Elemente sind optional, so dass sich die Assoziationen zwischen den Klassen meist der Form “*” oder “1..*” als Kardinalitäten zuordnen lassen. Dies ist das Resultat des Faktorisierungskonzeptes mit Abstraktion und Konkretisierung der Szenarien, wodurch es möglich ist, mehrere Scenario-DSL Instanzen zu einem VTD-Szenario zusammenzufassen.

Die Faktorisierung durch Basis- und Konkretisierungsszenario ist in den folgenden Quelltexten in Form der konkreten Scenario-DSL-Syntax dargestellt. Auch hier dient wieder das Beispiel des einfachen Szenario aus dem vorausschauenden Fußgängerschutz (vFGS) als Ausgangsbasis.

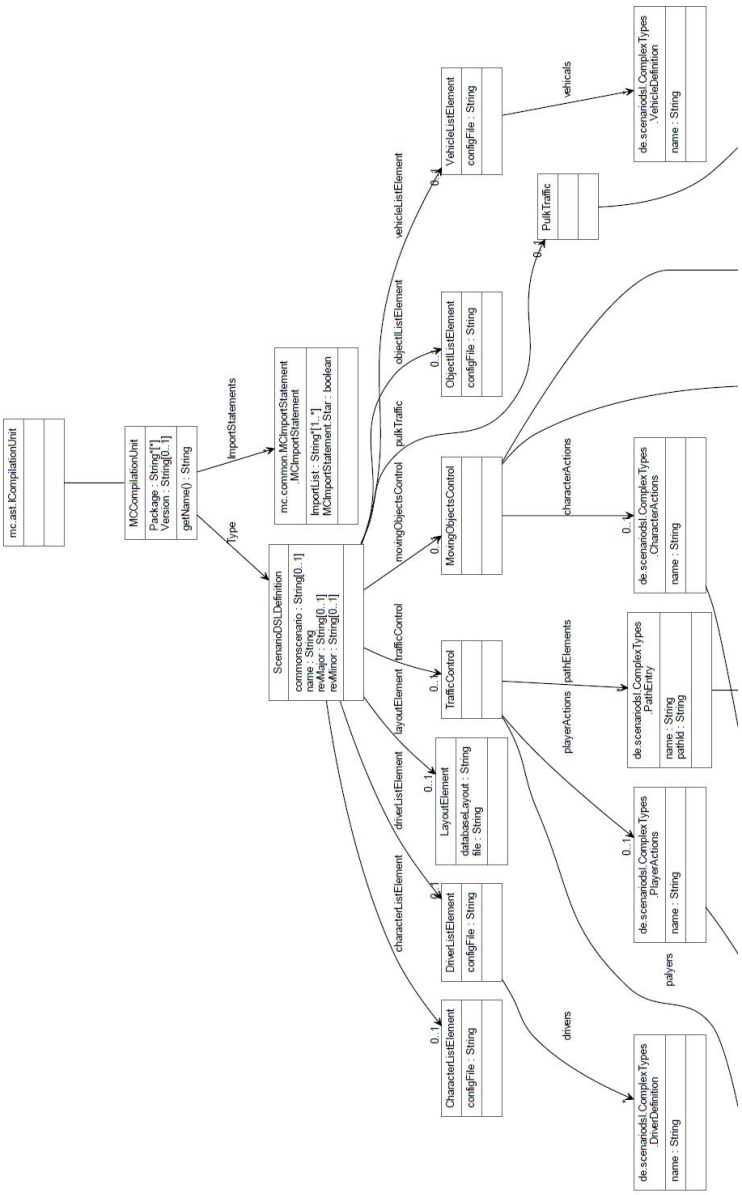


Abbildung 7.3: Das Meta-Modell der ScenarioDSL - oberer Teil

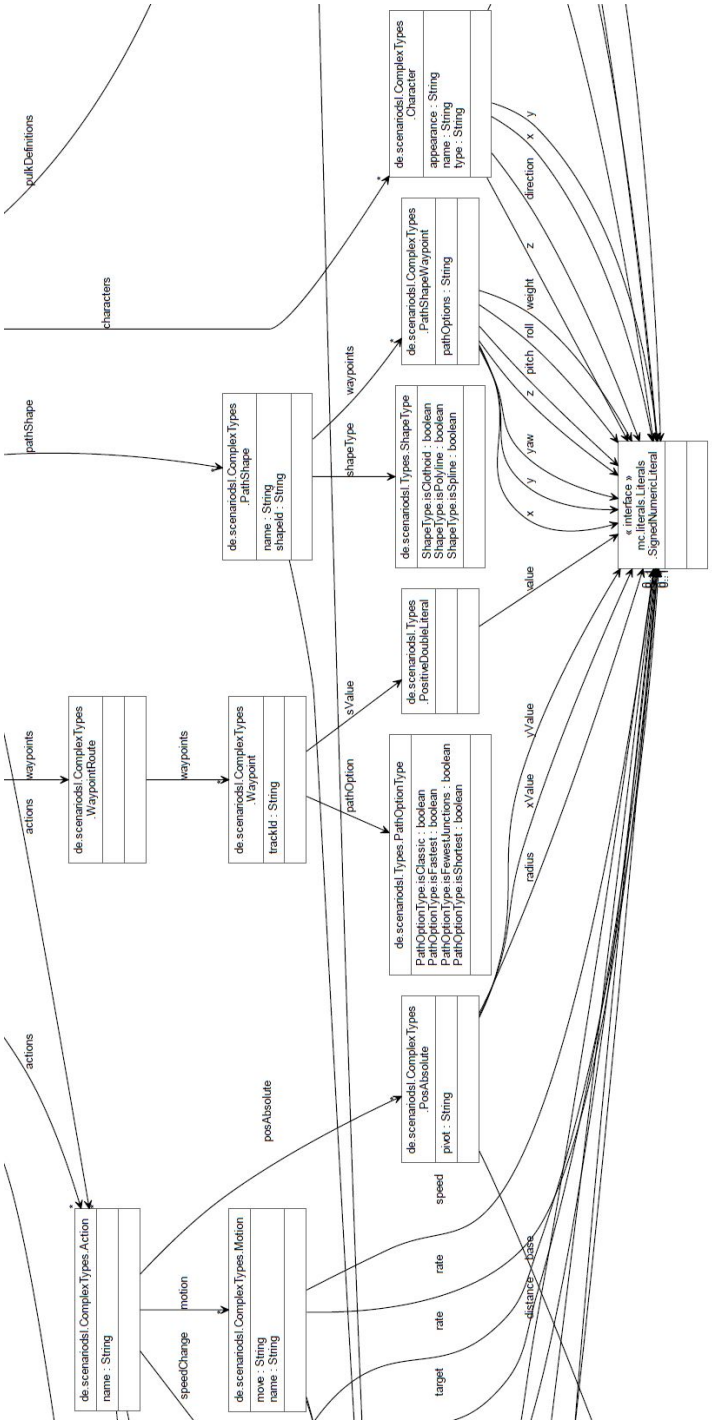


Abbildung 7.4: Das Meta-Modell der ScenarioDSL - mittlerer Teil

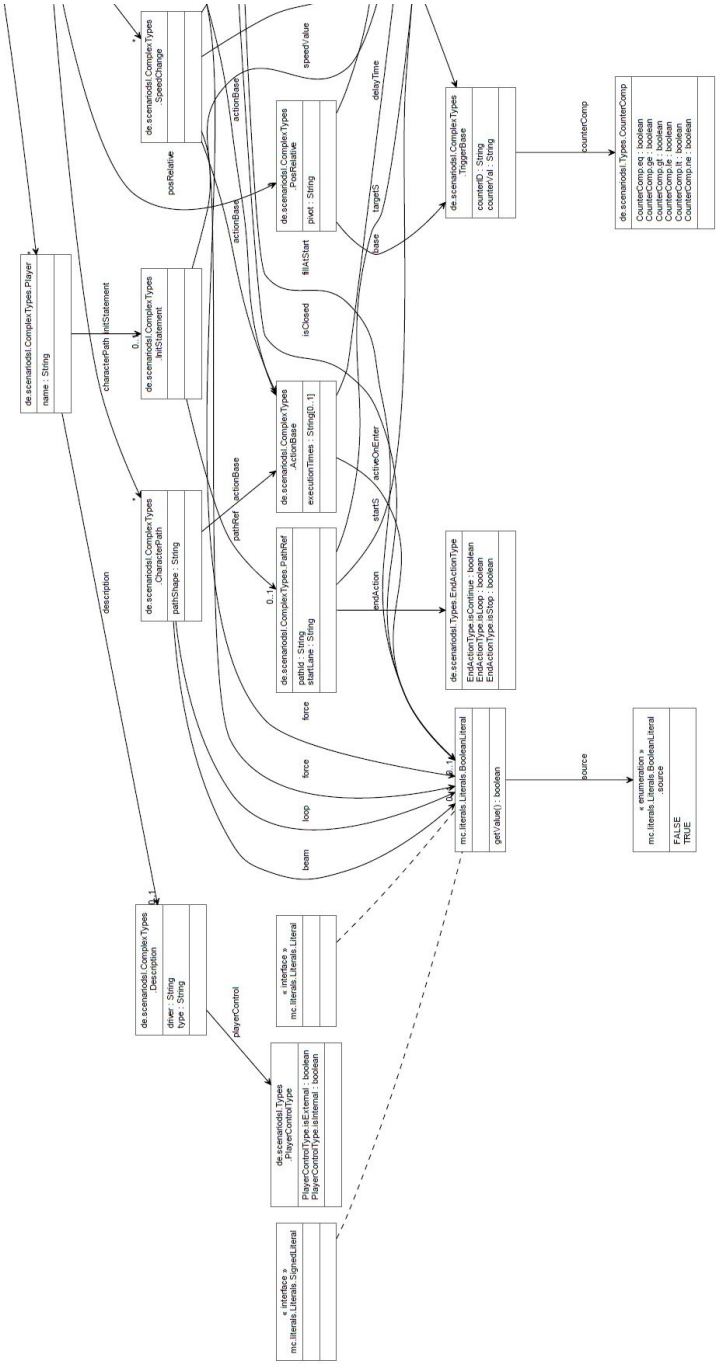


Abbildung 7.5: Das Meta-Modell der ScenarioDSL - linker Teil

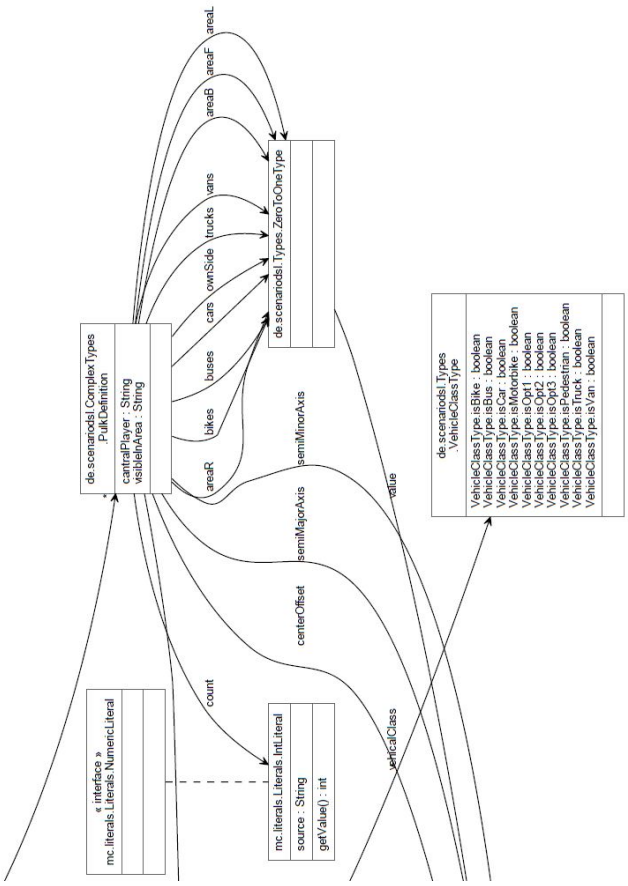


Abbildung 7.6: Das Meta-Modell der ScenarioDSL - rechter Teil

Die folgenden Zeilen zeigen den Quellcode mit den allgemeinen Elementen des Szenarios, wie z.B. die Pfade, die Konfiguration und andere Elemente. [KHP+15]

```
package acceptanceTest;

scenario BaseScenarioVFGS {
    Layout { database="Projects/Current/Ive/Testumgebung_v02.opt.ive" }
    VehicleList { configFile="Distros/Current/Config/Players/vehicleCfg.xml" }
    DriverList { configFile="Distros/Current/Config/Players/driverCfg.xml"}
    CharacterList { configFile="Distros/Current/Config/Players/characterCfg.xml" }
    ObjectList { configFile="Distros/Current/Config/Players/objectCfg.xml" }
    TrafficElements {}
    PulkTraffic {
        PulkDef { VisibleInArea=-1 CenterOffset=1.0000000000000000e+02 }
    }
    TrafficControl {
        Path Path01 PathId=1 {
            Waypoints {
                PathOption=shortest s=6.4304809570312509e+00 TrackId=1;
                PathOption=shortest s=1.9941957397460938e+03 TrackId=1;
            }
        }
        Player External {
            Description {
              Driver=DefaultDriver
              Control=external
              Type=VW_PassatVariant_2011_night}
            Init {
                Speed = 2.7777777777777777e+00
            }
        }
    }
```

```
    MovingObjectsControl {
        PathShape FG_PathShape01 {
            ShapeId=1
            ShapeType=polyline
            Closed=false
            Waypoints {
            X=0.0000000000000000e+00
            }
        }

        Character FG_01 {
            CharacterType=mideast_mped_07
            Class=pedestrian
            Appearance=sk_arab_male_7
            StartPosAbs = ( X=2.4106994628994016e-03)
        }

        CharacterActions FG_01 {
            Action moveAndSet {
                PosAbsolute {CounterComp=COMP_EQ Pivot=External}
                Motion { Move=walk Rate=3.0000000000000000e+00}
                CharacterPath {Loop=false}
            }
        }
    }
}
```

Im nachfolgenden Quelltext werden dann die ergänzenden Elemente und Konfigurationsangaben codetechnisch beschrieben, wie z.B. die konkreten Positionen von Fußgänger und Fahrzeug.

```
package acceptanceTest;

scenario CP1_20_VFGS RevMinor=2 RevMajor=1 extends BaseScenarioVFGS ↩
 {
    TrafficControl {
        Player External {
            Init {
                PathRef { StartS=9.4000000000000000e+02  }
            }
```

```
        }
    }

    MovingObjectsControl {
        CharacterActions FG_01 {

            Action Die {
                PosAbsolute {CounterComp=COMP_EQ Pivot=External}
                Motion { Move=walk Rate=3.0000000000000000e+00}
                CharacterPath {Loop=false}
            }
        }
    }
}
```

7.3.4 Technische Komponenten der Scenario-DSL

Damit die Scenario-DSL verarbeitet werden kann, müssen mehrere Komponenten mit Hilfe des DSL-Framework MontiCore implementiert werden. Dazu zählen im Frontend

- Parser
- Symboltabelle
- Kontextbedingungen

und im Backend der Generator. Diese wurden nach den gängigen Standards des Software-Engineering, wie in [HR17] beschrieben, entwickelt. Weiterhin wurden einige Maßnahmen getroffen, welche die Qualität der Implementierung sichern sollen:

- Dokumentation
- Beispiele als Proof of Concept
- Test

Innerhalb des Quellcodes wurden kontinuierlich JavaDoc Kommentare eingepflegt, welche die einzelnen Code Fragmente beschreiben. Weitere VTD Szenarien wurden prototypisch implementiert, um ein Proof of Concept bereitzustellen. Unit Tests übernehmen die Aufgabe der Funktionssicherung für die Abläufe und Funktionen. Gleichzeitig dokumentieren die Testfälle als Dokumentation des Funktionsumfanges des implementierten Werkzeugs.

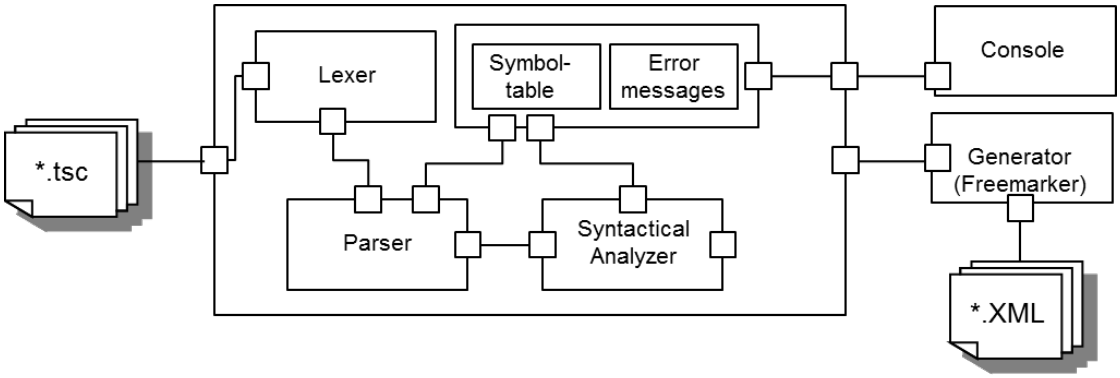


Abbildung 7.7: Die Architektur des Generators (vgl. [KHP+15])

In Abbildung 7.7 ist die Architektur des Generators dargestellt, welche als Eingang die Scenario-DSL-Instanzen vorsieht. Die Komponenten Parser und Lexer verarbeiten die Instanzen dann zum Abstract Syntax Tree (AST), der dann mit weiteren semantischen Informationen versehen wird. Die Überprüfung auf semantische Konsistenz wird durch einen weiteren Ablauf gewährleistet, wobei die auftretenden Fehler, Diagnosebeschreibungen und semantischen Anomalien über die Konsole ausgegeben werden. Der AST wird dann an den Generator übergeben, der auf Freemarker mit entsprechenden Templates und einer zugehörigen Java-Infrastruktur basiert, so dass am Ende XML-Dateien entstehen, die mit Hilfe der Simulationsumgebung VTD abgespielt werden können. [KHP+15]

Die dritte Anforderung zur Variabilität wird im Folgenden vorgestellt. Dabei handelt es sich um die Variationskomponenten, welche die VTD Szenarien dann mit weiteren relevanten Informationen für ihren jeweiligen Einsatzzweck konfigurieren.

7.4 Pre-Processing: Modellierung von Parametervariationen zur Trajektorien-Generierung

Die Simulation von Aktiven Sicherheitssystemen und deren Signalverläufe sind in Abhängigkeit vom gewählten Szenario in der Regel zunächst idealtypische Verläufe. Das heißt, dass meist der Fahrweg eines Fahrzeugs definiert wird, auf welchem dieses dann mit einer bestimmten Geschwindigkeit fährt und an einer bestimmten Stelle seinen Weg mit anderen Objekten kreuzt. Darauf reagiert das zu untersuchende System und regelt entsprechend die fahrdynamischen Größen des Fahrzeugs. Auch ein mehrmaliges Ausführen hintereinander soll immer zum gleichen Ergebnis führen: Das Fahrzeug fährt dem vorbestimmten Weg entlang und trifft auf andere Ver-

kehrsteilnehmer und die Funktion regelt auf diese Objekte, wobei der analytisch berechenbare Verlauf als “idealtypisch” zu bezeichnen ist. Eine tolerierbare Abweichung von diesem idealen Verlauf wird in diesem Zusammenhang als eine Signal- bzw. Parametervariation bezeichnet. Über alle möglichen Variationen lässt sich dann ein Zustandsraum definieren, der in Abhängigkeit der variierten Parameter und unveränderten Größen beschreibbar ist. Das Ziel ist es, sowohl diesen Zustandsraum möglichst vollständig zu modellieren und als das exponentielle Wachstum der daraus resultierenden Signalverläufe nach Möglichkeit gering zu halten, indem weiterführende Zusammenhänge erkannt werden. [KHP+15]

7.4.1 Modellierung zur Variation von Parametern zur Strukturierung von Testfällen

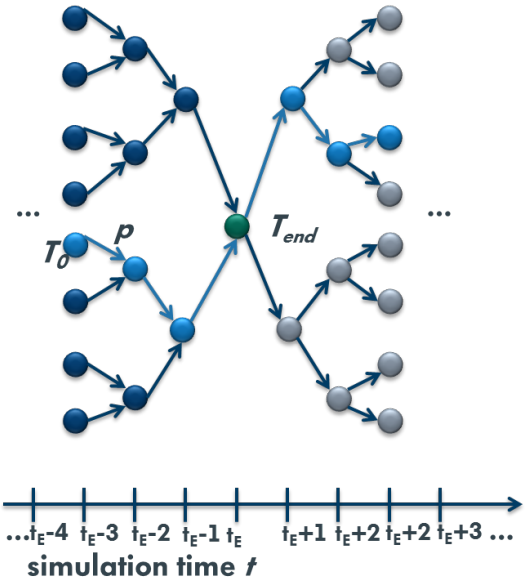


Abbildung 7.8: Graphische Darstellung zum Modell. [KHP+15]

In 7.8 wird das grundlegende Prinzip zur Modellierung von Parametervariationen veranschaulicht. Über die X-Achse verläuft die Simulationszeit, die Y-Achse repräsentiert z. B. einen Parameter. Der grün eingefärbte Punkt entspricht einem Stimulationsvektor für das Modell der Fahrzeugfunktion. Es gilt die Annahme, dass sich unter Berücksichtigung deterministischen Verhaltens sowohl des Funktionsmodells als auch des Systemkontextes die Stimulationsvektoren für die nachfolgenden Zeitschritte auf dieselbe Art und Weise weiter fortschreiben.

Eine Untersuchung eines Aktiven Sicherheitssystems zur Sensitivitätsanalyse oder für Robust-

heitstests wird durch die strukturierte Parametervariation mittels systematischer Veränderung einer Vektorkomponente entweder in den Zeitpunkten vor oder/und nach dem Punkt t_E erzielt und währenddessen die Auswirkungen der Veränderung auf das antizipierte Verhalten des SUT analysiert werden. Die Abbildung 7.8 zeigt zudem einen gerichteten, azyklischen Graphen, der alle mögliche Variationen des Stimulationsvektors als Baum beschreibt, wobei die möglichen Veränderungen vor t_E in dunkelblau und nach t_E in grau visualisiert sind. Der hellblaue Pfad beschreibt einen potentiellen Pfad durch den zweiseitigen Baum. [KHP+15]

Im Folgenden sei ein Beispiel zur besseren Veranschaulichung beschrieben. Dabei wird als variierender Parameter der Lenkeinschlag des Fahrzeugs angenommen, wobei es zwei Ausprägungen geben kann: Einen linksseitigen Lenkeinschlag L_l und einen rechtsseitigen Lenkeinschlag L_r. Daraus resultiert, dass entweder nach dem Punkt t_E in den linken Teilgraphen oder in den rechten Teilgraphen verzweigt werden, also entweder einmal nach links oder einmal nach rechts gelenkt wird. Das wiederholt sich zu den nachfolgenden Zeitpunkten t_{E+2}, t_{E+3}, ..., t_{E+m} rekursiv, so dass die Variationsmöglichkeiten sich um die Anzahl der Ausprägungen je Zeitpunkt potenzieren. Daraus resultiert, dass die rechts- und linksseitigen Teilgraphen die Variationen vor bzw. nach jenem potentiell maßgeblichen Zeitpunkt t_E repräsentieren.

Der hellblau markierte Pfad entspricht einer Abfolge von Variationen und unter der Annahme, dass keine weiteren Abweichungen von Parametern im Systemkontext und in der Fahrzeugfunktion erfolgen, ist diese Abfolge an Lenkeinschlägen nun systematisch und reproduzierbar:

- t_{E-3}: Links
- t_{E-2}: Links
- t_{E-1}: Rechts
- t_E: keiner
- t_{E+1}: Links
- t_{E+2}: Rechts
- t_{E+3}: Links. [KHP+15]

7.4.2 Anwendungsmöglichkeiten von modellbasierten Pfaden zur strukturierten Parametervariation

Für das Modell der strukturierten Parametervariation mit seiner formalisierten und systematisierten Darstellung gibt es nun eine Reihe von Nutzungsformen, die im folgenden kurz erläutert

seien:

- Äqivalenzklassenuntersuchung
- Zuverlässigkeitsuntersuchung
- Sensitivitätsuntersuchung
- Gesamtbewertung von Systemfunktionen (z.B. Endverbrauchertests)

Die Äquivalenzklassenuntersuchung kann beispielsweise durch eine Zusammenfassung von Pfaden zu Klassen durchgeführt werden. Mittels stochastischer Auswahl kann dann eine Untersuchung des Verhaltens der Fahrzeugfunktion im Aktiven Sicherheitssystem vorgenommen werden, wobei hier speziell die Ränder der Äquivalenzklassen mit geeigneten Repräsentanten von gesteigertem Interesse sind und bei der Absicherung der Robustheit helfen.

Während der Funktionsentwicklung kann nicht ausschließlich mit standardisierten Simulationsmodellen gearbeitet werden, da gerade nicht-funktionale Anforderungen an eine Funktion wie z.B. die Zuverlässigkeit eines vorgelagerten Sensors bei einem Aktiven Sicherheitssystem ebenso eine gewichtige Rolle spielen. Dies wäre exemplarisch die zuverlässige Bereitstellung von Messwerten: Durch die systematische Parametervariation **vor** einem maßgeblichen Zeitpunkt kann untersucht werden, wie sich eine nachgelagerte Systemkomponente **nach** diesem arbeiten würde. Erkenntnisse aus dieser Betrachtung können dann wiederum als Anforderungen an Zulieferer gestellt und gleichzeitig als Tests zur späteren Validierung im Rahmen der Abnahme fungieren. Es sei an dieser Stelle darauf hingewiesen, dass die hier gemachten Aussagen nicht im Widerspruch zu bereits zuvor gemachten Aussagen stehen.

Endverbrauchertests wie z.B. New Car Assessment Programs (NCAP) werden von unabhängigen Institutionen durchgeführt und prüfen Fahrzeuge anhand eines jeweiligen Protokolls auf verbraucherrelevante Aspekte wie beispielsweise die Fahrzeugsicherheit. Wie bereits in Kapitel 3 dargestellt, dürfen die Fahrzeuge und anderen Objekte in gewissen Toleranzen von der Ideallinie im Testszenario abweichen. Das hier eingesetzte Verfahren zur Parametervariation kann dazu herangezogen werden, um das gesamte Spektrum an Variationen durchzuspielen und die jeweiligen Konsequenzen für eine spätere Fahrzeugbewertung aufzuzeigen. Auch die Untersuchung von Gesamtsystemen mit seinen Komponenten kann in diesem Zusammenhang erfolgen und stellt damit eine Erweiterung des hier genannten Anwendungsfalls dar. [KHP+15]

7.4.3 Beherrschung der Komplexität des Modells

Eine entscheidende Herausforderung bildet der schnell anwachsende Lösungsraum für mögliche Pfade innerhalb des Modells. So sind bereits bei den in Abbildung 7.8 aufgeführten Verzweigungen und 10 Zeitschritten die Menge an Pfaden auf 1024 angewachsen. Dieses Wachstum muss durch geeignete Verfahren beherrschbar gestaltet werden, in dem die zu untersuchenden Pfade begrenzt werden. In der Regel sind hier Experten aus den jeweiligen Fachbereichen der Entwicklung heranzuziehen. Folgende Möglichkeiten bestehen, um die Komplexität des Graphen zu reduzieren:

- Beschränkung des Lösungsraumes
- Beschränkung der Parameterdimensionen
- Entkopplung von Simulations- und Variationszeitraum
- Plausibilisierung von Parametervariationen
- Abhängigkeit von Parametervariationen
- Wahrscheinlichkeitsannotationen
- Kenntnisse über die Systemkomposition

Die Beschränkung des Lösungsraumes kann allein durch die Begrenzung des betrachteten Zeitraumes erreicht werden, da sich dieser mit jedem Zeitschritt mindestens verdoppelt. Aber auch eine Beschränkung der Parameterdimensionen kann die Anzahl der Pfade beeinflussen. Je nach Ausprägung des variierenden Parameters vervielfacht sich pro Zeitschritt die Anzahl der Pfade des Graphen.

Bisher wurde auch davon ausgegangen, dass ein Zeitschritt innerhalb der Simulation auch einem Zeitschritt für eine Variation eines Parameters entspricht. Eine Variationsfolge kann aber aus physikalischen Gründen nicht möglich sein. Durch eine Entkopplung dieser Zeiten kann eine gewisse Menge an Pfaden bereits aus der Betrachtung herausgenommen und damit die Komplexität reduziert werden.

Eine Alternative, die Anzahl aller Pfade zu reduzieren, besteht darin, semantisch nicht sinnvolle Pfade ebenfalls herauszunehmen. Diese Plausibilisierung von aufeinander folgenden Variationen grenzt sich vom vorherigen Aspekt zur Entkopplung der Zeiten dahingehend ab, dass es auch im Falle einer gröberen Auflösung semantisch weniger sinnvolle Pfadfolgen geben kann. Gleichwohl besteht zwischen diesen beiden Aspekten eine Verwandtschaft im Ergebnis.

Eine weitere verwandte Maßnahme ist die Abhängigkeit von gleichen Parametervariationen festzulegen, wie häufig Pfadfolgen variiert werden (dürfen); dabei ändert sich die zeitliche Auflösung des Variationsraumes zunächst nicht. Es wird vielmehr festgelegt, wie oft die gleiche Ausprägung aufeinander folgen darf. Dieses Verhalten lässt sich auch in Form einer Änderungsfrequenz beschreiben.

Durch die Nutzung von Wahrscheinlichkeitsannotationen an den jeweiligen Kanten kann zu einer Bevorzugung gewisser Pfade führen, so dass sich wiederum die Gesamtanzahl der Pfade reduzieren lässt, wie beispielsweise durch die kumulierten Wahrscheinlichkeiten über alle Kanten. Es ist hierbei jedoch zu berücksichtigen, dass durch Wahrscheinlichkeiten eine Zufallskomponente Einzug erhält. Wie dies im Hinblick auf eine systematische Untersuchung dienlich ist, muss im Einzelfall entschieden werden.

Als letzte Möglichkeit ist hier die Berücksichtigung der einzelnen Systemkomponenten und ihrer funktionalen Eigenschaften zu nennen. So kann zum Beispiel ein enthaltener Sensor, der jeweils nur eine gewisse zeitliche messtechnische Rückschau erlaubt, um für gegenwärtige Werte und Werteprognosen wie bei Kalman-Filtern herangezogen zu werden. Dadurch kann die Untersuchung auf den Zeitpunkt in unmittelbarer Umgebung um den Punkt t_E beschränkt werden. [KHP+15]

7.5 Modellierung der Sicherheitsfunktion

Dieser Abschnitt befasst sich zunächst aus einer allgemeinen Sicht heraus mit der Modellierung der Sicherheitsfunktion, später jedoch auf die eigentliche Fallstudie beziehend. Dies ist darin begründet, dass die Individualität der Rahmenbedingungen zur Entwicklung passender Modelle und der Simulationsfokus zu weitreichend ist. Wie bereits in Kapitel 6 deutlich wurde, hängt das Modell vom eigentlichen Ziel ab: Was will mit der Simulation erreicht werden? Diese Frage lenkt mittelbar das Augenmerk auf die Festlegung des Detailgrades, welcher vom Verhalten der Modelle gefordert wird. Meist wird ein realitätsnaher Top-to-Bottom Ansatz gewählt, der eine Gesamtsimulation eines Systems mit der Modellierung der jeweiligen Einzelheiten und Effekten umfasst.

In diesem Abschnitt soll jedoch ein entgegengesetzter Ansatz herausgearbeitet werden, der sich zunächst der Frage widmet, welche Einzelkomponente den größten Beitrag zur Bestimmung der Zielgröße leistet und welche weiteren Komponenten ihren Input dafür liefern. Damit soll erreicht werden, dass man sich auf das Wesentliche konzentriert. Andernfalls kann die Realisierung der Abstraktion aus einer Gesamtsicht heraus in dem bereits erwähnten “Verzetteln” von Detailfra-

gen hinsichtlich Modellierung enden, so dass die eigentliche Qualitätsverbesserung nicht mehr im Vordergrund steht.

Eine pauschale Antwort auf die Frage, “Welcher Detaillierungsgrad für eine hinreichende Simulation benötigt wird?” kann nicht verallgemeinernd gegeben werden, da diese Frage allein von der Ausarbeitung der Zielgröße, dem zu untersuchenden System und vom Systemkontext abhängig sind. Es besteht jedoch die Chance, mit der richtigen Fokussierung und mit einer geeigneten Untersuchungsmethode, den Entwicklungsaufwand für die Modelle bzw. die Integration von bereits entwickelten Modellen zu reduzieren.

7.5.1 Komplexität von Modellen

In diesem Zusammenhang wird auf die Frage eingegangen, in welcher Komplexität die Modelle erstellt worden sein müssen, damit eine Aussagefähigkeit hinsichtlich der im Analyseteil entwickelten Fragestellungen erzielt werden kann. Speziell sollen dabei ideale Sensor- und Aktor-Modelle im Fokus stehen und ihre Verwendung hinsichtlich Vor- und Nachteile diskutiert werden.

Ausgangspunkt zur Modellbildung ist das Ziel zur Untersuchung der Transformation von Eingangs- zu Ausgangsgrößen. Das heißt, dass es ein Modell für die Erstellung bzw. die Generierung sinnvoller Daten als Eingangsgrößen geben muss. Diese werden beim Fahrzeug durch die umfelderfassenden Sensoriken und in Teilen auch durch die fahrdynamischen Erfassungssysteme erzeugt. Diese Zusammenstellung an Eingangsdaten wird dann anhand der implementierten Funktion in Entscheidungen für die Aktorik transformiert. Aufgrund dieser kontrollierten Transformation durch Vorgabe der Daten kann das eigentliche Subject Under Test (SUT) einer Qualitätskontrolle unterzogen werden.

Dabei ist grundsätzlich die Möglichkeit in Betracht zu ziehen, simplifizierte Modelle zu wählen. Hintergrund dieser Überlegung ist, dass ohnehin eine Vielzahl an Testfällen mit vereinzelten und kontrollierten Parametervariationen durchgeführt werden müssen, um mögliche verdeckte Fehlerwirkungen aufzuzeigen und entsprechend korrigieren zu können. Damit kann der Implementierungs- bzw. Kostenaufwand deutlich reduziert werden, um mehr der Fokus auf die Entwicklung bzw. den Test des SUT zu legen. Eine erste Herangehensweise die Modelle vergleichsweise simpel zu implementieren kann darin bestehen, dass diese zunächst auf den grundlegenden physikalischen Gesetzmäßigkeiten beruhen und die Abbildung von messtechnischen Fehlern oder funktionalen Unzulänglichkeiten nicht berücksichtigt werden, die jeweils in der Realität auftreten würden (z.B. “Geisterobjekte” als Konsequenz aus der fehlerhaften Objekter-

kennung im Post-Processing bei Radarsensoren). Dadurch ist es möglich jeweilige Seiteneffekte zu vermeiden, da die Auswirkungen einzelner Veränderungen an Parametern genauer untersucht und eine zufallsbehaftete Beeinflussung dieser ausgeschlossen werden kann. Gleichzeitig würde bei der Untersuchung des SUT das Ursachenspektrum eine Erweiterung auf die jeweiligen Modelle erfahren, welches es gilt auszuschließen. Daher ist auch eine separate Betrachtung der einzelnen Variablen notwendig, wodurch sich jedoch eine erhebliche Aufwandssteigerung aufgrund der erhöhten Modellkomplexität ergibt. Außerdem wird der Aufwand nicht mehr nur auf die Funktion selbst gerichtet, sondern auch in die Daten liefernden Modelle investiert. Somit erlaubt die Konzentration auf die kontrollierte und vereinzelte Parametervariation im Rahmen einer Sensitivitätsanalyse eine Betrachtung um einen möglichen Hotspot bei einer Fehlerwirkung.

7.5.2 Kontinuität der Eingangs- und Ausgangsdaten

Wichtig ist weiterhin, ob das SUT innerhalb einer Open-Loop oder Closed-Loop Simulation realisiert wird. Bei der Open-Loop Variante würden die Eingangsdaten kontinuierlich an das SUT gesendet, in die Ausgangsdaten transformiert und keine Rückwirkung auf die Generierung der Eingangsdaten haben, so dass eine reine “Verarbeitung” der Eingangsgrößen hin zu den Ausgangsgrößen im Vordergrund steht. Bei einer Closed-Loop Simulation betrifft das Berechnungsergebnis des SUTs ganz unmittelbar auch die Eingangsdaten aus den Sensormodellen und hat somit eine Rückkopplung auf das SUT an sich. Hierzu beeinflussen die Ausgangsgrößen als neue Eingangsdaten die Aktorik, die wiederum mit dem Systemumfeld bzw. -kontext interagiert. Das hat Auswirkungen auf die Wahrnehmung der Sensorik, die daraus neue Eingangsdaten für das SUT erzeugt.

7.5.3 Echtzeit und Skalierbarkeit

Weitere zu betrachtende Aspekte zielen auf die mögliche Forderung nach Echtzeitverhalten innerhalb der Simulation ab. Echtzeit ist meist dann unabdingbar, wenn im Rahmen einer Hardware-in-the-Loop Simulation echte Steuergeräte und die jeweilige Software zum Einsatz kommen. Das schränkt jedoch eine mögliche mehrfache Instanziierung der Simulationsumgebung ein. Eine Software-in-the-Loop Simulation hingegen kann die eigentliche Berechnung der Ergebnisse in Abhängigkeit der Leistungsfähigkeit der Hardware durchführen und dadurch die Simulationsgeschwindigkeit deutlich erhöhen. Auch die Parallelisierung der Berechnungen durch mehrfache Instanziierung erlaubt es, einen schnelleren Simulationsdurchlauf zu erhalten.

Diese Aspekte zu Echtzeit und Skalierbarkeit in Verbindung mit der Festlegung auf Nutzung oder Verzicht auf Steuergeräte sollten bei der Gesamtkonzeption der Simulationsumgebung berücksichtigt werden. Aufgrund der Vielzahl an Testfällen im Zuge der Variation einzelner Parameter bietet sich eine SIL-Simulation an, was allerdings von der Zielsetzung abhängt.

7.6 Infrastruktur zur Toleranzanalyse in Consumer Tests - eine Fallstudie

Im vorangestellten Kapitel 6 wurde im Rahmen der Fallstudie dargestellt, aus welchen Gründen eine Toleranzanalyse bei Consumer Tests erforderlich und ein Äquivalenzklassentest für sich in diesem Kontext nicht ausreichend ist. Im Folgenden wird zunächst aufgezeigt, welche grundlegenden Entscheidungen im Rahmen der Infrastruktur-Realisierung zuvor zu treffen sind. Danach wird die Architektur näher beleuchtet und welche zusätzlichen Simulationskomponenten für diesen Anwendungsfall realisiert wurden.

Grundlegende Entscheidungen für die Architektur sind folgende Komponenten:

- Das SUT ist eine Blackbox Komponente.
- Als Grundstruktur wird eine Software-in-the-Loop Simulation gewählt.
- Auswahl eines idealen Sensormodells, welches einen Radarsensor abbilden soll.
- Aufgrund der Projektstruktur wird auf die Kaufsoftware VTD und ADTF als Basissysteme für Systemkontext und als Framework für die Systemkomponenten der Funktion zurückgegriffen.
- Die Entwicklung und das Zusammenspiel wird in Teilen durch einen externen Partner realisiert.

Im zugrundeliegenden Beispiel wird ein Algorithmus zur Auslösung von Bremseingriffen von einem externen Partner verwendet. Einzig die Ein- und Ausgangsdaten sowie das grobe Funktionsverhalten sind bekannt. Der Algorithmus selbst bestimmt in Abhängigkeit von der eigenen Geschwindigkeit und der umliegenden identifizierten Objekte auf mögliche Kollisionen und gibt zu unterschiedlichen TTCs Warnlevel heraus. [BHK+14] Die Entscheidung für eine SiL Simulationsumgebung ergibt sich aus der Zielsetzung, dass eine Evaluation des SUT über eine Parametervariation erfolgen soll: Dadurch ist es grundsätzlich möglich, die einzelnen Szenarien parallelisiert und ggf. schneller als in Echtzeit ablaufen zu lassen, sofern es die übrigen Komponenten erlauben. Die Wahl von idealen Sensormodellen wurde bereits oben allgemein diskutiert

und gilt hier im Besonderen. Die Wahl auf die beiden Software Werkzeuge VTD und ADTF erfolgt deshalb, da es zwei Werkzeuge sind, die in einer Vielzahl von Projekten bei Volkswagen verwendet werden und daher als nicht veränderbare Rahmenbedingungen zu integrieren sind. [BHK+14]

Die Gesamtarchitektur (siehe Abbildung 7.9) sieht folgende Komponenten vor:

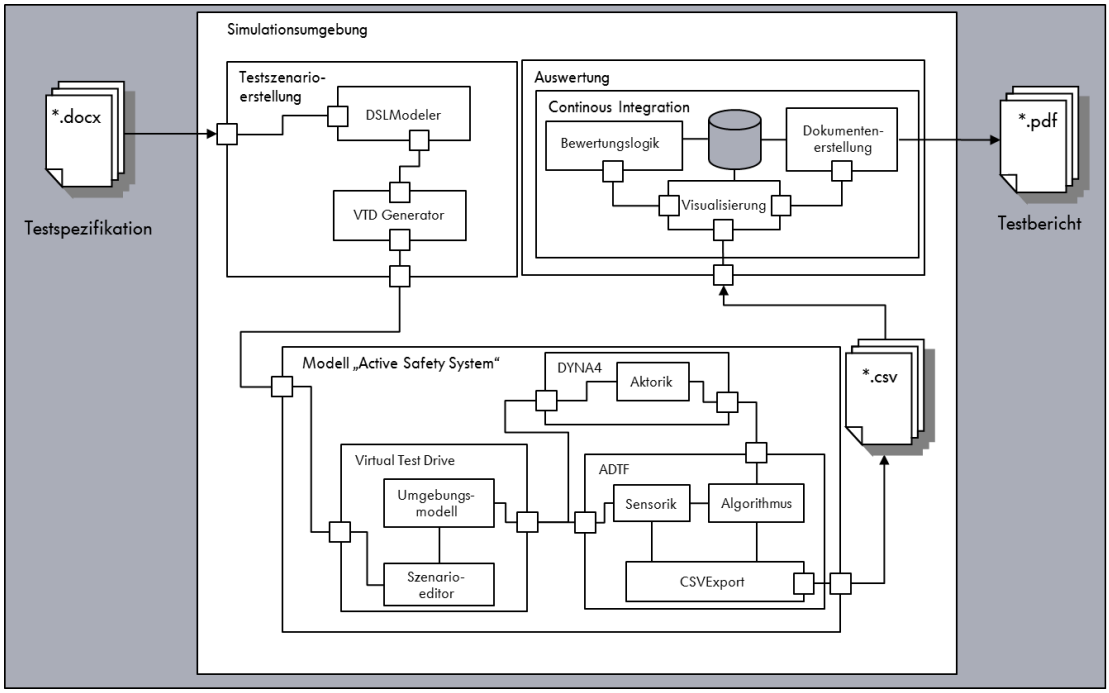


Abbildung 7.9: Infrastruktur zur Simulation von Consumer Tests (Eigene Darstellung)

Es gibt im Wesentlichen drei Teilsysteme, wie der Architektur in Abbildung 7.9 entnommen werden kann. Im ersten Teilsystem sind zunächst die Pre-Processing Komponenten zusammengefasst, die zum einen die Szenariengenerierung für die Grundszenarien vorsehen und gleichzeitig die Trajektoriengenerierung als Modell zur Parametervariation umfasst. Danach folgt das Teilsystem, in welchem das Modell zum Aktiven Sicherheitssystem abgebildet und später während der Simulation die Datenströme z.B. als Objektdaten für die Sensorik erzeugt. Das dritte Teilsystem umfasst die spätere Aus- und Bewertung der aufgezeichneten Datenströme. Das Teilsystem wurde bereits im obigen Abschnitt mit MontiCore und der Entwicklung der ScenarioDSL erläutert. Die konkrete Implementierung des ersten und zweiten Systems soll hier nun detaillierter erläutert werden.

7.6.1 Virtual Test Drive

Virtual Test Drive ist ein Werkzeug, dass ursprünglich von der Firma VIRES Simulationstechnologie GmbH für das Betriebssystem openSUSE Linux entwickelt wurde und aufgrund eines Verkaufs der Firma an die Fa. Hexagon nun von einem neuen Eigentümer gelenkt wird. Es verfügt über eine virtuelle 3D Umgebung, in welche Fahrzeuge, Fußgänger, andere Verkehrsteilnehmer, Gebäude, Straßen, Verkehrszeichen und das Gelände modelliert wurden. Die Umgebung ist modular aufgebaut, so dass zum Beispiel verschiedene Sensoren und Fahrdynamiken eingebunden werden können. Diese individuellen Module kommunizieren über den RDB, über den die beschreibenden Informationen der Objekte abgerufen werden können. Der SCP dient zur Kontrolle und Steuerung der späteren Simulation. [Aud14b, Gmb14] Weitere Details lassen sich aus den genannten Quellen entnehmen. ([NCDW09, NC14])

7.6.2 Automotive Data-and Time-triggered Framework (ADTF)

Das ADTF ist ein Framework, welches für die Entwicklung und den Test von Fahrerassistenzsystemen im Automobilbereich genutzt wird. Es erlaubt, eigene Komponenten innerhalb des Frameworks zu implementieren, welche Filter genannt werden. Diese können dann über Input und Output-Pins miteinander verbunden werden. Weiterhin kann man mit dem Framework diese Datenströme eines Fahrzeugs aufzeichnen und wieder abspielen, so dass sich dieses Framework, entwickelt von der Audi Electronic Ventures, als standardisierte Messtechnik etabliert hat. [Aud14a, Cor14]

Die Filter-Architektur der ADTF-Konfiguration sieht neben den notwendigen Filtern für die Funktion wie Sensormodell, den zu untersuchenden Algorithmus als eigenständige Einheit und den aufzeichnenden Filtern für die spätere Auswertung auch zwei Filter, die für die fahrdynamische Umsetzung verantwortlich sind. Der erste Filter liefert die Stimulationsdaten, die aus dem Modell der Parametervariation abgeleitet wurden. Der Filter zur Fahrdynamik sorgt dafür, dass das Fahrzeug an einer bestimmten Koordinate innerhalb der Simulationsumgebung während eines Simulationslaufes sein wird. [BBH+15a]

7.6.3 Ergänzende Kernkomponenten

Die beiden Software-Komponenten VTD und ADTF werden auf jeweils eine virtuelle Maschine installiert, die wiederum innerhalb der VirtualBox von Oracle laufen [Ora]. Als Betriebssystem

für die VTD Instanz muss ein Linuxsystem gewählt werden; in diesem Fall handelt es sich um eine Linux OpenSUSE Instanz. Die andere VM ist ein Windows System und beherbergt die ADTF Konfiguration inklusive des zu untersuchenden Algorithmus. Dabei handelt es sich um einen automatischen Notbrems-Assistenten, welcher Umgebungssituation dahingehend bewertet, ob es zu einer Kollision kommen kann oder nicht. Zudem entscheidet er im Vorfeld über mögliche Warnstufen, die an weitere Systemkomponenten signaltechnisch weitergereicht werden können.

Es wird davon ausgegangen, dass in Folge der umfangreichen Parametervariationen eine große Zahl von Testfällen simuliert werden müssen, die durchaus mehrere tausend umfassen können. Im gleichen Umfang werden auch Messdaten erzeugt, die zudem ausgewertet und in Relation zueinander gesetzt werden müssen. Um nun die Organisation dieser Daten zu ermöglichen, verfügt die Infrastruktur über Automatisierungskomponenten für die Ausführung von Simulationsläufen und die spätere Archivierung der Simulationsergebnisse. Dafür wurde ein Subversion Server eingerichtet und dahingehend konfiguriert, dass er automatisch den nächsten Simulationslauf beginnt, sobald ein neues Szenario im Repository verfügbar ist. Anschließend werden nach jedem spezifischen Testlauf alle erzeugten Simulationsdaten und Ergebnisse auf dem SVN Server abgelegt. Zudem erlaubt die SVN Struktur auch, dass ein Versionierungssystem auf Szenarien und Ergebnisse abgebildet wird, um evtl. direkte Vergleiche zwischen Softwareständen der untersuchten Komponenten durchführen zu können. Aber auch grundsätzliche Veränderungen an der Simulationsinfrastruktur wie z. B. Parameterveränderungen abseits der geplanten Variationen können so direkt miteinander verglichen werden. [BBH+15a]

7.6.4 Anwendung der Simulationsumgebung zur Toleranzanalyse von Consumer Tests

Die Szenarien werden, wie oben bereits beschrieben, in abstrahierter Form in der Scenario-DSL [BBH+14b] und mit Hilfe eines geeigneten Code-Generators als konkrete XML-Datei für VTD erzeugt. Mit Hilfe des Tools zur Trajektorien-Generierung wird eine weitere Datei erzeugt, welche weitere die jeweiligen Wegpunkte enthält. Außerdem wird noch eine weitere Datei mit den Eigenschaften des Szenarios erzeugt, wie z.B. die Positionen der einzelnen Objekte und ihre Geschwindigkeit sowie die Art des Szenarios. Durch die technologieunabhängige Beschreibung mit Hilfe von MontiCore [KRV07, KRV08, Voe11, HR17] ist es bei Austausch der Simulationstechnologie möglich, wie z.B. der Wechsel von VTD zu einem anderen Simulator, die erzeugten Modelle weiter zu verwenden, so dass nur der konkrete Code-Generator angepasst werden muss. Darüber hinaus können grundsätzlich die Generatoren sowohl für die Trajektorien als auch für die Grundszenarien so erweitert werden, dass sich neue oder veränderte Experimente mit jewei-

ligen Testfällen durchführen lassen. [BBH+15a]

Das Tool zur Trajektoriengenerierung erlaubt nun die Geschwindigkeit und die Gierrate des Fahrzeugs anzupassen, so dass es möglich ist, durch Veränderung dieser Parameter detailliertere Untersuchungen um Hotspots herum, z.B. durch Wahl einer feineren Auflösung, vorzunehmen. Eine genaue Beschreibung des Tools und seiner Anwendung erfolgt im nachfolgenden Abschnitt.

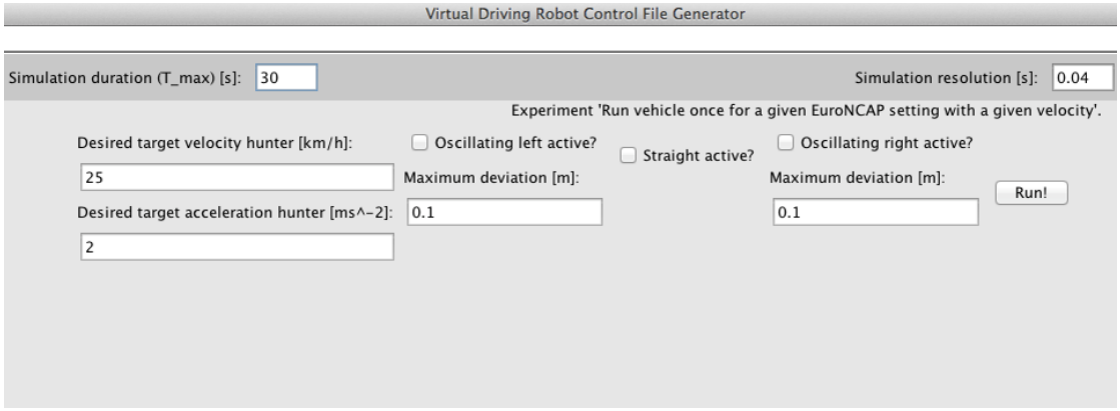


Abbildung 7.10: Testcase Generation Tool: Options to adjust CCRs Experiment

Jeder generierte Testfall wird innerhalb eines eigenen Verzeichnisses auf dem Windows-Rechner abgelegt. Nach dem Abschluss der Generierung aller erforderlichen Testfälle wird der Ordner mit samt seinen Dateien auf den SVN Server hochgeladen. Die ADTF-Konfiguration erkennt automatisch die neu hinzugekommenen Testfälle und startet jeweils einen Simulationslauf. Anschließend werden alle Ergebnisse aus der Simulation im SVN Verzeichnis abgelegt. Danach können die Ergebnisse ausgewertet und bewertet werden. [BBH+15a]

7.6.5 Nutzung der strukturierten Parametervariation

Nachfolgend sei die Funktionsweise des Tools zur Trajektoriengenerierung und die Nutzungsmöglichkeit der strukturierten Parametervariation näher erläutert. Dafür wird auf ein Grundszenario CCRs aus dem EuroNCAP mit dem Notbremsen auf ein stehendes Fahrzeug zurückgegriffen. Dabei gelten folgende Randbedingungen bzw. Annahmen:

- das Vehicle-under-Test steht an Position $p_{Hunter} = 0\ m$
- die Geschwindigkeit vom VUT beträgt $10\ km/h$
- die max. Verzögerungsleistung $-8\ m/s^2$

- das target vehicle befindet sich in Position $100\ m$
- das target vehicle hat eine Geschwindigkeit von $0\ km/h$
- die Geschwindigkeit des VUT wird in $0,1\ km/h$ Schritten bis $1\ km/h$ Abweichung variiert
- die Abweichung der seitlichen Ablage beträgt max. $0,1\ m$.

Ein Beispiel soll den obigen Zusammenhang und die möglichen Anzahl der Pfade aufzeigen. Dafür wird der Zeitpunkt t_E der Simulation bestimmt, zu welchem die entscheidenden Parametervariationen relevant werden. Aus dem Testprotokoll geht hervor, dass der entscheidende Punkt mit $TTC = 4s$ als Randbedingung aus dem Testprotokoll zum Punkt $t_E = 32,036\ s$ ohne eine Varianz in der Geschwindigkeit und t_E von $28,760\ s$ mit der maximal zulässigen Geschwindigkeitstoleranz liegt. Sofern man nun die max. Verzögerungsleistung berücksichtigt, ergibt sich daraus eine Zeitspanne von $1,25$ bis $1,28\ s$ je nach Geschwindigkeit bis zum Stillstand des auffahrenden Fahrzeugs. Diese Zeitspanne variiert selbstverständlich je nach Ausgangsgeschwindigkeit und kann evtl. zu einem Anprall zwischen dem VUT und dem target vehicle führen. [KHP+15]

Daraus folgt, dass bei einer Geschwindigkeit von $11\ km/h$ die Zeitspanne von $28,760\ s$ (resp. $TTC = 4\ s$) bis zum spätestmöglichen Bremspunkt mit $31,48\ s$ $2,72\ s$ beträgt und den interessierenden Variationsraum definiert. Es kann nun innerhalb dieser Sekunden eine Vielzahl von Pfaden unter Variation der Parameter modelliert werden, indem bspw. unter Berücksichtigung der maximalen Gierrate von $1\ °/s$ von der Ideallinie abgewichen wird. Durch die Wahl einer feineren Auflösung um den Faktor 4 ergeben sich für jede $250\ ms$ eine mögliche Veränderung der seitlichen Position. Da weiterhin sowohl eine seitliche Veränderung nach rechts als auch nach links sowie eine Fortsetzung der Richtung möglich ist, umfasst der Parameter drei mögliche Ausprägungen. Damit ergibt sich eine maximale Pfadanzahl von 3^{11} gleich 177.147 Pfaden, welche jedoch durch entsprechende Maßnahmen zur Komplexitätsbeherrschung reduziert werden kann. [KHP+15]

7.7 Zusammenfassung

In diesem Kapitel wurde aufgezeigt, wie die Infrastruktur eines Simulationsvorhabens von Aktiven Sicherheitssystemen fokussierter entwickelt werden kann. Es beginnt damit, dass die notwendigen Zielgrößen herausgearbeitet werden, die sich jeweils auf die Fragestellungen aus dem vorangegangenen Kapitel beziehen. Dies ist entscheidend für die funktionale Modellierung des

realen Systems innerhalb der Simulationsinfrastruktur. Weiterhin müssen konzeptionelle Vorüberlegungen getroffen werden, welche sowohl den Projektrahmen als auch wichtige Elemente der Softwarelösung später beeinflussen, z.B. ob es sich um eine Eigenentwicklung oder um vertriebene Software handelt. Nachdem diese Entscheidungen getroffen wurden, kann mit der Szenarioerstellung als Pre-Processing Baustein begonnen werden. Da die Szenarien generiert werden sollen, um Effizienzvorteile zu realisieren, wurde das MontiCore Framework eingeführt und eine Scenario-DSL auf Basis der xml-basierten Notation der VTD Szenarien entwickelt und implementiert. Neben diesem Pre-Processing Baustein wurde ein zweiter entwickelt, der es ermöglicht, die notwendigen Trajektorien der Fahrzeuge vorzugeben. Dieses Tool ermöglicht es unter Angaben von Toleranzbereichen in Geschwindigkeit, Gierwinkel, Startpunkt, etc, Pfade für die Szenarien in VTD als Parametrierung vorzugeben.

Weiterhin wurde auf die Modellierung der zu untersuchenden Funktion eingegangen und welche Aspekte bei der Modellierung berücksichtigt wurden. Dabei wird zunächst eine allgemeine Sicht eingenommen und später in der Fallstudie konkretisiert. Der Aspekt der Modellkomplexität wurde zuerst betrachtet: Dabei ist zwischen den beiden Punkten Nachbildung und simplifiziertem Modell sowie den Zwischenstufen zu wählen. Die Komplexität hängt von der jeweiligen Fragestellung ab. Auch muss entschieden werden, ob man eine Open-Loop oder Closed-Loop Konfiguration wählen möchte, wobei auch dies wieder vom Einsatzzweck und dem angestrebten Ergebnis abhängig ist.

In der Fallstudie wurde dann eine Infrastruktur zur strukturierten Parametervariation als Closed- und Software-In-The-Loop Simulation vorgestellt, die neben VTD und ADTF weitere Komponenten umfasst. Die Modelle innerhalb der Komponenten wurde bewusst vereinfacht gehalten. Dies umfasst sowohl das Sensormodell als auch das Aktormodell betrifft. Die Komponente zur Szenariengenerierung in Verbindung mit der Parametervariation von Trajektorien der Fahrzeuge erlaubt die automatische Erstellung von zahlreichen Testfällen zur systematischen Analyse von Entscheidungskomponenten innerhalb Aktiver Sicherheitssysteme.

Im anschließenden Kapitel 8 wird auf Möglichkeiten eingegangen, die Vielzahl von Simulationsdaten zielbringend auszuwerten und über einen Entwicklungsverlauf einer Fahrzeugkomponente Qualitätssicherung erzielen zu können.

8 Bewertungsverfahren

Mit Schaffung der Infrastruktur und der Prozessbeschreibung zur effektiven Realisierung einer Simulationsumgebung ist der zweite Baustein der hier vorgestellten Methodik abgeschlossen. Es umfasst ein System, dass aus mehreren Teilsystemen besteht und entsprechende Signale austauscht. Zusätzlich werden sämtliche Informationen zu Objekten, Trajektorien, Wegpunkten, usw. des zugrundeliegenden Umgebungsmodells bereitgestellt.

Es stellt sich jedoch die Frage, wie effektiv eine Überprüfung bzw. Kontrolle sowie eine Bewertung der Zielgrößen vorgenommen werden können, die zur Beantwortung der Fragestellungen aus der Analyse dienen. Daher wird nun in diesem Kapitel auf die Bewertungsverfahren von Signalen näher eingegangen, so dass entsprechende Interpretationshilfen für die Ergebnisse geschaffen werden. Es ist dabei wichtig, zwischen Basis-Metriken und übergeordneten Metriken, hier als Meta-Metriken bezeichnet, zu unterscheiden.

Dazu wird zunächst eine literarische Betrachtung zu Meta-Metriken vorgenommen, inwiefern diese bereits im Kontext von Aktiven Sicherheitsystemen bzw. auch im Rahmen von Simulationsaktivitäten zu diesen Systemen Verwendung gefunden haben. Wie noch gezeigt wird, konnte eine Verwendung von Meta-Metriken im genannten Kontext nach bestem Wissen des Autors nicht gefunden werden. Dies wurde auch bereits im Beitrag [BBH+13] festgestellt und hier erneut recherchiert.

Da auch keine Definition von Meta-Metriken identifiziert werden konnte, wird aufbauend auf dem Beitrag von [BBH+13] die Definition zu Meta-Metriken erweitert. Hierzu wird zunächst im Umfeld von Metriken geschaut, ob bereits konkrete Metriken existieren oder verwandte Definitionen gibt. Weiterhin wird auch auf die technische Erfassung von Signalen im Rahmen von Simulationsaktivitäten eingegangen. Anschließend wird der Unterschied zu den Meta-Metriken erläutert und welchem Zweck sie im beschriebenen Kontext dienen.

8.1 Literarische Betrachtung zu (Meta-)Metriken

Zu Beginn stellt sich die Frage, ob bereits spezifische Metriken bzw. Meta-Metriken für Aktive Sicherheitssysteme existieren und zum Einsatz kommen. Bereits im Beitrag [BBH+13] wurde hervorgehoben, dass dieses Thema bis zu jenem Zeitpunkt noch nicht umfänglich untersucht wurde. Bis zum jetzigen Zeitpunkt hat sich keine wesentliche Änderung ergeben. Nachfolgend sind daher eine Reihe von verwandten Beiträgen zu Metriken und Meta-Metriken aufgeführt, die als Ausgangsbasis dienen, um speziell diesem Aspekt mehr Bedeutung beizumessen.

Grundsätzlich ist das Themenfeld "Metriken" insbesondere in der Informatik nicht neu, so dass eine Vielzahl von Standardreferenzen vorhanden sind. Dazu zählen beispielsweise [LDA97, FP98, EFR07, Lig09, SSB10]. Sie befassen sich mit Prozessen zur Software Qualitätssicherung und dem Messen von Softwareentwicklung im Allgemeinen. Eine Einführung hierzu wird bei [Glo05] und [Aly06] angeführt.

Ein bedeutender Teil der Forschung beschäftigt sich insbesondere mit der Identifikation von neuen Metriken und ihrer Systematisierung. Dabei bildet der Ansatz, die richtige Metrik für den richtigen Zweck zur richtigen Zeit im Entwicklungsprozess zu liefern, einen Schwerpunkt und werden z.B. von Xenos et al. in [XSZC00] für objektorientierte Softwareentwicklung zusammenfassend aufbereitet. Im Zuge dieser Erstellung der Übersicht wurden auch Meta-Metriken verwendet, um die Bewertung von Grundmetriken zu verbessern.

Einen ähnlichen Ansatz verfolgen Stefani et al. ergänzend in [SX09], in welchem eine Reihe von Metriken in verschiedene Kategorien eingeteilt wurde, so dass daraus Meta-Metriken entwickelt werden konnten und so bei der Auswertung von E-Commerce Systemen unterstützten.

FLAME als Akronym für Formal Library for Aiding Metrics Extraction dient als Formalisierung von Definitionen objektorientierter Design-Metriken und wurde von Baroni und Abreu in [BA03] vorgestellt. Hierzu integrierten sie Teile der Object Constraint Language (OCL) als Funktionen in ihre FLAME Bibliothek und prüften hierzu ihre Methode anhand von verschiedenen Design Modellen. Am Ende präsentieren sie eine Zusammenstellung von entsprechend formalisierten Design-Metrik-Definitionen. Meta-Metriken finden hier jedoch keine Erwähnung.

Berger beschreibt in seinen Beiträgen zu Produktlinien, welches Potential Metriken bei bereits bestehender Software im industriellen Kontext bieten können. Dazu wird zunächst eine eigene Methode vorgestellt und dann anhand einer industriellen Fallstudie verifiziert. [Ber10, BRR10] Allerdings werden Meta-Metriken hier nicht explizit als Teil der Methode herausgearbeitet.

Woodings präsentiert in seinem Beitrag den Bedarf für mehr Messinstrumente innerhalb des

Entwicklungsprozesses von Softwareprojekten, um dort die Verbesserungsmöglichkeiten effektiver identifizieren zu können. Hierfür stellt er auch zwei Meta-Metriken vor: Erstere basiert auf deMarco's bestimmenden Qualitätsfaktor, welcher auf der schnellen Annäherung hin zu einer genauen Maßzahl innerhalb eines Projektes abzielt. Die zweite Meta-Metrik bietet eine niedrige Grenze für Fehler bei mehreren ersten Erfassungen von Metriken. [Woo99]

Flohr begründet in seinem Beitrag die Grundlagen zur Einführung von Quality Gates und einen Entwurf von geeigneten Kriterien, wobei Quality Gates Zeitpunkte im Entwicklungsprozess sind, an denen die Qualitätsanforderungen geprüft und ggf. Entscheidungen für Gegenmaßnahmen getroffen werden. Auch werden Aspekte vorgestellt, wie entsprechende Metriken verbessert werden können. [Flo08]

Weber et al. stellen in ihrem Beitrag nochmal die Wichtigkeit von Ansätzen heraus, die der übergeordneten Messung von Metriken und den zugrundeliegenden Messsystemen dienen. [WN10] Hierzu stellen sie einen Katalog an Anforderungen hinsichtlich Stabilität und Verständlichkeit bereit, die bei einer Messung von Software erfüllt sein sollen. Ihr Fokus liegt dabei auf Services und speziell web-basierten Services.

Anhand dieser Betrachtung konnte aktuell noch keine geeignete Definition für Meta-Metriken identifiziert werden, so dass ein eigener Ansatz für eine Definition vorgenommen wird. Hierzu wird zunächst auf Metriken eingegangen, um dann anschließend die Meta-Metriken aufbauend zu beschreiben, wobei einige Aspekte aus der Vorveröffentlichung in [BBH+13] aufgegriffen und weiter verfeinert werden.

8.2 Metriken

Schaut man in die Literatur nach Definitionen und Erklärungen hinsichtlich Metriken und ihren Stellenwert, so wird insbesondere der Begriff **der Maßzahl** als numerische Abbildung von software-spezifischen Eigenschaften hervorgehoben. Das Ziel einer (Software-)Metrik besteht darin, dass die Qualität des Softwareentwicklungsprozesses oder die Qualität des Softwareproduktes selbst über die Maßzahl bewertet werden kann, so dass im Bedarfsfall weitere Maßnahmen zur Qualitätsverbesserung angesetzt werden.

Der IEEE Standard 1061 definiert den Begriff der (Software-)Metrik wie folgt:

> *Eine Softwarequalität-Metrik ist eine Funktion, die eine Software-Einheit in einem Zahlenwert abbildet. Dieser berechnete Wert ist interpretierbar als der Erfüllungsgrad eine Qualitätseigenschaft der Software-Einheit.[Ass18]*

Darüber hinaus gibt es eine Reihe von unterschiedlichen Kategorien an Software-Metriken, die unterschiedliche Aspekte der Software-Entwicklung betrachten. Dazu zählen u.a.:

- Aufwands- und Kostenschätzung-Metriken
- Prozessmetriken
  - Fertigstellungsmetriken
  - Ressourceneinsatzmetriken
  - Ereignismetriken
- Produktmetriken
  - dynamische Produktmetriken
  - statische Produktmetriken
    - konventionelle (prozeduraler) Produktmetriken
    - objektorientierte Produktmetriken

Zu den Aufwand- und Kostenschätzung-Metriken gehören z.B. COCOMO-Methode, die Function-Point Methode oder die Delphi Methode. Die Prozessmetriken dienen im Rahmen des Qualitätsmanagements von Softwareentwicklung dem Controlling des zugrundeliegenden Entwicklungsprozesses. Dabei lassen sich drei Kategorien unterscheiden: Fertigstellungs-, Ressourceneinsatz- und Ereignismetriken. Ersteres zielt darauf ab, dass innerhalb eines (Teil-)Prozesses die Zeit bis zur Fertigstellung erfasst wird. Bei den Ressourceneinsatzmetriken werden die Mittel an Finanzen, Mitarbeitern oder Maschinen während der Entwicklung gemessen. Letztere misst im Wesentlichen die Anzahl an eintretenden Ereignissen wie aufgedeckte Fehlerwirkungen oder die Anpassungen bzw. Ergänzung von Anforderungen. ([ITW18])

Dynamische Produktmetriken dienen der Erfassung der Leistungsfähigkeit und Zuverlässigkeit von Software, wobei als Beispiele hier die Laufzeit einer Software betrachtet bzw. die Anzahl der auftretenden Fehlerwirkungen angeführt wird. Statische Produktmetriken hingegen zielen auf die Bewertung von Entwurf, dem Programm oder der Dokumentation ab, wobei dabei hinsichtlich Komplexität und Wartbarkeit gemessen wird.

Konventionelle Produktmetriken haben statistischen Charakter und werden in vier Gruppen unterteilt:

- Umfangsmetrik
- Logische Strukturmetriken
- Datenstrukturmetriken
- Stilmetriken

Beispiele für Umfangsmetriken sind Lines of Code (LOC) oder Halstead. Die logischen Strukturmetriken messen, welchen Grad der Verschachtelungen ein Programmteil hat oder wie viele Pfade einer Methode durchlaufen werden. Eine weitere konventionelle Metrik befasst sich mit den erzeugten Daten wie z.B. die Anzahl an verwendeten Variablen, ihre Lebensdauer und eine mögliche weitere Verwendung. Bei einer Stilmetrik liegt der Fokus auf Kommentaren und der Einhaltung von Namenskonventionen. Spezielle objektorientierte Metriken ergänzen die konventionellen Metriken durch Messung von typischen objektorientierten Aspekten wie z.B. Komplexität von Methoden einer Klasse, Grad von Vererbungen oder auch den Grad der Verknüpfung von Klassen. [ITW18]

Anhand dieses Kataloges wird deutlich, dass es vielfältige Einsatzmöglichkeiten von Metriken gibt, die sich entweder mit Kosten- und Aufwandsschätzungen befassen oder auf die Qualität des Entwicklungsprozesses bzw. der softwaretechnischen Umsetzung beziehen. In diesem Kontext steht jedoch ein anderer Aspekt im Fokus, der anhand dieser Kategorien von Metriken nicht eindeutig zugeordnet werden kann: Die Messung und die Beurteilung der Qualität, die sich aus der Funktionalität der entwickelten Software ergibt, also “wie gut” eine Softwarekomponente ihre Aufgabe erfüllt. Die bisher hier genannten Kategorien von Metriken können diese Aufgabe jedoch nicht übernehmen, so dass eine eigene Definition einer Metrik formuliert wird.

8.2.1 Das Goal-Question-Metric Paradigma (GQM)

Das GQM-Paradigma von Basili et al., veröffentlicht 1994 und in weiteren Veröffentlichungen bis in die heutige Zeit aktualisiert, befasst sich speziell mit der Aufgabenstellung, entsprechende Instrumente für die Qualitätsbewertung zu entwickeln. Mit Hilfe des GQM Paradigmas sollen im Rahmen des Softwareentwicklungsprozesses messbare Ziele definiert und in die vorhandenen Qualitätsmodelle integriert werden. Dafür wird ein Top-Down-Ansatz gewählt, weil eine effektive Messung erst durch die Zielsetzung im Rahmen des jeweiligen Projektkontextes ermöglicht wird, auf welche die Messung ausgerichtet sein muss. Sie bezieht sich in der Regel auf die Produkte, Prozesse und Ressourcen innerhalb des jeweiligen Lebenszyklus. [BCR94] Es können in dem System drei Ebenen unterschieden werden:

1. Konzeptionelle Ebene (Ziele)
2. Operationelle Ebene (Fragen)
3. Quantitative Ebene (Metriken)

Auf der konzeptionellen Ebene werden Ziele definiert und beziehen sich in der Regel auf ein Objekt. Ihre Definition erfolgt dabei wohlbegründet und ausgerichtet auf ein Qualitätsmodell und sie verfügen über mehrere Sichtweisen innerhalb eines bestimmten und abgrenzbaren Kontextes. Entsprechende Objekte können sein:

- Produkte
- Prozesse
- Ressourcen

Produkte umfassen beispielsweise Deliverables, Artefakte und alle Arten von Dokumenten, die innerhalb des jeweiligen Lebenszyklus entstanden sind. Prozesse beschreiben die eigentlichen Aktivitäten im Umfeld von Produkten, wie z.B. das Entwerfen, das Testen, aber auch Beschaffen von Informationen, die für weitere Aktivitäten verwendet werden. Ressourcen werden für das Ausführen der Aktivitäten in den Prozessen benötigt. Das können bspw. Hard- und Software aber auch Räumlichkeiten sein.

Auf der operationellen Ebene werden die Fragen erarbeitet, welche den Weg der Zielerreichung aufzeigen sollen. Dabei kann das zu messende Objekt anhand der bisherigen Erkenntnisse über sein Verhalten als Modell charakterisiert werden. Durch die Beantwortung der Fragen können dann weitere Charaktereigenschaften genauer aus der jeweiligen Sichtweise spezifiziert werden.

Auf der quantitativen Ebene werden die Metriken den einzelnen Fragen zugeordnet, so dass ihre Beantwortung auf einer quantitativen Basis erfolgen kann. Metriken können dabei einen

- objektiven Charakter oder
- subjektiven Charakter

besitzen. Einen objektiven Charakter erhält eine Metrik, wenn sie rein vom erfassten Messwert bezüglich des Objektes abhängig ist und eine andere Sichtweise diesen Messwert nicht beeinflusst. Klassische Beispiele sind hier Entwicklungszeit, Lines of Code, Fehleranzahl, etc.; dabei handelt es sich um absolute Messwerte von Produkten oder Prozessen. Hingegen kann eine möglicherweise zu berücksichtigende Sichtweise eines Messwertes oder auch im Falle einer nicht exakten Messung die Metrik einen subjekten Charakter erhalten, da nun individuelle Aspekte der Perspektive hinzukommen. Üblicherweise sind das nominale oder ordinal skalierte Werte,

wie z.B. die Erfahrung eines Programmierers mit einer Programmiersprache oder der Nutzungsgrad einer Methode. Somit unterliegt die Metrik einem Einfluss von außen und wird durch eine oder mehrere Personen festgelegt. [BCR94]

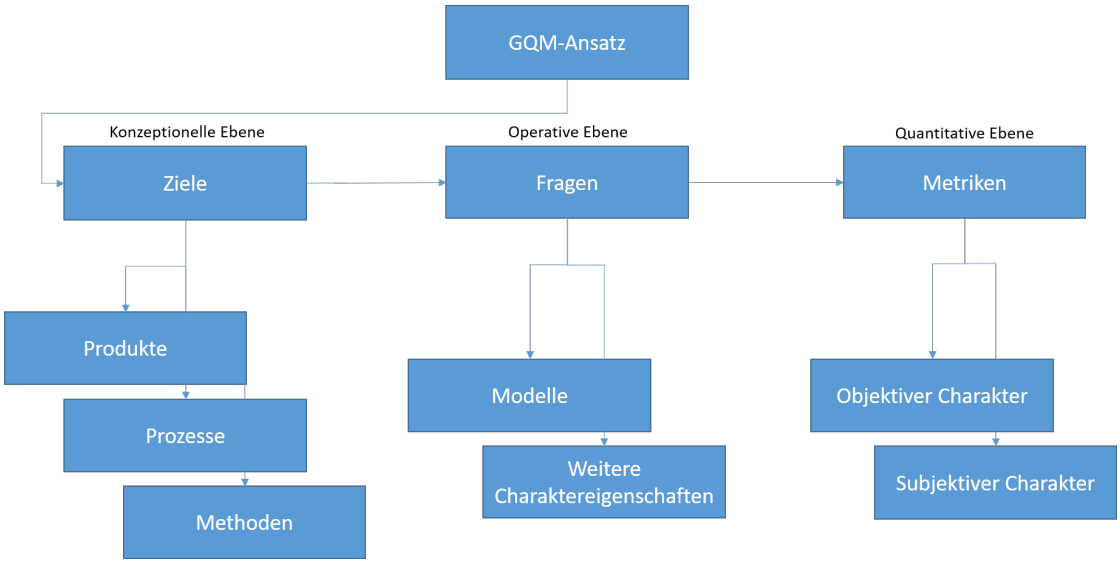


Abbildung 8.1: Illustration zur GQM-Methodik (Eigene Darstellung)

Insgesamt kann dieses Gefüge aus Zielen, Fragen und Metriken als ein gerichteter Graph (vgl. Abb. 8.1) dargestellt werden. Ziele dienen dabei auf der obersten Ebene als Knoten und verzweigen in die nächste Ebene der Fragen, wobei eine einzelne Frage dazu dienen kann, mehr als ein Ziel zu beschreiben. Um die Fragen beantworten zu können, werden quantifizierbare Metriken oder Verteilungen herausgearbeitet und als Knoten in der darunterliegenden Ebene aufgeführt. Allerdings können entsprechende Metriken bei der Beantwortung mehrerer Fragen beitragen, auch wenn sie dadurch auf andere Ziele letztlich verweisen. Die Fragen basieren auf den grundlegenden Modellen, welche den (System-)Kontext definieren. Je formaler die Modelle für den Kontext sind, desto formaler können die Fragen und auch die Metriken am Ende sein. [BCR94]

Für die Methodik dieser Arbeit soll die GQM-Methode aufgegriffen und adaptiert werden, indem sie in den vorliegenden Zusammenhang eingefügt und an möglicherweise anzupassenden Stellen verändert wird.

8.2.2 Funktionsmetriken - eine eigene Definition

Die Notwendigkeit einer eigenen Definition wurde bereits aufgezeigt. Die Namensgebung als Funktionsmetrik leitet sich von dem Fokus ab, auf welchen Messbereich abgezielt wird: Die Qualitätsmessung der Funktionsweise der Software. Dies ergibt sich daraus, dass eine entsprechende Software-Komponente von mehreren Herstellern je nach Auftragsvergabe implementiert werden kann. Zudem kann die Software-Komponente als Black-Box betrachtet werden kann, so dass keine weitere Einsicht auf den Code zu erlangen ist und daher eine andere Art von Qualitätsmessung erfolgen muss als im Vergleich zu einer 100%-Verfügbarkeit des Quellcodes.

Das Ziel dieser Art von Metriken besteht darin, eine Aussage über die Qualität der Arbeitsweise und der Funktionalität einer Software zu machen und dabei nicht wie bei dynamischen Produktmetriken auf das Laufzeitverhalten oder die Anzahl reiner Fehlerwirkungen abzuzielen. Diese Metrik eignet sich speziell für Personen wie Auftraggeber von Fahrzeugfunktionen und in diesem Zuge an dazugehörige Tester, welche möglicherweise zur Abnahmefähigkeit von solchen Funktionen berichten.

Daher wird unter dem Begriff Funktionsmetrik folgende Bedeutung definiert:

> *Eine Funktionsmetrik ist eine errechnete Größe, die sich aus einem oder mehreren Signalen oder Messkanälen aufgrund arithmetischer Operationen miteinander verknüpfen lassen, so dass sich ein neuer, interpretierbarer Wert ergibt.*

In der Regel werden gerade im Automobilbereich eine ganze Reihe von Tests und Versuchen unternommen und Messungen dienen dabei als Grundlage zur Beurteilung der Tests. Solche Messprotokolle werden durch den zuständigen Entwickler oder Tester interpretiert und eine Erkenntnis abgeleitet, in welcher Qualität eine bestimmte Fahrzeugfunktion vorliegt. Diese Arbeitsschritte erfolgen durchaus implizit, dass heißt, der Entwickler verfügt über die Erfahrung, wie er einen Messverlauf von Signalen zu interpretieren hat. Durch die Funktionsmetriken wird ein Instrument an die Hand gegeben, wodurch dieser Erkenntnisgewinn explizit gemacht wird. Eine große Herausforderung besteht jedoch darin, solche Funktionsmetriken in ihrem jeweiligen Systemkontext abzuleiten, also Qualitätsmerkmale für ein “gutes” Funktionieren von Software bzw. der Fahrzeugfunktion festzuhalten.

8.2.3 Technische Erfassung von Signalen

In Kapitel 7 wurde beschrieben, wie eine Infrastruktur aussehen kann, damit die für den jeweiligen Einsatzzweck eine geeignete Simulationsarchitektur existiert. Als Teil dessen ist zu

sehen, dass die Signale, die zwischen den Komponenten des Simulationsmodells ausgetauscht werden, für eine weitere Auswertung und Bewertung aufgezeichnet werden. Die Ausführungen zu den technischen Aspekten der Bewertung werden nun im folgenden beschrieben, um eine geschlossene Betrachtung des Aspektes innerhalb der Methodik zur Simulationsentwicklung zu gewährleisten.

Bevor nun auf die technischen Aspekte eingegangen wird, sollen noch einige Anforderungen aufgeführt werden, die aus Sicht des Autors dieser Arbeit dahingehend wichtig sind, da sie Flexibilität des Entwicklungsingenieurs oder des Testers gewährleisten. Dazu zählen:

1. Leichtgängigkeit dieses Teilsystems zur Bewertung
2. Auswertungsframework muss über eine hohe Verbreitung verfügen
3. Wartung und Pflege muss jedem inkl. denjenigen Stakeholdern möglich sein, die nicht eine informatik-basierte Ausbildung genossen haben
4. Framework muss arithmetische Operationen und idealerweise über mathematische Funktionen verfügen.

Unter der Leichtgängigkeit ist hier die Eigenschaft zu verstehen, dass für einen industriellen Kontext ein Format für die Messergebnisse und die technische Umsetzung ihrer Erzeugung verwendet wird, die für jeden Mitarbeiter leicht verständlich ist und auch weniger erfahrenen Entwicklungsingenieuren und Testern die Möglichkeit bietet, den Aufbau zu verändern bzw. zu erweitern.

Das Auswertungsframework sollte daher ebenfalls über einen hohen Verbreitungsgrad verfügen und die Anforderungen der Unternehmung in gewisser Weise berücksichtigen können. Häufig sind lizenzfreie Softwarelösungen von einem aufwendigen Prüfverfahren hinsichtlich Softwaresicherheit betroffen, so dass dies bei einer evtl. Nutzung im Projektverlauf mit eingeplant werden sollte. Auch zusätzliche Lizenzkosten müssen hinsichtlich Genehmigung in Form von weiteren Investitionskosten betrachtet werden. Daher kann es durchaus ratsam sein, auf bereits weitverbreitete Softwarelösungen zurückzugreifen, auch wenn diese aus funktionaler Sicht nicht immer der optimalen Lösung entspricht. Hier muss jedoch immer eine Einzelfallentscheidung vorbereitet und getroffen werden. Als Beispiel kann hier der Vergleich Microsoft Excel versus Mathworks Matlab genannt werden. Excel ist möglicherweise nicht so mächtig, was die Aufbereitung von Daten angeht, ist jedoch üblicher Weise auf fast jedem Mitarbeiter Rechner einer Unternehmung zu finden. Matlab bietet eine sehr umfassende Vielfalt an mathematischen Funktionen und Operationen sowie zur spezialisierten wissenschaftlichen Auswertung, ist jedoch auch in seiner Handhabung und der Anschaffung durchaus mit einem höheren Aufwand

verbunden als MS Excel.

Die Entwicklung von Metriken erfolgt immer innerhalb eines spezifischen Systemkontextes und ist daher von Anwendungsfall zu Anwendungsfall verschieden. Daher wird bereits zu Anfang aller Entwicklungsaktivitäten einer geeigneten Simulationsumgebung umfassend auf die Ausarbeitung von Fragestellungen, welche die Simulationsläufe beantworten sollen, Wert gelegt.

Im nächsten Schritt müssen dann für die Ausarbeitung von geeigneten Metriken als Grundlage zur Beantwortung der Fragestellungen entsprechende Signale innerhalb des abgebildeten Systems ausgewählt werden, welche später gemessen werden sollen. Darauf aufbauend können weitere abhängige Variablen entwickelt werden, die aufgrund von arithmetischen Verknüpfungen bereits weitere Beurteilungen zulassen. Eine innerhalb des Projekts zu klärende Frage besteht darin, ob die Bedatung der eigens entwickelten Metriken bereits im Rahmen der Simulationsrohdatenerzeugung in die Ergebnisdateien geschrieben werden oder erst im Anschluss während der Aufbereitung und Auswertung hinzugefügt werden. Als Vorteil für die erste Variante spricht, dass bereits alle bewertbaren Daten vorliegen und in der Ergebnisdatei zeitlich korrespondierend abgespeichert werden können, so dass keine weiteren Arbeitsschritte vorgenommen werden müssen. Das birgt allerdings auch den Nachteil, dass bei Nutzung der Rohdaten evtl. auch in anderen Projekten und Unternehmen die Metriken bekannt sind. Da gerade Metriken ein gewisses Know-How einer Firma dahingehend vertreten, welche Erkenntnisse aus einem Testprojekt gewonnen werden können, sind sie hinsichtlich Zugänglichkeit an Dritte besonders zu schützen. Daher hat die zweite Variante als Vorteil, dass die Rohdaten als solche in sich geschlossen bleiben und somit keinen Rückschluss auf mögliche Auswerteergebnisse erlaubt. Nachteilig daran ist, dass weitere Arbeitsschritte im Nachgang notwendig sind, ganz gleich, ob sie manuell oder automatisiert erfolgen.

Auch muss festgelegt werden, ob nicht im akuten Moment der Aufzeichnung nicht schon Messwerte z.B. gefiltert werden oder erst im Nachgang nach Rohdatenerhebung angepasst werden. Hier kann auch die Betrachtung helfen, dass reine Rohdaten unverfälscht sind und somit die Aufbereitung je nach Erkenntnissen weiter verfeinert und auf die Rohdaten angewendet werden können. Somit müssen nicht erneut Simulationsläufe durchgeführt werden.

Basierend auf der Methode von Basilli und Weiss “Goal-Question-Metric” besteht die Herausforderung darin, abgeleitet von den Fragen zur Erreichung der Ziele geeignete Signale den quantifizierten Metriken zuzuordnen. Auf folgende Fragen soll nun weiter eingegangen werden:

1. Auf welche Weise kann dies zielführend erfolgen?
2. Was gilt es dabei zu beachten?

Die Einführung einer Funktionsmetrik als spezielle Metrik für die Analyse und Bewertung von Aktiven Sicherheitssystemen definiert den Verwendungszweck und erlaubt Rückschlüsse auf den Kontext. Im Zentrum dieser Metriken steht die Erfassung und Bewertung von Funktionen, welche durch die einzelnen Systemkomponenten realisiert werden.

Da jede Frage einem Ziel zugeordnet ist bzw. eine Menge an Fragen zu einer Zielerreichung führen, müssen auch mehrere Metriken in einen Zusammenhang gesetzt werden. So besteht ebenfalls die Möglichkeit, dass eine Metrik sich aus mehreren Signalen zusammensetzt, die erfasst werden. Da dies immer einen konkreten Projekthintergrund mit konkreten Zielen und daraus abgeleiteten Fragen voraussetzt, damit eine geeignete Aggregation von Signalen erfolgt, kann an dieser Stelle zunächst nur eine abstrakte Methode definiert und beschrieben werden. Diese abstrakte Methode muss dann im konkreten Einzelfall weiter verfeinert und vervollständigt werden.

Die Methodik lautet dann wie folgt:

1. Sofern keine Ziele entwickelt wurden, entwickle geeignete Ziele zur Bewertung der Simulation.
2. Sofern noch keine Fragen herausgearbeitet wurden, leite entsprechende Fragen korrespondierend zu den Zielen ab.
3. Entwickle zu jeder Frage eine Metrik wie folgt
   a) Prüfe, welche Signale einen Bezug zur Frage haben
   b) Wähle das Signal aus, welches die Frage am ehesten beantwortet.
   c) Lege Bewertungskriterien fest, ab welchem Wert die Frage positiv und negativ beantwortet wird.
   d) Wenn ein Signal nicht ausreichend für die Beantwortung ausreichend ist, so füge weitere Signale der Metrik mit arithmetische Operatoren hinzu.
4. Beende die Zuordnung, wenn alle Fragen sowohl positiv als auch negativ beantwortet werden können.

Wegen der spezifischen Projektsituation ist dies aus Sicht des Autors der kleinste gemeinsame Nenner zur Vorgabe einer Systematik, auch wenn sie hier noch als abstrakt und weniger konkret formuliert ist.

Bei Metriken und insbesondere beim Einsatz von Simulationsumgebungen zur agilen Qualitätssicherung spielt die zeitliche Dimension der Betrachtungzeiträume eine gewichtige Rolle. Nach

eigener Auffassung können sich dabei drei Kategorien von Zeithorizonten ergeben:

1. operative Dimension
2. taktische Dimension
3. strategische Dimension

Die operative Dimension zielt klar mit den jeweils eingesetzten Metriken auf die unmittelbaren Aussagen eines einzelnen Simulationslaufes ab. Somit ist der Zeitpunkt der Simulationsdurchführung der Fixpunkt. Der zeitliche Ablauf eines Simulationslaufes ist davon unberührt. Die taktische Dimension ergibt sich aufgrund der zeitlichen Ausdehnung eines vollständigen Simulationslaufes aller vorgesehenen Testfälle. Da dies eine gewisse unspezifische Zeit in Abhängigkeit der Anzahl der Testfälle, der Komplexität der Modelle und des notwendigen Rechenaufwandes einnimmt, bis am Ende für alle Testfälle ein Simulationsergebnisses vorliegt, kann die dafür notwendige Zeit von Stunden bis hin zu Tagen und Wochen betragen. Eine Möglichkeit zur Steuerung dieser Dimension kann sich aus einer Parallelisierung der Simulationsläufe pro Testfall ergeben. Dadurch sind Beschleunigungen des Ablaufes in Abhängigkeit der vorhandenen Ressourcen und insbesondere der Rechnerkapazitäten realisierbar. Die strategische Dimension der zeitlichen Betrachtung zielt auf die Bewertung von verschiedenen Simulationsläufen untereinander ab und erlaubt damit eine Bewertung der Softwarequalität über den Projektverlauf. Wie grundsätzlich dafür verfahren wird und wie die Anwendung später innerhalb der Fallstudie erfolgt, wird in nachfolgenden Abschnitten beschrieben.

8.3 Meta-Metriken

Dieser Abschnitt befasst sich speziell mit der langfristigen Perspektive der Qualitätssicherung von Sicherheitsfunktionen. Die hier erläuterten Meta-Metriken dienen genau diesem Zweck, über einen Zeitraum bzw. über mehrere Simulationsläufe zu unterschiedlichen Zeitpunkten eine Bewertung der Funktionalität vornehmen zu können. Die bisherigen Aspekte zu Metriken und ihre Entwicklung hingegen haben zum Ziel, die Qualität einer Software-(funktion) während eines Simulationslaufes zu bestimmen bzw. zu beurteilen. Das Ziel zur Etablierung dieses Bewertungs- und Kontrollinstruments besteht darin, dass nicht nur zu einem jeweiligen Zeitpunkt eine qualitative Beurteilung des Entwicklungsstandes einer Funktion erfolgen muss, wozu eine Metrik dient, sondern auch über den Entwicklungsverlauf eine Bewertungsaussage zur Qualität machen zu können.

8.3.1 Meta-Metriken als Methodik für die Qualiätssicherung von Aktiven Sicherheitssystemen

Bereits in dem Beitrag [BBH+13] wurde der Methodik zugrundeliegende Begriff "Meta-Metrik" eingeführt, ist jedoch im damaligen Zustand losgelöst von einer eigentlichen Entwicklung von zugrunde liegenden Metriken betrachtet worden. Vielmehr wurden die Metriken als gegeben angenommen und anschließend zu Meta-Metriken erweitert. Durch den vorherigen Abschnitt ist diese Lücke nun zunächst gefüllt worden.

Der Begriff "Meta-Metrik" beschreibt ein methodisches Werkzeug, dass im Rahmen der Entwicklung von Aktiven Sicherheitssystemen verwendet wird und den Projektbeteiligten wichtige Informationen bereitstellt, in welcher Qualität sich eine Komponente oder ein Artefakt über seinen Entwicklungszeitraum befindet. Dadurch wird es möglich, die für die Entwicklung eingesetzten Ressourcen und Projektteilnehmer zielgerichteter dort einzusetzen, wo noch weitere Verbesserungen in der Qualität in Hard- oder Software erfolgen müssen. [BBH+13]

Daher wird die Funktionsmetrik, die bereits im Abschnitt zuvor definiert worden ist, durch die Definition der Meta-Metrik in seinem Anwendungsspektrum erweitert. In Berger et Kühnel [BBH+13] wurde bereits eine Definition genannt, da nach bestem Wissen der Autoren eine solche Definition bzw. eine solche Methodik vor einem solchen Kontext noch nicht veröffentlicht wurde. Die damalige Definition lautet:

> *The continuous determination of quantitative figures, which are defined over a set of results of simulation and test runs carried out for specific aspects, to steer and optimize the development process for an increased quality of the resulting product.* [BBH+13]

Zwar wird in dieser Definition auf die kontinuierliche Bestimmung hingewiesen, jedoch noch nicht genauer spezifiziert, worauf sich diese Kontinuität bezieht. Durch die Beschreibung von drei zeitlichen Dimension im Rahmen von Funktionsmetriken ist eine Präzisierung dieser Definition möglich. Eine weitere Präzisierung kann hinsichtlich der Menge von Ergebnissen erfolgen: Da sich Ergebnisse nicht nur auf direkte Messergebnisse sondern auch auf gemachte Bewertungen beziehen können, wie z.B. bei der GQM Methodik auf Ziele und daraus abgeleitete Fragen, wäre eine Unterscheidung an dieser Stelle hilfreicher und könnte zudem den Einsatzzweck genauer beschreiben.

Daher wird die vorherige Definition hier nun erweitert:

> *Meta-Metriken umfassen die strategische, zeitlich-kontinuierliche Bestimmung von quantitativen Größen abgeleitet aus Funktionsmetriken, die über eine Menge von Simulations- und Testläufen mit unterschiedlichen Zielsetzungen und Fragestellungen ausgeführt werden, um den Entwicklungsprozess für eine verbesserte Qualität des untersuchten Produktes zu steuern und zu optimieren.*

Die Methodik zur Entwicklung kann dann in Ergänzung zur Entwicklung der Funktionsmetriken wie folgt definiert werden:

1. Prüfe, ob mehr als ein Simulationslauf mit Funktionsmetriken vorhanden sind
2. Wiederhole für alle zu bestimmenden Funktionsmetriken
   a) Wähle eine Funktionsmetrik aus, die noch nicht als Meta-Metrik definiert wurde.
   b) Verknüpfe sie mit einem geeigneten Operator (z.B. "-")
   c) Gebe diese als neue Meta-Metrik aus, welche den gleichen Namen der Funktionsmetrik und als Indizes die beiden Zeitpunkte enthält.
   d) Definiere einen Erkenntnisgewinn für den negativen und den positiven Wertebereich der Meta-Metrik.
3. Erhebe zu jedem Simulationslauf die Meta-Metriken.
4. Prüfe, ob weitere Ziele und Fragestellungen die bisherige Erfassung ergänzen können.

Die Grundvoraussetzung für die Entwicklung von Meta-Metriken besteht in den zugehörigen Funktionsmetriken, da diese später in eine Beziehung zueinander gesetzt werden. Danach wird für jede bereits existierende Funktionsmetrik wiederholend, bis alle Funktionsmetriken bearbeitet wurden, wie folgt behandelt: Zunächst wird eine Funktionsmetrik ausgewählt. Dabei ist es unerheblich, ob es bereits konkrete Werte zu verschiedenen Simulationsläufen gibt oder ob die Funktionsmetrik noch ohne Wert und somit vor dem ersten Simulationslauf existiert. Somit kann eine Meta-Metrik auch als nachträgliches Werkzeug in eine bestehende Metriken-Karte integriert werden oder aber auch bereits zu Beginn eines neuen Simulationsprojektes. Danach verknüpfe die Funktionsmetrik-Instanzen mit einem geeigneten Operator. Hier ist zunächst der Minus-Operator als Beispiel angegeben, es kann jedoch auch ein anderer (arithmetischer) Operator gewählt werden. Die Wahl ist letztlich vom konkreten Ziel und der Fragestellung abhängig, welche einen Grenzwert und damit einen Wertebereich darüber bzw. darunter definiert, der dann

Aufschluss über eine Qualitätssteigerung oder -verschlechterung impliziert. Voraussetzung dafür ist jedoch, dass Simulationsläufe zu unterschiedlichen Zeitpunkten durchgeführt werden und das SUT ebenfalls in zwei Versionen vorliegt. Die Bezeichnung der Meta-Metrik orientiert sich anschließend an der zugehörigen Funktionsmetrik. Das Entscheidungsmerkmal ist dann jedoch die Indizierung der Zeitpunkte oder Revisionen. Die Erhebung erfolgt kontinuierlich zu jedem Simulationslauf. Weiterhin wird die Möglichkeit der Anpassung gegeben, dass sich ein solches Entwicklungsprojekt auch einer stetigen Verbesserung unterziehen kann.

8.3.2 Zur Notation und Lesart von Meta-Metriken

Im Beitrag von [BBH+13] wurden bereits mehrere Meta-Metriken definiert und erläutert. Eine Wiederholung dieser soll nicht erneut erfolgen, gleichwohl wird anhand eines Beispiels dieser Meta-Metriken die Notation und ihre Lesart beschrieben, um einen Standard für die Ausarbeitung weiterer Meta-Metriken anzubieten. Dabei wird sich auf die Metrik "Anzahl an erfolgreichen Tests" bezogen. N repräsentiert die Anzahl an individuell und eindeutig identifizierbaren Versionen eines Entwicklungsartefakts wie z.B. die Anzahl an Revisionen innerhalb eines Repositorys. Ein Entwicklungsartefakt kann entweder eine Softwarekomponente, ein Teil davon oder ein Implementierungsmodell einer Fahrzeugfunktion sein. Mit der Funktion $res(r, i)$ wird die Bewertung einer Testfalls ausgedrückt, wobei r für das Artefakt und i für Revisionsnummer steht. Die Funktion ist $false$, wenn ein Fehler im SUT auftritt, sonst ist das Ergebnis $true$. [BBH+13]

$$
\begin{aligned}
R_{\textit{erfolgreich}}(r, N) &= \sum_{i=n_0}^{N} \textit{res}(r, i) \qquad (8.1)\\
\text{wo } \textit{res}(r, i) &= \begin{cases} 1 & \text{wenn Artifakt } r \text{ zur Revision } i \text{ erfolgreich getestet,} \\ 0 & \text{sonst.} \end{cases}
\end{aligned}
$$

$$
R_{\textit{fehlerhaft}}(r, N) = N - R_{\textit{erfolgreich}}(r, N)
$$

Eine zugehörige Meta-Metrik kann dann beispielsweise mit folgender Gleichung 8.2 bestimmt werden.

$$
\begin{aligned}
R^{+}(r,N) &= \frac{R_{\mathit{succeeded}}(r,N)}{N}, \\
R^{-}(r,N) &= 1 - R^{+}(r,N), \\
\\
Q_1(r,N_1,N_2) &= R^{+}(r,N_2) - R^{+}(r,N_1) \text{ where } N_1 \leq N_2. \qquad (8.2)
\end{aligned}
$$

Sie beschreibt, in welcher Rate erfolgreich getestete und fehlerhaft getestete Artefakte innerhalb eines betrachteten Entwicklungszeitraums auftreten. Durch diese Art von Meta-Metriken können "Heatmaps" erzeugt werden, welche Anomalien innerhalb von Entwicklungsartefakten aufzeigen. Weiterhin kann über Q_1 bestimmt werden, wie erfolgreich Simulationsläufe zu einem Entwicklungsartefakt über zwei Entwicklungszeiträume durchgeführt wurden: Ist Q_1 positiv oder gleich null, so hat die Qualität des Artefaktes in der Zeit nicht abgenommen.

Weitere Meta-Metriken wurden wie bereits eingangs erwähnt im Beitrag von [BBH+13] beschrieben, so dass an dieser Stelle auf eine erneute Darstellung verzichtet wird.

8.3.3 Anwendung von Meta-Metriken auf V-Modell

Im Zuge der Entwicklung von Aktiven Sicherheitssystemen können diverse Prozessmodelle und Methoden wie z.B. das V- bzw. W-Modell, Rapid Prototyping (RP), Extreme Programming (XP) oder andere agile Methoden zur Erfüllung der unterschiedlichen Entwicklungsaufgaben zum Einsatz kommen. Diese Methoden bieten auch Unterstützung bei der Qualitätssicherung an, insbesondere zu den jeweiligen Meilensteinen und Quality Gates. Im Kapitel 2 wurde bereits der Testprozess von Spillner et al. vorgestellt und wie Hard- und Software in diesem Bereich zielgerichteter getestet werden kann. Quality Gates können nun diesen Blickwinkel um einen inkrementell-geprägten Blickwinkel erweitern, indem diese Quality Gates ihre eigenen Anforderungen an ein SUT definieren, aus denen sich wiederum weitere Testfälle ableiten können, die zu einem speziellen Zeitpunkt erfüllt sein müssen. Durch die Anwendung von zentralisierten Repositories zur Ablage des entwickelten Codes und der Testfälle kann die Granularität bis auf einzelne Revisionsnummern heruntergebrochen werden, da jede Revision für sich auch als gegenwärtiger Stand des SUTs, gleich ob eine Funktion, ein Teilsystem oder ein Codefragment, angesehen werden. Aufgrund der zeitlichen Abfolge von Revisionsnummern zu Artefakten lässt sich die relative Qualität durch Meta-Metriken feststellen. [BBH+13]. Dabei ist darauf zu achten, dass grundsätzlich bei einem Vergleich von Meta-Metriken zu unterschiedlichen Zeitpunk-

ten lokale Minima und Maxima identifiziert werden, damit eine ganzheitliche Betrachtung der Qualitätsveränderung erfolgen kann.

Die exemplarische Anwendung der Funktions- und Meta-Metriken erfolgt nun im Rahmen der Fallstudie.

8.4 Fallstudie

An dieser Stelle soll nun die Fallstudie fortgesetzt werden. Dabei werden die zuvor genannten Aspekte zum GQM-Paradigma und den Funktions- und Meta-Metriken auf experimentell erzeugte Daten aus einer bereits durchgeführten Studie aufgegriffen. [BBH+15b]

In 2 ist als Ziel der Simulationsumgebung u.a. die qualitative Bewertung von Komponenten eines Fahrerassistenzsystemen im Consumer Test Kontext festgelegt worden. Konkret ist das Ziel, einen Entscheidungsalgorithmus auf sein Verhalten im Systemkontext zu analysieren.

Daraus ließe sich z.B. die Frage ableiten:

> *Wie verhält sich ein Entscheidungsalgorithmus, der u.a. Ausweichtrajektorien als Entscheidungsgrundlage berechnet, innerhalb von Consumer Test Szenarien, deren Parameter gewisse Toleranzbereiche zulässt?*

Mit dieser Frage können zwei Funktionsmetriken herausgestellt werden; zum einen die Funktionsmetrik “Time-To-Collision”, welche den Auslösezeitpunkt des Algorithmus vor der Kollision definiert. Sie dient der Bestimmung des weiteren zeitlichen Verlaufs bis zum Kontakt mit einem Objekt in den Consumer Test Szenarien.

$$ttc_v = -\frac{v_{start}}{a} - \sqrt{\frac{v_{start}^2}{a^2} + 2 * \frac{D_x}{a}} \tag{8.3}$$

Zum anderen lässt sich auf dieser zeitlichen Grundlage dann auch die Restgeschwindigkeit des Fahrzeugs als Funktionsmetrik bei einer entsprechenden Modellierung der Bremsleistung bestimmen.

$$v_{rest} = v_{start} - a * ttc_v \tag{8.4}$$

Nachfolgend sei in der Tabelle 8.2 für die Variation der seitlich zulässigen Abweichung von der Ideallinie aufgeführt.

Test Geschwindigkeit $[km/h]$	linksseitige Fahrspur		ideale Fahrspur		rechtsseitige Fahrspur	
	D_x $[m]$	v_{Rest} $[m/s]$	D_x $[m]$	v_{Rest} $[m/s]$	D_x $[m]$	v_{Rest} $[m/s]$
10	3,97	0	4,08	0	3,97	0
15	5,46	0	5,46	0	5,46	0
20	6,98	0	7,43	0	6,98	0
25	8,72	1,33	9,28	0	8,72	1,33
30	11,27	2,84	11,60	2,39	11,27	2,84
35	14,75	3,42	14,75	3,42	15,14	2,99
40	19,38	3,33	19,38	3,33	19,38	3,33
45	20,81	6,07	21,82	5,46	20,81	6,07
50	23,24	7,71	23,80	7,45	23,24	7,71

Abbildung 8.2: Der Auslöseabstand D_x und die Restgeschwindigkeit v_{rest} für jede Testgeschwindigkeit sowie die ideale, rechts- und linksseitige Fahrspur. (vgl. [BBH+14a]

Die oben genannte Tabelle zeigt, dass es von der Fahrspur abhängig ist, welche Restgeschwindigkeit bei steigender Testgeschwindigkeit am Ende noch vorliegen kann. Sofern nun beispielsweise eine Punktevergabe nach Kategorien wie z.B. $15 - 20\ km/h$ gebildet erfolgt, kann bei gleichem Testfall ein unterschiedliches Ergebnis vorliegen (vgl. hierzu Testfall mit $45\ km/h$). Hier würde die Restgeschwindigkeit beispielsweise einmal unterhalb von $20\ km/h$ liegen und bei seitlicher Ablage des Fahrzeugs darüber, sodass die Anfahrt einen Einfluss auf das Abschneiden des Testfahrzeugs auf das Consumer Test Ergebnis hat.

Aber auch bei Anwendung der Methodik zum Äquivalenzklassentest kann eine solche Einteilung des Resultats in Kategorien mit dem entsprechenden Effekt auftreten und möglicherweise zu abweichenden Ergebnissen führen. Um dies sichtbar zu machen, könnte eine neue Funktionsmetrik entwickelt werden, die erfasst, an welcher Stelle solche Kategorie-Sprünge auftreten. Diese ließe sich wie folgt definieren:

$$R_{Sprung}(v_{left}, v_{limit}, v_{ideal}) = \begin{cases} 1 & \text{wenn } v_{left} > v_{limit} > v_{ideal}\,, \\ 0 & \text{sonst.} \end{cases} \tag{8.5}$$

Analog gilt für Testfälle mit rechtsseitiger Fahrspur:

$$R_{Sprung}(v_{ideal}, v_{limit}, v_{right}) = \begin{cases} 1 & \text{wenn } v_{ideal} < v_{limit} < v_{right}\,, \\ 0 & \text{sonst.} \end{cases} \tag{8.6}$$

Test Geschwindigkeit $[km/h]$	linksseitige Fahrspur		ideale Fahrspur		rechtsseitige Fahrspur	
	v_{Rest} $[km/h]$	ctk_{links} $[\#]$	v_{Rest} $[km/h]$	$-$ $[-]$	v_{Rest} $[km/h]$	ctk_{rechts} $[\#]$
10	0	0	0	-	0	0
15	0	0	0	-	0	0
20	0	0	0	-	0	0
25	4,79	0	0	-	4,79	0
30	10,22	1	8,60	-	10,22	1
35	12,31	0	12,31	-	15,14	1
40	11,39	0	11,39	-	11,39	0
45	21,85	1	19,66	-	21,85	1
50	27,75	0	26,82	-	27,75	0

Abbildung 8.3: Dargestellt ist die Restgeschwindigkeit v_{rest} und mit ctk_{links} bzw. ctk_{rechts} die Indikation, ob bei abweichender Trajektorie ein Sprung zwischen zwei Kategorien erfolgen würde.

Hieraus ließe sich dann weitere Optimierungspotentiale durch weitere Entwicklungstätigkeiten identifizieren.

Darüber hinaus ist es nun möglich, durch Bildung einer Meta-Metrik, also durch eine zeitliche, übergeordnete Erfassung der Entwicklung, Trends oder Anomalien festzustellen und so durch geeignete Maßnahmen gegenzusteuern bzw. die Qualität zu verbessern.

Das Prinzip der Meta-Metriken funktioniert, wie bereits in [BBH+13] veranschaulicht wurde. Hierzu wurde aus einem Simulationsframework die Daten aus einem Entwicklungsprojekt verwendet. Dabei wurden 1.867 einzelne Revisionen für alle zu dem Projekt zugehörigen Kompo-

nenten entwickelt [Ber10]. Anhand der Komponente zur Fahrzeugkontrolle und der Revisionshistorie wurde eine exemplarische Anwendung verdeutlicht.

Um nun also aus der Funktionsmetrik R_{Sprung} eine Meta-Metrik zu generieren, wird zunächst die relative Verteilung über das Szenario mit den verschiedenen Geschwindigkeiten gebildet, so dass die Verteilung der Kategoriesprünge bekannt ist.

$$\begin{aligned} R^{+}(R_{\mathit{Sprung}}, N_{tc}) &= \frac{R_{\mathit{Sprung}}(v_{\mathit{left}}, v_{\mathit{ideal}}, v_{\mathit{limit}})}{N_{tc}}, \\ R^{-}(R_{\mathit{Sprung}}, N_{tc}) &= 1 - R^{+}(R_{\mathit{Sprung}}, N_{tc}), \\ Q_4(R_{\mathit{Sprung}}, N_1, N_2) &= R^{+}(R_{\mathit{Sprung}}, N_2) - R^{+}(R_{\mathit{Sprung}}, N_1), \\ &\quad \text{where } N_1 \leq N_2. \end{aligned} \tag{8.7}$$

Daraus ergibt sich mit $Q_4 > 0$ eine mögliche Qualitätsverschlechterung, da nun vermehrt Sprünge über Kategorien angezeigt werden. Analog bildet mit $Q_1 < 0$ die Meta-Metrik eine Verbesserung der Qualität ab. Eine weitere exemplarische Darstellung wird hier nicht vorgenommen, da sich aus Gründen der Einfachheit eine Errechnung ergibt.

8.5 Zusammenfassung

In diesem Kapitel wurde zunächst aufgezeigt, dass (Meta-)Metriken als Interpretations- und Bewertungshilfen gebraucht werden, um die Ausgangsfragen aus der Analyse beantworten zu können. Hierfür wurde zunächst nach einer entsprechenden Definition von beiden Instrumenten gesucht. Diese konnten jedoch nur teilweise ausfindig gemacht und herangezogen werden, so dass eine eigene Definition sowohl für Funktionsmetriken als auch für Meta-Metriken herausgearbeitet wurde.

Zudem wurde mit Hilfe der GQM-(Goal-Question-Metric)-Methode von Basili et al. ein etabliertes Verfahren in diese vorgestellte Methodik integriert, um anhand von Ausgangsfragen geeignete Funktionsmetriken und anschließend Meta-Metriken entwickeln zu können. Dies wurde dann anhand von simulierten Daten aus einem Experiment veranschaulicht.

Im folgenden Kapitel wird vorgestellt, wie geeignete Experimente mit der entwickelten Simulation durchgeführt wird und wie anhand der Ergebnisse eine Antwort auf die Ausgangsfragen gefunden werden kann.

9 Die Durchführung von Simulationsläufen und ihre Auswertung

Nachdem die Messergebnisse innerhalb einer Simulationsumgebung mit einem im Zentrum befindlichen SUT erzeugt und mit Hilfe von Funktions- und Meta-Metriken quantifiziert wurden, stellt sich nun die Frage, welcher Erkenntnisgewinn daraus abgeleitet wird. Dies ist unmittelbar auch mit der Beantwortung der Forschungsfragen bzw. der Zielerreichung im GQM-Modell verknüpft.

Neben der Ergebnisaufbereitung und -darstellung muss auch die Adressierung der Ergebnisse zu den Ausgangszielen und -fragen erfolgen. Gleichzeitig sollte aber auch eine kritische Prüfung der eigenen Ergebnisse in Betracht gezogen werden.

Wohlin, Runeson et Höst stellen in ihrer Methodik ein Vorgehen für die Durchführung von Experimenten und empirischen Studien vor, auf die an dieser Stelle ebenfalls zurückgegriffen wird. Es erfolgt dann eine Adaption für den hier vorliegenden Kontext zur Entwicklung von Aktiven Sicherheitssystemen. [RH08, WRH+12]

9.1 Zum Entwurf einer Untersuchung

Wie schon Anfangs in Kapitel 5 einleitend aufgezeigt wurde, sind bei einer Untersuchung zwei unterschiedliche Paradigmen denkbar:

1. Untersuchungen mit explorativem Charakter
2. Untersuchungen mit erklärendem Charakter

Hier stellt sich nun die Frage, wann wird welches Paradigma im Zuge von Simulation und insbesondere im vorliegenden Kontext verwendet? An dieser Stelle soll eine Orientierung geboten werden, anhand welcher kontextueller Merkmale im Projektrahmen eine Zuordnung erfolgen kann.

Wie bereits in Kapitel 5 eingangs erwähnt, werden innerhalb einer explorativen Untersuchung die zu betrachtenden Objekte in ihrer ursprünglichen Umgebung auf mögliche Phänomene geprüft. [RH08, WRH+12] In einer ersten Annäherung wird davon ausgegangen, dass ein solches Objekt in einer mehr oder weniger realen Umgebung eingebettet ist und die notwendigen Daten durch die tatsächlichen Datenerzeuger bereitgestellt werden. Dadurch bleiben die Daten während ihrer Erzeugung vergleichsweise unbeeinflusst. Daraus wird gefolgert, dass aus dieser Eingrenzung eine Simulation bzw. eine simulative Erzeugung von Daten einer explorativen Untersuchung nicht dienlich sein kann, da sonst mögliche Phänomene durch die Verfälschung nicht richtig erfasst werden könnten.

Hingegen liegt der Fokus bei Untersuchungen mit erklärendem Charakter nach Runeson et al. vor allem in der Kontrollierbarkeit der Randbedingungen bzw. die Sicherstellung von gleichen Bedingungen und Aufbau. Durch wenige, gezielte Veränderungen kann die eigentliche Absicht erreicht werden: Ursache-Wirkung-Beziehungen zwischen den Objekten zu identifizieren. [WRH+12] Daher eigenen sich solche kontrollierten Experimente für einen simulativen Kontext, da hier die jeweiligen Objekte in einer eigenen geschaffenen Umgebung auf ihre Wirkungen bzw. jenen Ursachen hin untersucht werden können. Das setzt jedoch voraus, dass das Verhalten der Objekte soweit beschreibbar bzw. abstrahierbar ist und sich ein Modell für eine Simulationsumgebung entwickeln lässt.

Um nun eine empirische Untersuchung durchzuführen, ist es entscheidend, zunächst ihre Art festzulegen. Zur Auswahl stehen nach Runeson et al.:

- Umfragen
- Fallstudien
- (Quasi)-Experimente.

Da sich im Kontext dieser Arbeit der Schwerpunkt auf die technischen Aspekte von Sicherheitsfunktionen und ihrer Entwicklung bezieht, sind Umfragen im Sinne einer Sammlung von Informationen zu oder über Personen und deren Wissen, Verhalten oder Einstellungen nicht geeignet. Fallstudien und Experimente sind im vorliegenden Kontext der Arbeit wesentlich besser einzusetzen, da sie einerseits Phänomen-Untersuchungen im Rahmen von Fallstudien erlauben, wenn also der Kontext nicht eindeutig definiert ist oder andererseits im Rahmen von Experimenten Parametervariationen mit überwiegend gleichbleibenden Randbedingungen gearbeitet wird. Als einen Spezialfall wird das Quasi-Experiment ausgewiesen, welches die wesentlichen Eigenschaften eines Experimentes aufweist, jedoch die Zufallskomponenten während der Untersuchung eliminiert. [RH08, WRH+12, S. 58 und S. 146]

Aufgrund dieser Einordnung von Untersuchungsarten lässt sich relativ schnell ableiten, dass für den vorliegenden Kontext die (Quasi-)Experimente die bevorzugte Wahl sind, um aus den bisherigen Arbeitsschritten die Aussagen abzuleiten, die zu den Antworten der Ausgangsfragen führen. Zudem ist zu prüfen, ob die Zufallskomponente im Aufbau des Experiments eliminiert werden soll.

Die Ausarbeitung des Experiments greift auf die bisherigen Entwicklungen innerhalb der einzelnen Methodik-Bausteine zurück. Um aus den Informationen Wissen zu generieren, müssen die Randbedingungen bekannt sein. Durch die Beschreibung der Simulationsläufe als Experiment können die Randbedingungen einerseits definiert und andererseits eventuelle Kausalitäten bzw. Abweichungen vom erwarteten Verhalten identifiziert werden.

9.2 Aufbau eines Experiments - Details zu Untersuchungsraum und -tiefe

Für den Aufbau eines Experiments werden die fünf Hauptaspekte von Runeson et al. aufgegriffen und insbesondere mit den bisherigen Arbeitsergebnissen verknüpft, um die Untersuchung und die damit verbundene Datenerzeugung auf die bisherigen Ergebnisse aus den ersten Prozessschritten zur Methodik abzustimmen. An einigen Stellen kommt es sicher zu Überschneidungen. Da aber gerade dieser Vorbereitungsschritt wichtig für die Duchführung ist, sollte er in jedem Falle auch noch einmal vorgenommen werden. Die Standardisierung des Experiment-Aufbaus und die Vergegenwärtigung der vorliegenden Situation unterstützt die Zielsetzung des Experiments. Auch im Hinblick auf spätere Rückkopplungen aus den Experiment-Ergebnissen können so die Randbedingungen besser dokumentiert und bei erneuter Durchführung mit evtl. Anpassungen am Aufbau ausgeführt werden. Weiterhin ist denkbar, dass nur eine Schwerpunktverschiebung hinsichtlich eines anderen Untersuchungsgegenstandes geben kann. Der grundsätzliche Aufbau bleibt jedoch gleich. Mehr dazu wird im nachfolgenden Abschnitt erläutert.

9.2.1 Gegenstand der Untersuchung

Gegenstand der Untersuchung ist üblicherweise das Objekt, über das eine Aussage getroffen werden soll. In vorliegenden Fall wird es eine Komponente einer Sicherheitsfunktion oder mehrere Funktionen sein. Weiterhin können aber auch andere Systemkomponenten in den Fokus rücken, so dass einzelne Komponenten innerhalb der Simulation ausgetauscht werden. So bleibt

der grundsätzliche Aufbau im Vergleich fast unverändert, jedoch wird der Schwerpunkt durch den Austausch des Experiments verschoben.

9.2.2 Zweck der Untersuchung

Hier knüpft die Definition des Zwecks an der Deklaration des Gegenstandes an: Der Zweck der Untersuchung kann sich trotz des gleichbleibenden Aufbaus verändern. Auch gibt es mögliche Veränderungen, wenn die Veränderung der einzelnen Komponenten bzw. der Artefakte, die untersucht werden sollen, ausgetauscht werden. Weiterhin kann aber auch eine Veränderung im Aufbau den Zweck der Untersuchung geringfügig beeinflussen. Dies ist z.B. der Fall, wenn ein Sensor innerhalb eines Systems ausgetauscht wird, jedoch der Gegenstand der Untersuchung der Entscheidungsalgorithmus bleibt.

9.2.3 Qualitativer Schwerpunkt

Bei der Ausrichtung des qualitativen Schwerpunkts der Untersuchung geht es um den vorrangigen Effekt, der während des Experiments untersucht werden soll. Dies kann z.B. die Effektivität oder die Zuverlässigkeit einer Sicherheitsfunktion sein, so dass nicht nur eine Sicherheitsfunktion die ihr bestimmte Aufgabe erfüllt, sondern dies auch dauerhaft im Falle der Wiederholung leistet.

9.2.4 Perspektive

Die eingenommene Perspektive hängt hier letztlich von den Stakeholdern des Untersuchungsgegenstandes ab. Da es pro Untersuchungsgegenstand mehrere Stakeholder geben kann, ist auch hier eine weitere Definition erforderlich. So kann z.B. eine Zuverlässigkeitsuntersuchung zu einem Algorithmus sowohl aus der Perspektive des Entwicklers geprüft werden, als auch aus Sicht des Kunden. Auch innerhalb eines Stakeholders können unterschiedliche Sichtweisen eingenommen werden. Die Interessenlage der jeweiligen Person ist dabei entscheidend und sorgt somit für eine Konkretisierung der qualitativen Ausrichtung.

9.2.5 Kontext

Der Kontext beschreibt die jeweiligen Randbedingungen des Kontextes. Dies ist bereits im Vorfeld der Simulationsentwicklung erfolgt, jedoch nicht vor dem konkreten Hintergrund des je-

weiligen Experiments. Somit ist hier eine Vergegenwärtigung bzw. weitere Konkretisierung von Vorteil. Auch können im Falle noch fehlender, notwendiger Randbedingungen Eingriffe auf die Simulationsumgebung identifiziert und vorgenommen werden. Beispielsweise wären bei zwei Untersuchungen, die jeweils als Kontext die zulässigen Abweichungen aus dem EuroNCAP bzw. dem USNCAP zum Gegenstand haben, fast identisch in ihrem Aufbau. Eine Unterscheidung ergibt sich erst aus den unterschiedlich zulässigen Toleranzen für das untersuchte Fahrzeug.

Insgesamt lässt sich festhalten, dass durch die Definition dieser fünf Aspekte zum Untersuchungsraum und -tiefe eine weitere Detaillierung dessen vorgenommen wird, was bereits im Vorfeld während der Simulationsentwicklung begonnen wurde. Es ist zudem keine Wiederholung der bereits getroffenen Annahmen, Angaben und Festlegungen aus den ersten Prozessschritten dieser Methodik. Vielmehr dienen sie als Ausgangslage für die Definition der Experimente. Gleichzeitig wird die Möglichkeit gegeben, evtl. bis dahin noch nicht erkannte, wichtige und notwendige Bedingungen in die Entwicklung der Simulationsumgebung bzw. in die (Meta-)Metriken zu integrieren. Außerdem wird nachvollziehbarer, wie die Datenerzeugung abgelaufen ist.

9.3 Planung eines Experiments

In Kapitel 5 wurde bereits die Methodik von Runeson et al. in ihren Ansätzen vorgestellt. Für einen vertiefenden Blick sei auf jenes Werk verwiesen. An dieser Stelle soll jedoch eine Einordnung des vorliegenden simulativen Umfeldes in die Methodik erfolgen.

9.3.1 Kontextauswahl

Bei der Kontextauswahl ist im weiteren Sinne die Reichweite bzw. auch das (personelle) Umfeld des Experiments im Fokus. In der Regel werden die Personen, welche die Simulationsläufe durchführen, aus dem Entwicklungsbereich kommen, wenn die Softwarestände des SUT noch aus einer frühen Entwicklungsphase stammen und evtl. Adaptionen an der Simulationsumgebung vorgenommen werden müssen.

9.3.2 Variablenauswahl

Bei der Variablenauswahl kann zwischen abhängigen und unabhängigen Variablen unterschieden werden. [WRH+12, RH08] Die unabhängigen Variablen sind diejenigen, die in der Unter-

suchung beeinflusst werden können, während die abhängigen Variablen nur einer bedingten bis keiner Einflussmöglichkeit unterliegen. Letztere sind üblicherweise Bestandteil im Rahmen der Hypothesenbildung und müssen daher sehr sorgfältig ausgewählt und überwacht werden, da sie damit das Endergebnis maßgeblich bestimmt. Im hier vorliegenden Kontext wären beispielsweise die zustandsbeschreibenden Parameter der Objekte wie Geschwindigkeit, Position, etc. von Fahrzeugen zu den unabhängigen Variablen und eine Variable Auslöseabstand zum Objekt zu den abhängigen Variablen zuzuordnen.

9.3.3 Subjektauswahl

Die Auswahl des zu untersuchenden Objektes ist in der Regel immer auf das zu entwickelnde Objekt im Rahmen eines Entwicklungsprojektes bezogen. Das kann zum einen die Sicherheitsfunktion selbst, zum anderen aber auch angrenzende Systemkomponenten sein. Dies schließt sowohl eine spezielle Funktion innerhalb der Komponente Teil der Untersuchung als auch alle realisierten Funktionen in ihrer Gesamtheit ein. [WRH+12]

9.3.4 Wahl der Design Typen

Hierunter wird verstanden, dass das Experiment im Aufbau gewissen Eigenschaften unterliegt. Dazu zählen z.B. Zufälligkeiten, Gruppierung oder auch Ausgleich. [WRH+12] Bei der Berücksichtigung des Zufalls werden die unabhängigen Variablen eines Experiments in ihren Werten zufällig definiert, wodurch unerwünschte Faktoren vernachlässigt werden können, wie beispielsweise eine Ungenauigkeit bei der Positionsbestimmung von Sensoren zur Umfelderfassung. Entsprechende Methodiken können z.B. die Monte-Carlo-Simulation sein. Ein weiteres Konzept sieht die Blockbildung vor, so dass ein Effekt, der sich auf einige Objekte gleichermaßen auswirkt, dadurch eliminiert wird, dass alle betreffenden Objekte zu einer Gruppe zusammengefügt werden. Damit gelten sie für alle gruppierten Objekte und können somit in der Untersuchung vernachlässigt werden.

Neben diesen generellen Aspekten zum Entwurf gibt es mehrere Standard Typen, die jeweils von der Anzahl der unabhängigen Variablen (Faktoren) und der Anzahl der Ausprägungen abhängen. Runeson et al. unterscheiden neben anderen vier gängige Typen:

1. einen Faktor mit zwei Ausprägungen
2. einen Faktor mit mehr als zwei Ausprägungen
3. zwei Faktoren mit zwei Ausprägungen
4. mehr als zwei Faktoren mit zwei Ausprägungen [WRH+12, RH08]

Die Wahl der Faktoren und deren Ausprägungen hängt auch hier wieder vom Kontext ab. Ein Beispiel sei an dieser Stelle jedoch angeführt, auf das sich später auch im Rahmen der Fallstudie bezogen wird: Nr. 2 mit einem Faktor und mehr als zwei Ausprägungen. Im Rahmen der Toleranzanalyse und ihre Auswirkungen auf ein SUT in den Consumer-Test-Senzarien wird als ein Faktor bzw. eine unabhängige Variable variiert und ihren Einfluss auf die abhängige Variable TTC untersucht. Dabei kann die Veränderung wie z.B. die Geschwindigkeit des Fahrzeugs in mehr als zwei Ausprägungen verändert werden.

9.3.5 Auswahl von Instrumentation

Hierbei handelt es sich um die Begleitgabe von Hilfsmitteln, wie z.B. der konkrete Code einer Softwarekomponente oder auch anderen Artefakten, die bei der späteren Durchführung unterstützen können. Das Ziel muss dabei sein, dass diese Hilfsmittel das Ergebnis der Untersuchung nicht beeinflussen. Andernfalls würde eine Verfälschung der Ergebnisse genau um dieses Hilfsmittel erfolgen und die spätere Schlussfolgerung nicht auf einen eindeutigen Effekt zurückzuführen sein. Andere Hilfsmittel könnten aber auch Anleitungen zur Durchführung der Experimente oder bereits existierende Messergebnisse sein.

9.3.6 Zur Validität des Experiments

Runeson et al. beziehen sich im Rahmen der Validität des Experiments und seiner späteren Ergebnisse auf die Ausführung von Cook and Campell. ([CS63, CC79] in [WRH+12]) Dabei werden mehrere Kategorien von Validitätsbetrachtungen mit verschiedenen Reichweiten unterschieden:

- interne Validitätsbetrachtung
- externe Validitätsbetrachtung
- konstruktive Validitätsbetrachtung
- Validitätsbetrachtung bzgl. der Schlussfolgerungen.

Bei der internen Validitätsbetrachtung wird die Wirkfolge von Anfangsbedingungen und dem Ergebnis in den Fokus genommen, so dass eine Aussage darüber getroffen wird, ob evtl. nicht berücksichtigte Faktoren das Ergebnis doch beeinflussen würden. Die externe Validitätsbetrachtung prüft die Übertragbarkeit der Ergebnisse und deren ursächlicher Effekt auf den Kontext außerhalb der Untersuchung, damit die Grundlage für eine Generalisierung gegeben ist. Im Rahmen der konstruktiven Validitätsbetrachtung geht es um den Zusammenhang von Theorie und Beobachtung und deren jeweiligen Ursache-Wirkungszusammenhänge. Das bedeutet, dass die Untersuchung im Aufbau die Ursache widerspiegeln muss und der Aufbau für den jeweiligen Effekt ebenfalls im Ergebnis auffindbar sein muss. [WRH+12, S. 103]

Die Validitätsbetrachtung bzgl. der Schlussfolgerung wird bei Runeson als Prüfung der Validität insbesondere bei statistischen Tests erläutert. Dies ist dann der Fall, wenn insbesondere empirische Erhebungen wie z.B. bei Umfragen, etc. vorgenommen werden. Die dort vorhandene Zufallskomponente muss entsprechend bei der Schlussfolgerung berücksichtigt werden. Da jedoch in diesem der Arbeit vorliegenden Umfeld Funktions- und Softwaretest ohne eine Zufallsgröße einen Schwerpunkt bilden und u.a. auch mit “bestanden” oder “nicht bestanden” als Ergebnis gearbeitet wird, ist eine klassische Betrachtung im Sinne Runesons et al. mit rein statistischen Schlussfolgerungen nicht zielführend. Vielmehr muss eine Einzelfallbetrachtung vorgenommen werden, wenn es um die Behandlung dieser Art von Validitätsbetrachtung geht.

Im Rahmen der Planung muss bereits eine vorgreifende Prüfung auf die Validitätsaspekte gelegt werden, damit die dafür erforderlichen Rahmenbedingungen geschaffen werden können. Andernfalls könnte erst nach Durchführung der Studie auf mögliche Validitätsverletzungen oder strittige Punkte reagiert werden. Gleichwohl ist es jedoch gerade im Falle von bisher noch nicht behandelten Fragestellungen möglich, dass Validitätsaspekte im Vorfeld nicht identifiziert werden konnten. Diese sind im Nachgang der Untersuchung herauszuarbeiten und entsprechend zu argumentieren. Dabei ist es grundsätzlich denkbar, die Validitätsbetrachtung erst im Rahmen der Bewertung der Ergebnisse zusammenfassend aufzuführen.

9.4 Durchführung des Experiments

Für die Durchführung werden eine Reihe von weiteren Aspekten angeführt, die sich zunächst auf Studien mit Teilnehmern beziehen. Dazu gehören zum Beispiel Ort der Durchführung, Motivation der Teilnehmer, Messinstrumente, etc. Diese Aspekte sind im vorliegenden Kontext zur Simulation in dieser Ausprägung so nur von geringer Relevanz. Vielmehr spielen andere Aspekte dabei eine Rolle, wie z.B. die Vorbereitung des Simulationssetups oder auch die Überwachung

der einzelnen Simulationsläufe.

Im Zusammenhang mit der unmittelbaren Durchführung ist es wichtig, dass insbesondere bei einer Verwendung von verteilten Systemen auf eine korrekte Konfiguration der Einzelsysteme geachtet wird. Verteilte Systeme kommen speziell dann zum Einsatz, wenn mehrere Simulationskomponenten, Restbussimulationen oder auch spezielle bzw. komplexe Modelle benötigt werden, die ihr eigenes Framework benötigen und über diverse Schnittstellen je nach Zielsetzung für die Simulation zu einem Gesamtsystem zusammengeführt werden.

Eine weiterer Punkt wäre die Schaffung von parallelen Berechnungen durch die Verwendung von weiteren Instanzen der Simulationsumgebung, so dass mehrere Simulationsläufe entweder mit oder ohne veränderte Randbedingungen durchgeführt werden. Ersteres wird angewendet, um die Dauer der Durchführung zu verkürzen. Dies ist dann durch den Aufwand an Ressourcen begrenzt, da nicht jeder Ressourceneintrag auch wirtschaftlich vertretbar ist. Hier gilt es ein entsprechendes Maß zwischen Aufwand und Zeitverkürzung zu finden. Unveränderte Randbedingungen werden dann in mehreren Instanzen parallel simuliert, wenn entweder das gleiche SUT dahingehend überprüft wird, ob evtl. Einflüsse aus der technischen Infrastruktur heraus das Ergebnis beeinflussen oder wenn mehrere SUT untersucht werden sollen. Hier ist grundsätzlich eine sequenzielle Durchführung der Simulationsläufe möglich.

Weiterhin sollte bei der Durchführung intern auf das Zeitverhalten und die Einhaltung der Abläufe geachtet werden, damit am Ende bei Latenzen nicht falsche Schlussfolgerungen gegenüber dem SUT vorgenommen werden.

9.5 Analyse und Diskussion der Ergebnisse

Die bereits in der Planung berücksichtigten Validitätsbetrachtungen werden am Ende der Durchführung erneut abgeglichen und ggf. an der einen oder anderen Stellen ergänzt, je nach Auftreten der im Vorfeld gemachten Annahmen oder der neu hinzugekommenen Aspekte.

9.5.1 Interne Validitätsbetrachtungen

Hinsichtlich von Simulationsumgebungen geht es bei den internen Validitätsbetrachtungen darum, dass der Einfluss nicht wahrgenommener oder bedachter Faktoren Einzug in das Ergebnis erhält. So könnte z.B. innerhalb eines Sensormodells, welches in einem eigenen Framework in die Simulationsumgebung eingebunden ist, eine Zufallskomponente, die z.B. einen Messfehler

nachbilden soll, das Ergebnis in die ein oder andere Richtung beeinflussen. Denn wenn nicht bekannt ist, um welchen Wert es sich genau handelt, ist ein Faktor in der Ursache-Wirkung-Kette enthalten, der am Ende die Argumentation erschwert eine hinreichende Schlussfolgerung zu ziehen. Speziell bei solchen zufallsbasierten Variablen muss dann bekannt sein, um welchen Zufallswert es sich konkret handelt und wie er in einem Zusammenhang zu anderen Zufallswerten und anderen gleichen Berechnungen innerhalb von Methoden steht.

Des Weiteren gibt auch die Problemstellung den Rahmen vor, inwiefern eine Untersuchung und ihre Konstruktion valide ist. So ist beispielsweise zu prüfen, ob es sich tatsächlich um eine zu lösende Problemstellung handelt oder ob das Problem vielmehr nur konstruiert ist und keinen reellen Bezug aufweist. Es handelt sich um eine valide Problemstellung, wenn die definierenden Rahmenbedingungen durch Dritte festgelegt und durch gesetzesähnliche Regelungen formalisiert wurden.

9.5.2 Externe Validitätsbetrachtungen

Für die externen Validitätsbetrachtungen wird die Übertragbarkeit der Untersuchung auf andere Konstellationen von Rahmenbedingungen geprüft, welche zuvor die Grenzen der Untersuchung definiert haben. Gleichzeitig wird somit aufgezeigt, welche Rückschlüsse nicht möglich sind. Diese Nennung von Pro und Contra Aspekten zur Übertragbarkeit auf einen anderen Kontext außerhalb der Untersuchung bildet durchaus den Kern der Ergebnisse. Konkret für Simulationsumgebungen können das z.B. Ansätze sein, die auch mit anderen Systemkomponenten durchführbar sind. Ein weiter Punkt ist die Übertragbarkeit vom simulativen Kontext auf den späteren Produktivkontext. Wesentliche Herausforderung dabei ist es, aus der Abstraktion während der Modellbildung nun den Bogen wieder zurück auf den ursprünglichen Zustand zurück zu spannen und die richtige Argumentation, untermauert durch die Ergebnisse, anzuführen. Dabei ist insbesondere im Auge zu behalten, dass das Modell keine 1:1 Abbildung der Realität darstellt und dies nicht erstrebenswert ist. Vielmehr profitiert die Simulation aus der Abstraktion und damit aus der Komplexitätsreduktion, welche angemessen gewürdigt werden muss. Hier gilt: Der Kontext ist für jedes Simulationsprojekt so individuell, dass eine Einzelfallbetrachtung pro Projekt erforderlich ist.

9.5.3 Konstruktive Validitätsbetrachtungen

Bei den konstruktiven Validitätsbetrachtungen kommt es im Rahmen von Simulationsexperimenten in Abstimmung mit dem vorliegenden Kontext darauf an, dass man einerseits die Ursa-

che valide im Aufbau der Untersuchung darstellt, andererseits der Effekt im Ergebnis hinterher beobachtbar sein muss. Nach Runeson leitet sich das aus der Theorie zunächst ab ([WRH+12]), so dass aus den Rahmenbedingungen hervorgeht, was die Ursache ist und wie diese sich dann im Aufbau widerspiegeln. In der Theorie wird auch ein entsprechender Effekt annehmbar sein, der sich dann in den Ergebnissen darstellen muss. Somit müssen die Modelle die wesentlichen Wirkzusammenhänge darstellen und gleichzeitig alle aus dem Systemkontext relevanten Variablen aufnehmen können. Als Beispiel sei das EuroNCAP Test-Protokoll erwähnt, welches jeweils einen idealen Testfall beschreibt, hingegen aber gewisse Abweichungen von der Ideallinie zulässt. Das hat Auswirkungen auf das Verhalten des untersuchten Algorithmus, so dass der Effekt in einem Simulationsergebnis vorhanden sein wird.

9.6 Rückkopplung und Zuordnung der Ausgangsfragen

Im finalen Schritt sind dann die jeweiligen Forschungsfragen unter Berücksichtigung der Ergebnisse zu beantworten und in einer anschließenden Aussage zusammenzufassen. Dies ist sicherlich eine vermeintlich selbstverständliche Annahme, ist jedoch für den Abschluss der Untersuchung eine wichtige Aufgabe und kann unter Umständen während der Ergebniszusammenstellung nicht die nötige Berücksichtigung finden.

9.7 Fallstudie

Die Vorstellung der Methodik wird durch die beispielhafte Erläuterung einer Studie in Consumer-Test-Umfeld zum Ende geführt. Dazu werden aus der exemplarischen Darstellung eines Experiments die einzelnen Arbeitsschritte näher erläutert.

9.7.1 Art der Untersuchung und Forschungsfragen

Die Auswahl der Untersuchung basiert an dieser Stelle auf einem Quasi-Experiment, der folgende Forschungsfragen zugrunde liegen:

- *RQ1: Wie verhält sich ein Funktionsalgorithmus eines Aktiven Sicherheitssystems in den ursprünglichen Consumer Test Szenarien CCRs, CCRm und CCRb?*
- *RQ2: Zu welchen Parameterkonstellationen tritt ein untypisches Verhalten auf?*

Die erste Frage prüft, wie sich das Auslöseverhalten des Algorithmus bei systematisch variierten Randbedingungen innerhalb der Toleranzen der Consumer Test Szenarien verhalten wird. Konkret sind dabei zwei Fragen im Fokus:

1. *Wie verändert sich das Auslöseverhalten der Bremsfunktion bei Veränderung von Geschwindigkeit und seitlicher Versetzung zur Ideallinie?*
2. *Was bedeutet dies für die Restgeschwindigkeit des Fahrzeugs?*

Die zweite Frage zielt darauf ab, mögliche Anomalien im Funktionsverhalten zu identifizieren und so Handlungspotential aufzuzeigen. Dadurch wird es möglich, gezielt reale Funktionsuntersuchungen z.B. auf Prüfgeländen durchzuführen. Als Anomalie ist hier ein Funktionsverhalten gemeint, dass sich von einem zu erwartenden Verhalten unterscheidet. Da dieses Verhalten nicht unbedingt als Fehler bzw. Fehlerwirkung zu verstehen ist, wird eine neutrale Form der Benennung gewählt.

9.7.2 Aufbau des Experiments

Gegenstand der Untersuchung ist ein Algorithmus zur Auslösung einer Notbremsung. Dieser liegt als Blackbox vor, so dass über interne Zustände keine Informationen zur Verfügung stehen. Die Untersuchung soll aufzeigen, an welcher Stelle möglicherweise noch Optimierungspotential in seinem Verhalten vorliegt und wo weitere Tests sinnvoll wären. Zweck der Untersuchung ist es, das Verhalten im Rahmen einer Parametervariation näher zu untersuchen, da sich ein reiner Äquivalenzklassentest hier als nicht ausreichend erweist, so dass der Fokus der Untersuchung einer Verbesserung der Zuverlässigkeit zu erzielen. Dabei wird die Perspektive eines Entwicklers eingenommen, der speziell Anforderungen an das System projekttechnisch verfolgt, die für ein entsprechendes Testergebnis bei einem Consumer Test wie z.B. EuroNCAP definiert worden sind. Die Untersuchung erfolgt im Kontext der Consumer Test Definition von EuroNCAP in seiner Fassung von 2015. Diese Auswahl erfolgt als Abstraktionsmaßnahme gegenüber einer heutigen Entwicklung.

9.7.3 Planung des Experiments

Die Untersuchung wird im Umfeld der Simulationsumgebung, wie in Kapitel 7 beschrieben, verwendet. Dabei werden zwei Variablen verwendet:

- Geschwindigkeit des Vehicle under Test (VUT)
- seitliche Ablage des VUT

Die Ausprägungen orientieren sich an dem im Protokoll zur Durchführung von EuroNCAP Consumer Tests jeweils angegebenen Werten im jeweiligen Szenario (vgl. Kapitel 3).

Um den Effekt von Seiteneinflüssen durch Sensormodelle oder fahrdynamische Modelle zu reduzieren, wurden zunächst simple Modelle in die Simulationsumgebung implementiert, so dass eine spätere Rückkopplung und Überprüfung von Ergebnissen auf Einflüsse aus den Modellen nicht umfassend erfolgen kann, weil eine Beeinflussung nur bedingt möglich ist.

Da es sich bei der Fragestellung um eine qualitative Bewertung des Artefakts im Software-Entwicklungsprozess unter Laborbedingungen handelt und die Rahmenbedingungen aus einem fremdentwickelten Testprotokoll stammen, ist eine Übertragbarkeit auf ähnliche Kontexte gegeben. Durch die ausgedehnte Parametervariation wird eine Verhaltenslandkarte des Artefakts erstellt, so dass a priori bereits ein umfassendes Wissen über das Funktionsverhalten existiert.

Dies gilt auch insbesondere für die Konstruktionsbedingungen des Experiments, da durch die Einbindung des Algorithmus mit seinen Eingangs- und Ausgangsparameter alle wesentlichen Einflüsse vorgenommen werden und die Toleranzbereiche aus dem Consumer Test Protokoll festgeschrieben sind.

9.7.4 Durchführung des Experiments

Zur Durchführung des Experiments werden eine Vielzahl von Pfaden mit dem Parameter-Tool aus Kapitel 7 erzeugt, wobei diese sowohl für das VUT als auch das Zielfahrzeug (EVT) genutzt werden. Jeder Knoten repräsentiert gewisse Eigenschaften der simulierten Fahrzeuge wie z.B. Position, Ausrichtung, Geschwindigkeit, etc.. Jeder Knoten wird zu jedem Tick, den die Zeitkontrolle alle $40\ ms$ aussendet, von der Simulation verarbeitet. Alle Signale und Funktionsmetriken werden zur weiteren Verarbeitung im comma-separated-value Format gespeichert.

Das in Abbildung 9.1 dargestellte Modell visualisiert den Ablauf des Experiments: Während der ersten Phase wird das Fahrzeug solange positioniert bis der Algorithmus den Zustand der Notbremsung auslöst. Danach erfolgt die Berechnung der Restgeschwindigkeit und des daraus resultierenden EuroNCAP Ergebnisses. In der Phase 1 wird für einen Simulationslauf das VTD Szenario, der virtuelle Fahrer und die Fahrzeugdynamik anhand der Informationen aus der Szenariobeschreibung und dem Pfad p vorbereitet. Via SCP Befehl wird die Simulation dann

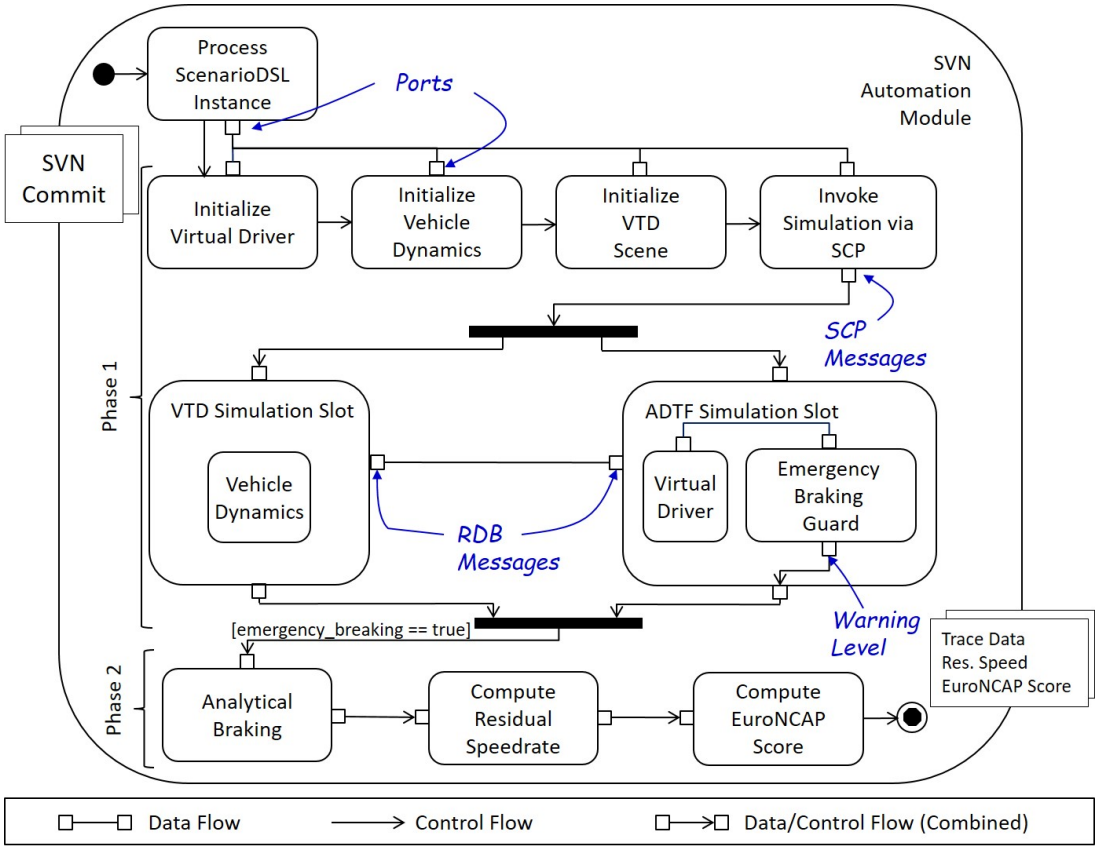


Abbildung 9.1: Phasen Modell zur Durchführung des Experiments

bis zur Auslösung des Algorithmus gestartet. Anschließend erfolgt die Berechnung der Restgeschwindigkeit durch die Bestimmung des Abstandes zum Zielfahrzeug. In [BBH+15b] wurden beispielsweise auf diese Art 2.732 Testfälle durchgeführt und hat in ca. $45,5$ Stunden Zeit zur Berechnung erfordert, wobei ein Testfall etwa eine Minute an Zeit benötigt.

9.7.5 Auswertung des Experiments

Die Auswertung erfolgt zuñachst auf Basis der erzeugten CSV-Dateien. Ausgewählt werden auf der Basis die jeweiligen Anfangsgeschwindigkeiten und die jeweilige Gierrate und die daraus resultierende Abweichung von der Ideallinie zum Zeitpunkt der Auslösung. Weiterhin kann darauf aufbauend die time-to-collision (TTC) bestimmt werden.

Anhand der Abbildung 9.2 lässt sich beispielhaft aufzeigen, dass dort einige Punkte in den Verläufen bemerkenswert sind und durchaus als Anomalien gewertet werden können. Denn auf-

grund der Betrachtung der Ausweichmöglichkeit des Algorithmus sollte eine seitliche Ablage des Fahrzeugs zu einer Reduzierung der TTC führen. Bei $31\ km/h$ als max. Toleranz für die Testgeschwindigkeit ergibt sich fast eine maximale seitliche Abweichung, jedoch ist die TTC in jenem Bereich annähernd auf einem Niveau von $> 1,5\ s$. Auch bei $45\ km/h\ -\ 46\ km/h$ ist eine maximale seitlich Abweichung und eine vergleichsweise geringe Abweichung in der TTC festzustellen. Grundätzlich ist jedoch von einer Reduzierung der TTC und damit mit einer höheren Restgeschwindigkeit auszugehen, wenn die maximal zulässige Abweichung ausgereizt wird.

9.7.6 Validitätsbetrachtungen

Im folgenden wird nun die Validitätsbetrachtung vorgenommen, wobei sowohl auf die dem Experiment vorangegangenen Überlegungen und Annahmen erläutert werden.

Interne Validitätsbetrachtungen

Die vorliegende Problemstellung ist eine konkrete, durch externe (hier: außerhalb des Projekts) Stellen formulierte und vorgegebene Situation im Rahmen der Markteinführung von Neufahrzeugen. Der Consumer Test Kontext ist somit eine für Automobilhersteller relevante Thematik und muss durch sie hinreichend betrachtet werden.

Die Infrastruktur ist darauf ausgerichtet, dass die für die Simulation notwendigen Komponenten und Modelle so simpel wie möglich gehalten worden sind, damit evtl. Seiteneffekte bei der Untersuchung des Algorithmusverhaltens vermieden werden. Das gilt sowohl für das Sensormodell als auch für das Fahrdynamikmodell, so dass es dadurch möglich ist, das reine Verhalten des Algorithmus bewerten zu können.

Entscheidend für die Fragestellung ist die Entscheidungslogik des Algorithmus innerhalb der zu untersuchenden Komponente. Die jeweiligen Einflussgrößen, die für den Algorithmus notwendig sind, werden entsprechend aus der Infrastruktur geliefert. Eine Änderung der konkreten Werte wird somit eine Auswirkung auf das Verhalten der Auslösezeitpunktes haben.

Da bei manchen Simulationsläufen Abweichungen in den Auslösezeitpunkt festgestellt werden konnten, wurde zudem noch eine Fehleranalyse vorgenommen. Hierzu wurden mehrere Simulationsläufe durchgeführt und anschließend die Abweichung innerhalb einer Parameterkonfiguration auf Abweichung untereinander geprüft. Hierdurch wird eine Einschätzung gegeben, zu welchem Anteil Abweichungen auftreten.

Die Abbildung 9.3 zeigt das Ergebnis aus fünf Simulationsdurchläufen in Form der Anzahl der Häufigkeiten der jeweiligen Auslösezeitpunkte. Zusätzlich werden auch die Abweichungen von den am häufigsten aufgeführten Zeitpunkten ergänzt und in Abhängigkeit davon, ob sie oberhalb oder unterhalb des konkreten Zeitpunktes liegen, als minimaler oder maximaler Wert indiziert.

Für die rechten und linken Pfade ergeben sich bei insgesamt 1210 Simulationsläufe mit 1153 vom Zeitpunkt her gleichen Simulationen. Die verbleibenden Abweichungen aggregieren sich dabei zur Fehlerrate, die hier bei $4,7\%$ liegt. Die zeitliche Abweichung selbst beträgt hier minimal $0,04s$. Die Ursache für diese leichte Abweichung kann unterschiedliche Ursachen haben. Einerseits kann es eine funktionale Ursache innerhalb des Testobjektes, also hier dem Algorithmus zum Notbremsen haben. Andererseits kann die Ursache innerhalb der Simulationskomponenten bzw. der zugrundeliegenden Infrastruktur der Simulationsumgebung liegen. Die Ermittlung dessen im zweiten Punkt ist insbesondere durch die Nutzung von nicht eigens entwickelten Komponenten mit erheblichem Aufwand verbunden. Daher ist eine Fokussierung auf das konkrete Testobjekt zu empfehlen, da dies die Zielorientierung im Entwicklungsprozess unterstützt.

Analog ergibt sich für die idealen Pfade bei ebenfalls fünf Simulationsdurchläufen und 605 Simulationsläufen eine Anzahl von 580 Simulationen mit jeweils gleichen Zeitpunkten. Die Fehlerrate beträgt daher $4,7\%$.

Externe Validitätsbetrachtungen

Die Übertragbarkeit der erzielten Ergebnisse im Kontext lassen sich methodisch auch in einem gewissen Rahmen auf andere Fahrerassistenzsysteme überführen. Gleichwohl ist die Aussagekraft bei Verwendung gleicher Komponenten in der Infrastruktur rein auf Consumertest und ähnlichen Testszenarien im Protokoll beschränkt. Letztlich ist es jedoch möglich durch eine entsprechende Adaption zunächst der Zielsetzung (vgl. Kapitel 6) und daraus abgeleiteten Anpassungen an der Infrastruktur eine Nutzung auf Kontexte zu ermöglichen.

Inhaltlich bleiben die Ergebnisse im Experiment übertragbar auf den allgemeinen Bereich der Consumer Test, die weltweit eine Vergleichbarkeit und Qualitätssicherung bei der Sicherheit von Fahrzeugen untereinander sicherstellen. Gleichzeitig stellt das Protokoll als externe Vorgabe eine verringerte Beeinflussbarkeit von speziellen Testfällen dar, so dass eine interne subjektive Schwerpunktsetzung nicht möglich ist.

Durch Nutzung eines hohen Abstraktionsgrades bei den Sensormodellen im Rahmen der Infrastruktur sind gewonnene Erkenntnisse zum Auslöseverhalten des Algorithmus zunächst auf die funktionale Erprobung einer einzelnen Komponente möglich. Eine grundsätzliche Abschätzung

eines Ergebnisses mit einem realen Fahrzeug unter realen Bedingungen auf einem Testgelände ist an dieser Stelle noch nicht möglich. Hierzu sind weitere und wesentlich komplexere Modelle notwendig, die jedoch nur mit erheblichem Zusatzaufwand und entsprechenden Abgleichen zwischen Realität und Simulation erfolgen müssten. Zur Argumentation, aus welchen Gründen dies nicht zielführend sein kann, wurde bereits in Kapitel 5 eingegangen.

Konstruktive Validitätsbetrachtungen

Die geschaffene Infrastruktur ist in der Lage, das Verhalten im Kontext der Consumer Test Programme mit ihren vorgegebenen Szenarien und zugehörigen Toleranzen entsprechend abzubilden und auszuwerten. Das zuvor theoretische Verhalten des Algorithmus und seiner Auslösestrategie ist anhand der Experimente belegbar, gleichwohl konnten einige Anomalien als weitere Untersuchungsgegenstände weiterführender Experimente identifiziert werden.

9.7.7 Abschluss der Untersuchung

Eingangs wurden folgende Forschungsfragen aufgestellt:

- *RQ1: Wie verhält sich ein Funktionsalgorithmus eines Aktiven Sicherheitssystems in den ursprünglichen Consumer Test Szenarien CCRs, CCRm und CCRb?*
- *RQ2: Zu welchen Parameterkonstellationen tritt ein untypisches Verhalten auf?*

Grundsätzlich verhält sich der Algorithmus wie erwartet: die ideale Trajektorie führt in der Regel zu einer früheren Auslösung als die jeweiligen Trajektorien rechts- und linksseitig von der Ideallinie, gleich welches Szenario man dafür vorsieht. Auch wenn in dieser Fallstudie nur das CCRm Szenario exemplarisch für die Methodik erläutert wurde, so können in [BBH$^+$15b, BBH$^+$15a] ergänzend dazu die entsprechenden Ergebnisse eingesehen werden.

Gleichzeitig wurden einige Punkte und Anomalien aufgezeigt, die ein untypisches Verhalten aufweisen und somit nicht der erwartenden Funktionalität des Algorithmus entsprechen. Ergänzende Strategien wären in diesem Fall die weitere Erprobung in realen Testszenarien und weitere Experimente und Modellerweiterung, die eine vertiefende Wirkung und weitere Auffächerung der vorgegebenen Parametervariation bedeuten würde.

9.8 Zusammenfassung

In diesem Kapitel wurde aufgezeigt, wie Experimente im Kontext von Simulationsvorhaben durchgeführt werden und welche Besonderheiten sich ergeben. Dazu wurde zunächst eine kontextbezogene Einordnung vorgenommen. Hier wurde sich auf die bisherigen Ausführung von Runeson und Höst bezogen und wesentliche Merkmale von empirischen Studien im Software-Engineering herausgestellt. Der entscheidende Fokus liegt dabei auf die Veränderbarkeit und Kontrollierbarkeit von zu untersuchenden Parametern.

Im Weiteren wurde Aufbau, Planung und Durchführung von Experimenten, hier als geeignete Art einer Untersuchung identifiziert, erläutert und mit Besonderheiten aus dem vorliegenden Kontext der Consumer Tests ergänzt. Bei der Analyse und der Auswertung der Ergebnisse ist insbesondere wichtig, die getroffenen Annahmen und Einschränkungen sowie die Möglichkeiten der Übertragung im Rahmen der internal, external und construct validity herauszustellen.

Abgeschlossen wurde das Kapitel 9 durch die Fortsetzung der zuvor begonnenen Fallstudie zu den Consumer Tests. Dazu wurde beispielhaft ein Experiment aufgebaut und entsprechend ausgewertet, um die Methodik und ihren Umgang an dieser Stelle zu erläutern.

Das nachfolgende Kapitel 10 wird nun die beiden Forschungsfragen aus Kapitel 1 aufgreifen und beantworten und herausstellen, aus welchen Gründen die vorliegende Methodik Vorteile bietet und an welcher Stelle auch Einschränkungen gelten.

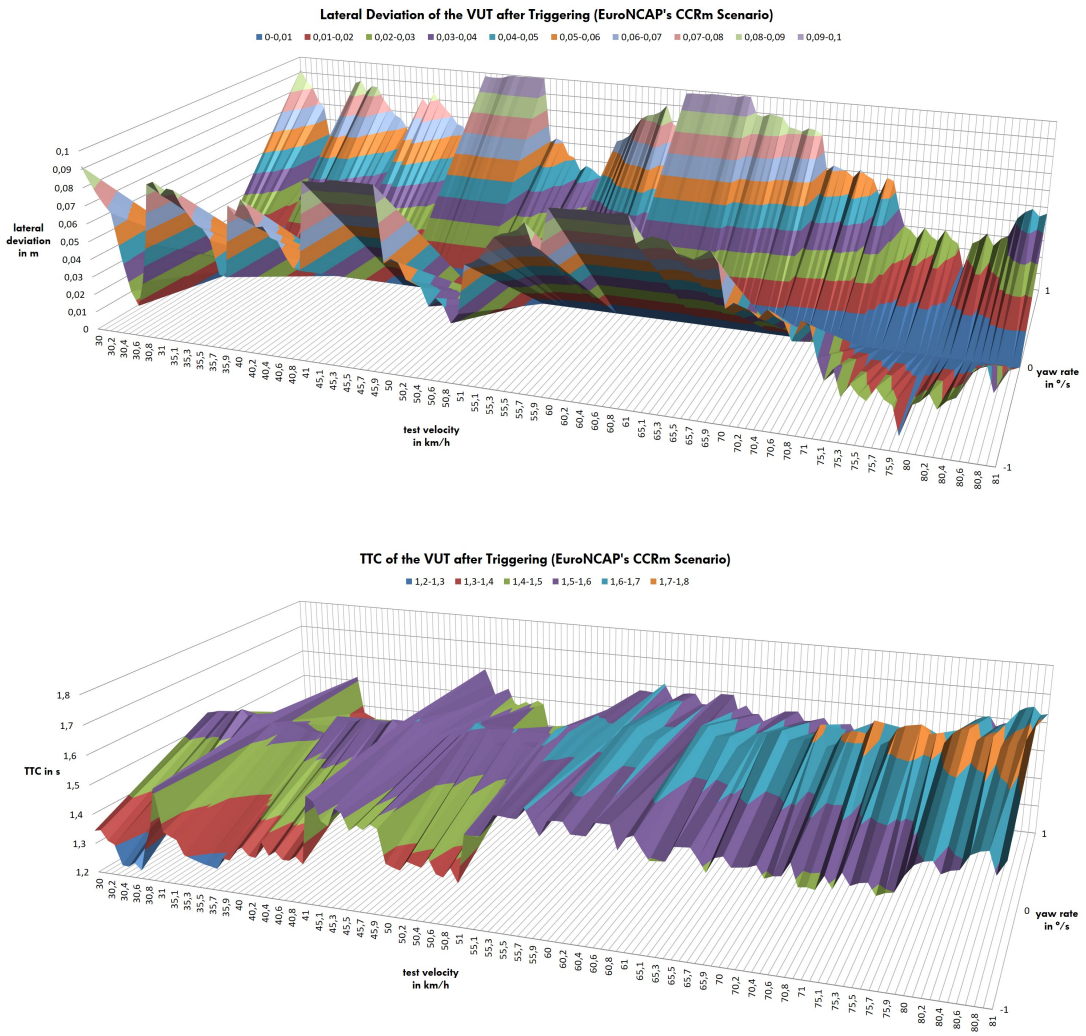


Abbildung 9.2: Beispielhafte Darstellung einer Auswertung zu EuroNCAP

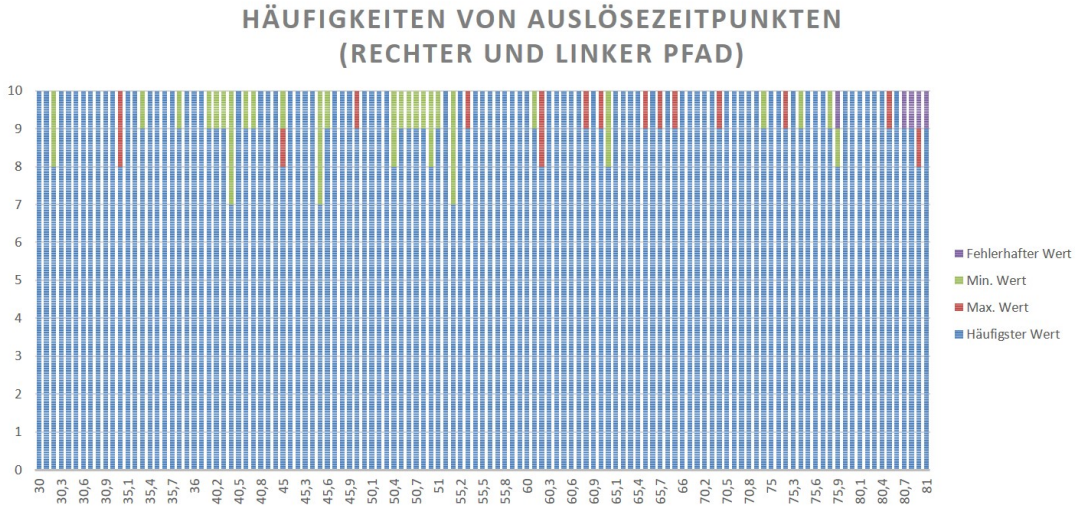


Abbildung 9.3: Häufigkeitsverteilung der Auslösezeitpunkte bei linkem und rechtem Pfad

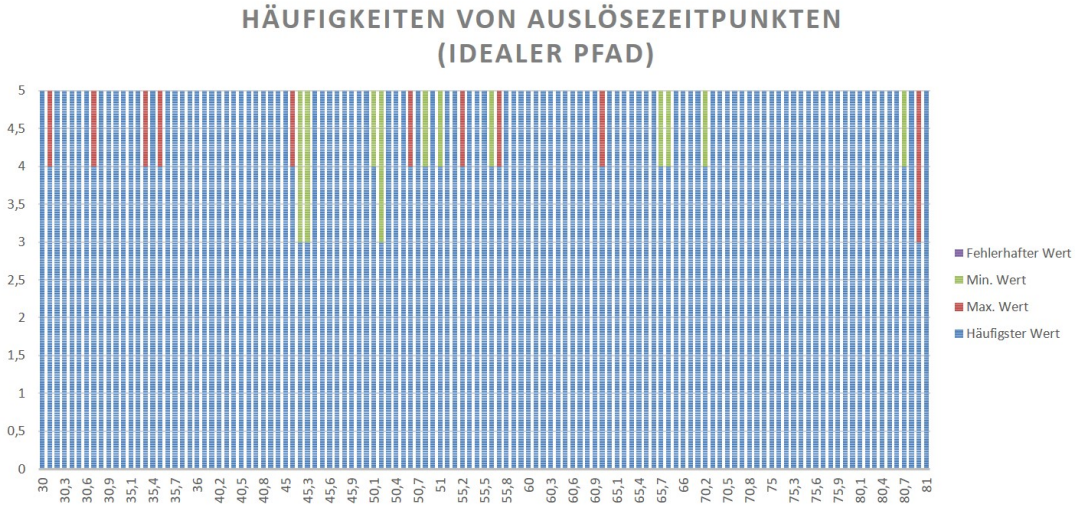


Abbildung 9.4: Häufigkeitsverteilung der Auslösezeitpunkte bei idealem Pfad

10 Diskussion der Ergebnisse und Schlussfolgerung

Dieses Kapitel stellt den Abschluss zur Methodik dar, indem zunächst die Forschungsfragen aus Kapitel 1 aufgegriffen werden. Zur Erinnerung seien diese hier nochmal aufgeführt:

- *RQ1: Welche Methoden zur systematischen Entwicklung einer Simulationsinfrastruktur für Aktive Sicherheitssysteme in Consumer Tests existieren gegenwärtig?*
- *RQ2: Wie kann die Entwicklung einer Simulationsumgebung für Aktive Sicherheitssysteme systematisiert und dadurch effizienter gestaltet werden?*
- *RQ3: Welche Erkenntnisse lassen sich simulativ bei der Bewertung von Aktiven Sicherheitssystemen im Rahmen von Consumer Tests erzielen?*

Diese Forschungsfragen stecken den Rahmen für den Inhalt und die Ergebnisse dieser Arbeit ab. Abschließend werden dann im Kontext der Threats of Validity die Ergebnisse der Arbeit behandelt.

10.1 Forschungsfrage RQ1

In Kapitel 4 ist herausgearbeitet worden, dass es eine methodische Lücke gibt, wie solche Infrastrukturen systematisch entwickelt werden können. Dazu wurde zunächst das wissenschaftliche Umfeld zur Simulation von Aktiven Sicherheitssystemen auf entsprechende Ansätze und Konzepte geprüft. Dabei wurde festgestellt, dass es nur wenige (Forschungs-)Beiträge gibt, die sich inhaltlich mit dem Kontext um Aktive Sicherheit befassen; sofern dies doch der Fall war, so handelte der Beitrag selten von Notbremssystemen. Weiterhin wurde untersucht, ob speziell im Rahmen von Consumer Test Toleranzanalysemethoden existieren, wobei entsprechende Beiträge dazu nicht in den einschlägigen Online-Bibliotheken gefunden werden konnten. Außerdem konnte auch keine übergeordnete Methodik identifiziert werden, die sich hinsichtlich der

Entwicklung der Simulationsumgebung befasst und die notwendigen Komponenten und Methodiken abseits der üblichen Modellierung des betrachteten Systems beschreibt. Somit besteht methodisch nicht nur eine Lücke zur Entwicklung einer Infrastruktur in diesem Bereich, sondern auch bei der Toleranzanalyse von entsprechenden Sicherheitskomponenten.

10.2 Forschungsfrage RQ2

Im Zuge der zweiten Forschungsfrage wurde der Entwurf einer Methodik zur agilen Entwicklung von Simulationsumgebungen vorgestellt. Diese sieht vier übergeordnete Prozessbausteine vor, die eine zielgerichtete Entwicklung ermöglichen. Im ersten Schritt wird eine Kontextanalyse durchgeführt, die insbesondere auf Methoden des modellbasierten Software Engineering zurückgreift. Hierdurch wird die Strukturierung der jeweils notwendigen Entwicklungsschritte erreicht. Dies beginnt mit der Erfassung der Aufgaben im jeweiligen Entwicklungsprozess, die meist anhand von Dokumenten und Aufzeichnungen in der jeweiligen Organisationseinheit festgehalten sind. Im Fall der Fälle und bei noch unzureichender Dokumentation ist eine Prozessanalyse entsprechend durchzuführen. Danach erfolgt die Einbettung der Simulationsaufgaben innerhalb dieser Prozesse. Auch hier wurden Methodiken vorgestellt, die bei der Strukturierung unterstützen. Das umfasst sowohl die Entwicklung geeigneter Use-Cases und ihrer Visualisierung, die Zuordnung entlang des zugrunde liegenden Entwicklungsprozesses und Definition der Zielsetzung.

Bei der Entwicklung der Infrastruktur im Zuge des zweiten Bausteins wurde hervorgehoben, dass insbesondere am Anfang weitreichende Entscheidungen für die Simulationsumgebung getroffen werden müssen. Dies betrifft sowohl die Zielgrößenauswahl als auch den Komplexitätsgrad der verwendeten Modelle zu den Komponenten. Darüber hinaus wurden zwei Artefakte in Ergänzung des Pre-Processings vorgestellt, welche im Rahmen einer Toleranzanalyse von Consumer Test Szenarien Verwendung finden. Diese Ergänzung der Methoden zur Infrastrukturentwicklung sichern den weiteren Prozess zur Entwicklung solcher Simulationsumgebung ab.

Im dritten Baustein wurde für die verbesserte Auswertung die Methodik der Meta-Metriken eingeführt, welche über Projektlaufzeit eine Qualitätsaussage über das vorliegende Testobjekt erlauben und damit eine schnelle (agile) Reaktion bei Qualitätsverschlechterung ermöglichen. Mangels Konkretisierung der Metriken im Kontext Aktiver Sicherheitssysteme wurden zudem die Funktionsmetriken eingeführt, welche als Basis für die übergeordneten Meta-Metriken dienen. Unter Verwendung der GQM Methodik und die Integration der oben genannten Metriken

in das Vorgehen konnten weitere Möglichkeiten der agilen Entwicklung geschaffen werden.

In dem vierten Baustein wurde der Umgang mit Simulationsexperimenten erläutert und durch Ergänzung der Richtlinien von Runeson und Höst eine kontextabhängige Verfeinerung im Umfeld von Aktiven Sicherheitssystemen dargelegt. Die Anwendung und Adaption auf den vorliegenden Kontext unterstützen bei der Präzisierung der Untersuchungsbedingungen und führen somit zu einem effizienten Entwicklungsprozess.

Durch die Verwendung des in dieser Arbeit beschriebenen Prozesses mit seinen vier Bausteinen wird somit eine systematische und effiziente Entwicklung von Aktiven Sicherheitssystemen gewährleistet und verbessert.

10.3 Forschungsfrage RQ3

Simulativ lassen sich zur Qualitätsbewertung von Aktiven Sicherheitssystemen insbesondere in Consumer Test relevanten Szenarien mögliche Erkenntnisse über das Verhalten von zunächst einzelnen Komponenten des Systems erzielen. Hierzu wurde technisch die Möglichkeit einer Sensitivitätsanalyse in Form einer Parametervariation innerhalb der zulässigen Toleranzen der Consumer Test Szenarien geschaffen. Mittels umfangreicher konkreter Tests auch auf Basis zunächst simpler Modelle lassen sich einige Anomalien im Verhalten der Komponente herausstellen. Durch die zunächst bewusst gewählte Unabhängigkeit und detaillierte Analyse einer Softwarekomponente kann im Vorfeld von realen Tests und entwicklungsbegleitend während der Implementierungsphase des Source Code eine erste Überprüfung erfolgen. Dies ist deshalb notwendig, weil der Äquivalenzklassentest im Rahmen der Softwaretests nicht hinreichend genug ist, das Funktionsverhalten und damit die Qualität der Software zu beschreiben.

10.4 Validitätsbetrachtungen zur Methodik

Im Folgenden soll im Rahmen der Validitätsbetrachtungen die Methodik aus einer übergeordneten Perspektive betrachtet und kritisch diskutiert werden.

10.4.1 Interne Validitätsbetrachtungen

Die betrachtete Problemstellung umfasst die Entwicklung von Simulationsmethoden aktiver Sicherheitssysteme, die so im Rahmen auch der Funktions- und Systementwicklung bei Auto-

mobilherstellern vorliegt. Insbesondere die Entwicklung solcher Simulationsumgebungen stellt immer wieder eine Herausforderung dar und führt nicht selten zu erheblichem Aufwand mit in Teilen ungewissem Ausgang, ob alle Ziele erreicht werden. Der Bedarf zur Entwicklung der hier vorgestellten Methode wurde neben dieser gemachten Erfahrung auch anhand des durchgeführten Systematic Literature Reviews identifiziert und bestätigt.

Als Erweiterung der Methodik ist die jeweilige Sensor- und Aktorikentwicklung anzuführen, da dieser Themenschwerpunkt einer eigenen Disziplin bedarf. Auch die Erweiterung weiterer Protokolle und neuer Test Szenarien wie z.B. mit Fußgängerschutz sind noch nicht in die Methodik integriert. Weiterhin wurde aus der angeführten Begründung die Wahl für simplifizierte Modelle innerhalb der Infrastruktur getroffen. Eine weitere Erweiterungsmöglichkeit stellt die Rückkopplung von realen Tests dar. Ein weiteres Potential hierzu bietet möglicherweise die Nutzung von KI-Methoden im Rahmen der Bewertung der Funktions- und Meta-Metriken, insbesondere dann, wenn die Simulationsinfrastruktur über mehrere Instanzen erweitert wird. In welchem Umfang dies stattfindet, soll allerdings im Kapitel 11 erfolgen.

10.4.2 Externe Validitätsbetrachtungen

Die Übertragbarkeit der Methodik auf andere Kontexte im Fahrerassistenzumfeld ist grundsätzlich möglich, da die Methodik darauf angelegt ist und die einzelnen Schritte im Vor- und Nachgang für die Schaffung der Infrastruktur grundlegender Natur sind. Lediglich die konkrete Implementierung der daraus abgeleiteten Infrastruktur ist auf den jeweiligen Kontext begrenzt, was sich aus der Sache selbst ergibt. Dazu zählt beispielsweise in diesem Fall, dass das Parametervariationstool die zulässigen Toleranzbereiche in Consumer Tests im Besonderen abbildet und daher auf den Consumer Test zunächst begrenzt ist. Gewiss ist das Tool auch durch entsprechende funktionale Erweiterungen übertragbar.

Aus Gründen des Ausschlusses von Seiteneffekten wurde entschieden, simplere Modelle für Sensoren und Aktuatoren zu verwenden. Dies birgt die Vorteile, dass das Testobjekt zunächst in den Vordergrund rückt und damit die Ursachen von Anomalien nicht in den Modellen selbst zu suchen sind. Außerdem kann dies helfen, den Entwicklungsaufwand und den Anpassungsbedarf der Infrastruktur zu verringern. Gleichwohl muss die Verwendung von einfachen Modellen aber auch auf die Zielsetzung hin überprüft werden, so dass dies zunächst auch zur gewünschten Zielerreichung führen kann; gegebenenfalls ist iterativ eine Verfeinerung von umfangreicheren Modellen anzustreben.

Des Weiteren ist die Übertragbarkeit der Einschränkung von Äquivalenzklassentests und ihrer

Aussagefähigkeit hinsichtlich Qualitätsverbesserung auf andere Kontexte offen. Daher gilt diese Aussage zunächst im Rahmen von Consumer Tests mit den jeweiligen Protokollvorgaben in Form der zulässigen Toleranzen. Hier ist allerdings denkbar, dass folgender Zusammenhang gesehen werden kann: Je dynamischer die Eingangsdaten eine Zustandsänderung unterliegen, desto eingeschränkter ist die Reichweite eines Äquivalenzklassentests bzw. umso stärker steigt die Menge an Repräsentanten.

10.4.3 Konstruktive Validitätsbetrachtungen

Die hier vorgestellte Methodik baut auf mehreren Methodiken auf, die unter Berücksichtigung der gegebenen Projektrahmenbedingungen systematisch einander ergänzend zusammengestellt wurden. Dabei wurden eigene Methodiken zur Lückenschließung ergänzt wie z.B. die Parametervariation oder auch die Meta-Metriken. Gleichzeitig wurden weitere Rahmenbedingungen aus dem Projektkontext unter Berücksichtigung von Informationssicherheit und Vertraulichkeit in die Methodik integriert bzw. in Teilen abstrahiert. In Form einer Fallstudie wurde die Methodik als Proof of Concept bestätigt. In einem nächsten Schritt wäre eine Anwendung in einem weiteren Projektkontext erstrebenswert.

10.5 Zusammenfassung

Dieses Kapitel widmet sich zunächst der Beantwortung der anfänglich aufgestellten Forschungsfragen. Dabei konnte aufgezeigt werden, dass es

1. ein Bedarf zur Entwicklung dieser Methodik gibt, indem umfangreich nach bereits vorhandenen Methoden im Kontext Consumer Test recherchiert wurde,
2. in Folge des identifizierten Bedarfs eine Methodik entwickelt wurde, einen Beitrag zur Systematisierung der Entwicklungstätigkeiten von Simulationsumgebungen zu bieten und
3. letztlich eine agile Qualitätsverbesserung bei Entwicklung von Consumer Test relevanten Aktiven Sicherheitssystemen zu ermöglichen.

In Ergänzung wurde auch aufgezeigt, wo die Methodik in sich aus Sicht des Autors valide ist, in welchen Bereichen eine Übertragung auf andere Kontexte möglich erscheint und wo eventuelle Einschränkungen der Methodik momentan noch zu sehen sind. Dazu zählt beispielsweise die Übertragung auf mögliche andere Kontexte in Fahrerassistenzumfeld und die Erweiterung

um weitere hinzugekommene Szenarien wie dem Fußgängerschutz bei EuroNCAP und anderen Consumer Test Organisationen.

11 Ausblick auf zukünftige Arbeitsfelder

Dieses Kapitel wird einen Ausblick darauf geben, in welchen Bereichen weitere Arbeiten und Forschungstätigkeiten konzentriert werden sollten. Dabei wird an dieser Stelle formal auf eine üblicherweise vorangestellte Zusammenfassung dieser Arbeit aus Gründen der Wiederholung verzichtet. Dieser Teil ist inhaltlich in Kapitel 10 bereits aufgeführt.

11.1 Modellbasierte und generative Methoden des Software Engineering

Im Zuge der Szenario- und Testfallerstellung werden in dieser Methodik weitere Methoden des Software Engineering insbesondere die modellbasierte und generative Softwareentwicklung eingesetzt. Weiterhin wurden in der Analysephase modellbasierte Methoden zur Erfassung der Projektrahmenbedingungen und der Zielsetzung verwendet. Da für die Entwicklung dieser Methodik eine intensive Auseinandersetzung mit den Projektbedingungen erfolgen musste, wurde der Schwerpunkt auf die Durchgängigkeit der Methodik gelegt. Dabei wurde jedoch darauf geachtet, dass anhand der Artefakte weitere Methodiken ansetzen können, um eine Weiterentwicklung der Methodik vorzunehmen.

So ist ein Ansatz die erstellten Modelle aus der Phase 1 dahingehend zu erweitern, generative Methoden verwenden zu können. Dies ist anhand der verwendeten Diagrammarten der UML/P und der Techniken zu domänenspezifischen Sprachen sowie ihrer Werkzeuge durchaus denkbar. Auch die Auswahl der verwendeten Werkzeuge im Rahmen der Infrastruktur könnten weiter modelliert und so technologieunabhängiger beschrieben werden, so dass eine dauerhafte Abhängigkeit von spezifischen Werkzeugen vermieden wird.

Es kommt hier hinzu, dass im industriellen Umfeld der Automobilhersteller generative Softwareentwicklung bisher noch kaum Verwendung gefunden hat, obwohl die Methoden bereits seit geraumer Zeit bekannt sind. Denn auch wenn Unternehmen immer nach neuen Innovationen auch im Bereich von Prozessen streben, so unterliegen alle Innovationen immer der harten

Bedingung der Wirtschaftlichkeit. Daher wundert es nicht, dass erst in der heutigen Zeit modellbasierte Methoden Einzug in die Analysen von Prozessen erhalten. Dabei wird allerdings die generative Softwareentwicklung noch nicht überzeugend eingesetzt. Worin die Ursachen darin liegen, wäre ein weiteres Feld, welches einer Untersuchung bedarf.

11.2 Einsatz künstlicher Intelligenz und Felddatengewinnung

Tesla hat in vielen Bereichen der Entwicklung von Fahrzeugen und der eingesetzten Technologie neue Wege beschritten. Unter anderem verfügt Tesla über eine umfangreiche Infrastruktur zur Felddatenerfassung als ständig wachsende Datengrundlage für die Entwicklung von Aktiven Sicherheitssystemen und anderen Fahrerassistenzsystemen. [Mul19] Hieraus lassen sich kontinuierliche Testfälle für alle Bereiche bis hin zum autonomen Fahrfunktionen dynamisch erweitern und damit eine verbesserte Funktion vor Kunde realisieren. Die Daten können auch als Trainingsmenge für Neuronale Netze dienen, die Sensoren dazu befähigen, frühzeitiger Objekte in der Umgebung zu bestimmen und entsprechend mit Zustandsattributen wie Geschwindigkeit, Lage, etc. auszustatten.

Basierend auf dem gleichen Prinzip kann auch ein Abgleich zwischen real durchgeführten Consumer-Testfällen und den virtuell durchgeführten Testfällen erfolgen, die als besonders kritisch für die Funktion von den Entwicklern bewertet werden. Darauf aufbauend können entsprechende Trajektorien gewonnen werden, welche den Untersuchungsraum reduzieren. Anschließend ist eine Konzentration auf diese Pfade und ihre Untersuchung mittels Parametervariation möglich.

Inwiefern beispielsweise auch KI-Methoden wie das Predictive Health Monitoring in diesem Zusammenhang eingesetzt werden kann, ist an dieser Stelle noch nicht betrachtet worden. Die Methodik wird in der Regel im Rahmen der Instandhaltung von Maschinen und den ihnen übergeordneten Prozessen angewendet. Es stellt sich hier die Frage, ob bspw. auf Prozessebene und hier auf einer Metaebene die Entwicklung solcher Funktionen überwacht werden könnte.

11.3 Einsatz weiterer agiler Methoden - Quo vadis, Automotive Industria?

In der Studie “Agile in Automotive - State of the Practice” [Saz14] wurde dargelegt, dass in einer Reihe von Aktivitäten im Automobilbereich agile Methoden eingesetzt werden. Dennoch

wird darauf hingewiesen, dass der Grad der Umsetzung in Teilen als nicht durchführbar erachtet wird. In [Par19] wird allerdings der Umbruch in der Automobilindustrie als Motivator aufgeführt, warum insbesondere im Projektmanagement auf Methoden wie SCRUM oder Kanban zurückgegriffen wird. Darüber hinaus bleiben allerdings einige vielversprechende Methoden wie bereits die oben erwähnte generative Softwareentwicklung mit ihren vorgelagerten modellbasierten Ansätzen noch unberücksichtigt.

Als weiterer Treiber kann mittlerweile aber auch der Aufstieg das Unternehmen Tesla Motors interpretiert werden, das mit großer Sicherheit solche Methoden umfassender einsetzt. Dies begründet auch, warum der Grad der Fremdvergabe speziell im Softwarebereich reduziert wird und darüber hinaus die eigenen Softwarekompetenzen innerhalb der Automobilhersteller gezielt aufgebaut werden. Daher ist davon auszugehen, dass die agilen Methoden auch weiterhin umfassender in Entwicklungsprozesse integriert werden.

Welche Rolle dabei Simulationsmethoden spielen werden, wird nicht zuletzt davon abhängen, ob insbesondere ein neuer Weg zur Nutzung von Simulations- und Felddaten dazu führt, Softwarefunktionen in Fahrzeugen bzgl. ihrer Qualität zu verbessern, dabei aus der Schleife der “Realitätsnachbildung” herauszukommen und den Weg abstrakterer Simulationsformen bei gleichzeitiger Sicherstellung der Kundensicherheit und -zufriedenheit zu beschreiten.

Literaturverzeichnis

[AABL12] AUWERAER, Herman V. ; ANTHONIS, Jan ; BRUYNE, Stijn D. ; LEURIDAN, Jan: Virtual engineering at work: the challenges for designing mechatronic products. In: *Engineering with Computers* 29 (2012), September, Nr. 3, 389–408. http://dx.doi.org/10.1007/s00366-012-0286-6. – DOI 10.1007/s00366–012–0286–6

[ABB+10] AMDITIS, Angelos ; BERTOLAZZI, Enrico ; BIMPAS, Matthaios ; BIRAL, Francesco ; BOSETTI, Paolo ; LIO, Mauro D. ; DANIELSSON, Lars ; GALLIONE, Alessandro ; LIND, Henrik ; SAROLDI, Andrea ; SJOGREN, Agneta: A Holistic Approach to the Integration of Safety Applications: The INSAFES Subproject Within the European Framework Programme 6 Integrating Project PReVENT. In: *IEEE Transactions on Intelligent Transportation Systems* 11 (2010), September, Nr. 3, 554–566. http://dx.doi.org/10.1109/tits.2009.2036736. – DOI 10.1109/tits.2009.2036736

[ABRM13] AZIMI, Seyed R. ; BHATIA, Gaurav ; RAJKUMAR, Ragunathan ; MUDALIGE, Priyantha: Reliable Intersection Protocols Using Vehicular Networks. In: *Proceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems*. New York, NY, USA : ACM, 2013 (ICCPS '13). – ISBN 978–1–4503–1996–6, 1–10

[ACB12] ASTUDILLO, Darwin ; CHAPUT, Emmanuel ; BEYLOT, Andre-Luc: Map update application: Performance measurements on a highway scenario. In: *2012 International Conference on Wireless Communications in Underground and Confined Areas*, IEEE, August 2012

[Ack09] ACKER, Bernd: *Simulationstechnik - Grundlagen und praktische Anwendungen*. expert Verlag, 2009

[AIB19] ARBEITSKREIS INFORMATIK BEGRIFFSNETZ, Gesellschaft für Informatik (.: *Vorgehensmodelle - Prinzip, Methode, Werk-*

zeug. http://www.informatikbegriffsnetz.de/arbeitskreise/vorgehensmodelle/themenbereiche/prinzipMethodeWerkzeug.html. Version: 2019

[Aly06] ALYOKHIN, Vadym: *Management von Softwaresystemen - Systembewertung: Metriken und Prozess*. http://www4.in.tum.de/lehre/seminare/hs/WS0506/mvs/files/Ausarbeitung_Alyokhin.pdf, 2006

[AOB12] ATTIA, R. ; ORJUELA, R. ; BASSET, M.: Coupled longitudinal and lateral control strategy improving lateral stability for autonomous vehicle. In: *2012 American Control Conference (ACC)*, IEEE, Juni 2012

[Ass18] ASSOCIATION, IEEEE S. ; SOCIETY, IEEE C. (Hrsg.): *IEEE Standard 1061*. https://standards.ieee.org/standard/1061-1998.html. Version: 2018

[ASZ+12] ALBERS, Albert ; SCHWARZ, Alexander ; ZINGEL, Christian ; SCHROETER, Jens ; BEHRENDT, Matthias ; ZELL, Andreas ; LEONE, Carmelo ; ARCATI, Antonio: System-Oriented Validation Aspects of a Driver Assistance System Based on an Accelerator-Force-Feedback-Pedal. Version: Oktober 2012. http://dx.doi.org/10.1007/978-3-642-33838-0_20. In: *Lecture Notes in Electrical Engineering*. Springer Berlin Heidelberg, Oktober 2012. – DOI 10.1007/978–3–642–33838–0_20, 221-233

[Aud14a] AUDI ELECTRONICS VENTURE GMBH: *ADTF Software-Umgebung für Applikationen und Tools*. online. http://www.audi-electronics-venture.de/aev/brand/de/leistungen/Entwicklungstools/adtf.html. Version: 2014. – Accessed at 03.06.2014

[Aud14b] AUDI ELECTRONICS VENTURE GMBH ; AUDI ELECTRONICS VENTURE GMBH (Hrsg.): *Virtual Test Drive - Simulation und Absicherung vorausschauender Funktionen in virtuellem Umfeld*. online. http://www.audi-electronics-venture.de/aev/brand/de/projekte/virtual_test_drive.html. Version: 2014. – Accessed at 03.06.2014

[AUT19] AUTOSAR: *Standards - Classic Plattform*. https://www.autosar.org/standards/classic-plattform/. Version: 2019

[AZS+11] AUKEN, R. Michael v. ; ZELLNER, John W. ; SILBERLING, Jordan Y. ; KELLY, Joseph ; CHIANG, Dean P. ; BROEN, Peter ; KIRSCH, Amanda ; SUGIMO-

TO, Yoichi: Extension of the Honda-DRI Safety Impact Methodology for the NHTSA Advanced Crash Avoidance Technology (ACAT) Program and Application to the Evaluation of an Advanced Collision Mitigation Braking System - Final Results of the ACAT-I Program. In: *SAE International Journal of Passenger Cars - Mechanical Systems* 4 (2011), April, Nr. 1, 488–508. http://dx.doi.org/10.4271/2011-01-0581. – DOI 10.4271/2011–01–0581

[BA03] BARONI, Aline L. ; ABREU, Fernando Brito e.: A Formal Library for Aiding Metrics Extraction. In: *ECOOP Workshop on Object-Oriented Re-Engineering, Darmstadt, Germany*, 2003

[BA04] BECK, Kent ; ANDRES, Cynthia: *eXtreme Programming Explained: Embrace Change*. Addison-Wesley, 2004. – ISBN 978–0321278654

[Bal82] BALZERT, Helmut: *Die Entwicklung von Software-Systemen*. 1982

[BBG+11] BILBAO, Ainara ; BRAZÁLEZ, Alfonso ; GARCÍA, Inigo ; TYBEL, Michael ; AGUIRIANO, Nerea: Virtual Instrument Cluster for enhancing the configurability of an automotive simulator. 88 (2011), Nr. 8, 957–971. https://doi.org/10.1177/0037549711425050

[BBH+13] BERGER, Christian ; BLOCK, Delf ; HONS, Christian ; KÜHNEL, Stefan ; LESCHKE, André ; RUMPE, Bernhard ; STRUTZ, Thorsten: Meta-Metrics for Simulations in Software Engineering on the Example of Integral Safety Systems. In: *Proceedings des 14. Braunschweiger Symposiums AAET 2013, Automatisierungssysteme, Assistenzsysteme und eingebettete Systeme für Transportmittel* Bd. 14, 2013, S. 136–148

[BBH+14a] BERGER, Christian ; BLOCK, Delf ; HEEREN, Sönke ; HONS, Christian ; KÜHNEL, Stefan ; LESCHKE, André ; PLOTNIKOV, Dimitri ; RUMPE, Bernhard: Simulations on Consumer Tests: A Systematic Evaluation Approach in an Industrial Case Study. In: *Proceedings of 2014 IEEE 17th International Conference on Intelligent Transportation Systems*. Piscataway, NJ, USA : IEEE Press, 2014 (IEEE ITSC '14). – ISBN 978–1–4799–6078–1/14, 1474–1480

[BBH+14b] BERGER, Christian ; BLOCK, Delf ; HEEREN, Sönke ; HONS, Christian ; KÜHNEL, Stefan ; LESCHKE, André ; PLOTNIKOV, Dimitri ; RUMPE, Bernhard: Simulations on Consumer Tests: Systematic Evaluation of Tolerance Ranges by Model-Based Generation of Simulation Scenarios. In: *30. VDI-VW-Gemeinschaftstagung Fahrerassistenz und Integrierte Sicherheit*. Düsseldorf, Ger-

many : VDI-Verlag, 2014 (VDI-Berichte; 2223). – ISBN 978–3–18–092223–2, 403–418

[BBH+15a] BERGER, Christian ; BLOCK, Delf ; HEEREN, Sönke ; HONS, Christian ; KÜHNEL, Stefan ; LESCHKE, André ; PLOTNIKOV, Dimitri ; RUMPE, Bernhard: Simulations on Consumer Tests: A Systematic Evaluation Approach in an Industrial Case Study. 7 (2015), S. 24 – 36. http://dx.doi.org/10.1109/MITS.2015.2474956. – DOI 10.1109/MITS.2015.2474956

[BBH+15b] BERGER, Christian ; BLOCK, Delf ; HONS, Christian ; KÜHNEL, Stefan ; LESCHKE, André ; PLOTNIKOV, Dimitri ; RUMPE, Bernhard: Large-Scale Evaluation of an Active Safety Algorithm with EuroNCAP and US NCAP Scenarios in a Virtual Test Environment - An Industrial Case Study. In: *Proceedings of 2015 IEEE 18th International Conference on Intelligent Transportation Systems*. Piscataway, NJ, USA : IEEE Press, 2015 (IEEE ITSC '15). – ISBN 978–1–4673–6595–6, 2280–2286

[BBW09] BALDAUF, Daniel ; BURGARD, Esther ; WITTMANN, Marc: Time perception as a workload measure in simulated car driving. In: *Applied Ergonomics* 40 (2009), Nr. 5, 929 - 935. http://dx.doi.org/http://dx.doi.org/10.1016/j.apergo.2009.01.004. – DOI http://dx.doi.org/10.1016/j.apergo.2009.01.004. – ISSN 0003–6870

[BCDG15] BENEDETTO, Francesco ; CALVI, Alessandro ; D'AMICO, Fabrizio ; GIUNTA, Gaetano: Applying telecommunications methodology to road safety for rear-end collision avoidance. In: *Transportation Research Part C: Emerging Technologies* 50 (2015), Nr. 0, 150 - 159. http://dx.doi.org/http://dx.doi.org/10.1016/j.trc.2014.07.008. – DOI http://dx.doi.org/10.1016/j.trc.2014.07.008. – ISSN 0968–090X. – Special Issue on Road Safety and Simulation

[BCR94] BASILI, Victor R. ; CALDIERA, Gianluigi ; ROMBACH, Dieter H.: The Goal Question Metric Approach. In: *Encyclopedia of Software Engineering*, John Wiley & Sons, 1994 (A Wiley-Interscience publication). – ISBN 0–471–54004–8, 0–471–37737–6

[BE06] BLUM, J. ; ESKANDARIAN, A.: Managing Effectiveness and Acceptability in Intelligent Speed Adaptation Systems. In: *2006 IEEE Intelligent Transportation Systems Conference*, IEEE, 2006

[Ber10] BERGER, Christian: *Automating Acceptance Tests for Sensor- and Actuator-based Systems on the Example of Autonomous Vehicle*, RWTH Aachen, Diss., 2010

[Ber14] BERGER, Christian: Scenario Pattern Matching in Large Sensor Recordings with Simulation Models for Cyber-Physical Systems. In: *Proceedings of the 2014 Summer Simulation Multiconference*. San Diego, CA, USA : Society for Computer Simulation International, 2014 (SummerSim '14), 38:1–38:8

[BFH+07] BISKUP, Hubert ; FISCHER, Thomas ; HESSE, Wolfgang ; MÜLLER-LUSCHNAT, Günther ; SCHESCHONK, Gert: Ein Begriffsnetz für die Software-Entwicklung. (2007), Nr. 3, S. 217–224

[BHK+14] BLOCK, Delf ; HEEREN, Sönke ; KÜHNEL, Stefan ; LESCHKE, André ; RUMPE, Bernhard ; SEREBRO, Vladislavs: Simulations on Consumer Tests: A Perspective for Driver Assistance Systems. In: *Proceedings of International Workshop on Engineering Simulations for Cyber-Physical Systems*. New York, NY, USA : ACM, 2014 (ES4CPS '14). – ISBN 978–1–4503–2614–8, 38:38–38:43

[BHW15] BOSSE, Nicola ; HEPPNER, Felix ; WEILKIENS, Tim: *Notationsübersicht UML 2.5*. `https://www.oose.de/wp-content/uploads/2012/05/UML-Notationsübersicht-2.5.pdf`. Version: 2015

[BMF07] BOCK, Thomas ; MAURER, Markus ; FARBER, Georg: Validation of the Vehicle in the Loop (VIL): A milestone for the simulation of driver assistance systems. In: *2007 IEEE Intelligent Vehicles Symposium*, IEEE, Juni 2007

[Bro10] BROY, Manfred ; GLOTZBACH, Ulrich (Hrsg.) ; SALEM, Samia (Hrsg.): *Cyber-Physical Systems - Innovation durch Software-intensive eingebettete Systeme*. Broy, Manfred, 2010

[BRR10] BERGER, Christian ; RENDEL, Holger ; RUMPE, Bernhard: Measuring the Ability to Form a Product Line from Existing Products. In: *Proceedings of the Fourth International Workshop on Variability Modelling of Software-intensive Systems (VaMoS)*, University of Duisburg-Essen, 2010

[BTR+14] BOSTELMAN, Roger ; TEIZER, Jochen ; RAY, Soumitry J. ; AGRONIN, Mike ; ALBANESE, Dominic: Methods for improving visibility measurement standards of powered industrial vehicles. In: *Safety Science* 62 (2014), Nr. 0, 257 - 270. `http://dx.doi.org/http://dx.doi.org/10.1016/j.ssci.2013.08.020`. – DOI http://dx.doi.org/10.1016/j.ssci.2013.08.020. – ISSN 0925–7535

[BZBP13] BUNGARTZ, Hans-Joachim ; ZIMMER, Stefan ; BUCHHOLZ, Martin ; PFLUEGER, Dirk: *Modellbildung und SImulation - Eine anwendungsorientierte Einfuehrung*. Springer Spektrum, Berlin, Heidelberg, 2013 https://doi.org/10.1007/978-3-642-37656-6. – ISBN 978–3–642–37655–9

[CC79] COOK, Thomas D ; CAMPBELL, D T: *Quasi-Experimentation: Design and Analysis Issues for Field Settings*. Houghton Mifflin, 1979

[CC09] CHANG, Chien-Yen ; CHOU, Ying-Ru: Development of Fuzzy-Based Bus Rear-End Collision Warning Thresholds Using a Driving Simulator. In: *IEEE Transactions on Intelligent Transportation Systems* 10 (2009), Juni, Nr. 2, 360–365. http://dx.doi.org/10.1109/tits.2009.2020204. – DOI 10.1109/tits.2009.2020204

[CLA+12] CHIEN, S. ; LI, Lingxi ; ARI, A. M. ; MEADOWS, A. ; BANVAIT, H. ; CHEN, Yaobin ; MOURY, M. T. ; WIDMANN, G. R.: A new scoring mechanism for vehicle crash imminent braking systems. In: *IEEE Intelligent Transportation Systems Magazine* 4 (2012), Nr. 4, 17–29. http://dx.doi.org/10.1109/mits.2012.2217561. – DOI 10.1109/mits.2012.2217561

[CLC13] CHIEN, Stanley ; LI, Lingxi ; CHEN, Yaobin: A Novel Evaluation Methodology for Combined Performance of Warning and Braking in Crash Imminent Braking Systems. In: *IEEE Intelligent Transportation Systems Magazine* 5 (2013), Nr. 4, 62–72. http://dx.doi.org/10.1109/mits.2013.2254176. – DOI 10.1109/mits.2013.2254176

[Com19] COMMISSION, European: *Vision Zero 2050 Statement*. https://ec.europa.eu/transport/road_safety/what-we-do_en. Version: 2019

[Cor14] CORPORATION, Elektrobit: *EB Assist ADTF - Driver Assistance Applications and Safety System Development*. online. http://automotive.elektrobit.com/driver-assistance/eb-assist-adtf. Version: 2014. – Accessed at 03.06.2014

[Cro92] CROUST, Gerhard: Modelle der Software-Entwicklung. In: *Vorgehensmodelle - Prinzip, Methode, Werkzeug*. 1992

[CS63] CAMPBELL, Donald T. ; STANLEY, Julian C.: *Experimental and Quasi-Experimental Designs for Research*. Rand McNally College Publishing, 1963 http://www.amazon.com/exec/obidos/redirect?tag=citeulike-20&path=ASIN/B000BWLQBE

[CS08] CHEN, Liang-Kuang ; SHIEH, Bo-Jun: Investigation of the Conflict Between the Driver and a Vehicle Steering Assist Controller. In: *Proceedings of the 10th WSEAS International Conference on Automatic Control, Modelling & Simulation*. Stevens Point, Wisconsin, USA : World Scientific and Engineering Academy and Society (WSEAS), 2008 (ACMOS'08). – ISBN 978–960–6766–63–3, 23–28

[CSD+12] CHEN, Yongquan ; SUN, Yuandong ; DING, Ning ; CHUNG, Wing K. ; QIAN, Huihuan ; XU, Guoqing ; XU, Yangsheng: A real-time vehicle safety system. In: *2012 IEEE/SICE International Symposium on System Integration (SII)*, IEEE, Dezember 2012

[CWY13] CHANG, Tang-Hsien ; WU, Yao-Jan ; YU, Shang-Min: Dilemma zone avoidance development: an on-board approach. In: *IET Intelligent Transport Systems* 7 (2013), März, Nr. 1, 87–94. http://dx.doi.org/10.1049/iet-its.2011.0066. – DOI 10.1049/iet–its.2011.0066

[CYSK14] CHOI, Jaewoong ; YI, Kyongsu ; SUH, Jeeyoon ; KO, Bongchul: Coordinated Control of Motor-Driven Power Steering Torque Overlay and Differential Braking for Emergency Driving Support. In: *IEEE Transactions on Vehicular Technology* 63 (2014), Februar, Nr. 2, 566–579. http://dx.doi.org/10.1109/tvt.2013.2279719. – DOI 10.1109/tvt.2013.2279719

[DHAT14] DRESSLER, Falko ; HARTENSTEIN, Hannes ; ALTINTAS, Onur ; TONGUZ, Ozan: Inter-vehicle communication: Quo vadis. In: *IEEE Communications Magazine* 52 (2014), Juni, Nr. 6, 170–177. http://dx.doi.org/10.1109/mcom.2014.6829960. – DOI 10.1109/mcom.2014.6829960

[DKV00] DEURSEN, Arie van ; KLINT, Paul ; VISSER, Joost: Domain-specific Languages: An Annotated Bibliography. 35 (2000), Nr. 6, S. 26–36. http://dx.doi.org/10.1145/352029.352035. – DOI 10.1145/352029.352035. – ISSN 0362–1340

[DT10a] DOSHI, Anup ; TRIVEDI, Mohan M.: Attention estimation by simultaneous observation of viewer and view. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, IEEE, Juni 2010

[DT10b] DOSHI, Anup ; TRIVEDI, Mohan M.: Communicating Driver Intents: A Layered Architecture for Cooperative Active Safety Applications. In: *13th International IEEE Conference on Intelligent Transportation Systems*, IEEE, September 2010

[Dyn19a] DYNAMICS, AB: *ADAS Development using ABD Driving Robots - Application Note AN6311 - Issue 3*. 2019

[Dyn19b] DYNAMICS, AB ; BEST, Anthony (Hrsg.): *Company Website*. www.abdynamics.com/en/products/track-testing/driving-robots/. Version: 2019

[EFR07] EUSGELD, Irene ; FREILING, Felix C. ; REUSSNER, Ralf: *Dependability Metrics: Advanced Lectures*. Springer, 2007

[ENCAP13] EUROPEAN NEW CAR ASSESSMENT PROGRAMME, The: Test Protocol - AEB Systems / EuroNCAP. 2013 (1.0). – Test Protocol

[ENCAP16] EUROPEAN NEW CAR ASSESSMENT PROGRAMME, The: Assessment Protocol - Overall Rating / EuroNCAP. 2012 - 2016 (6.0). – Test Protocol

[Eur14] EURONCAP: *EuroNCAP - The Official Site of the European New Car Assessment Programme*. http://www.euroncap.com/, June 2014

[FBML98] FISCHER, Thomas ; BISKUP, Hubert ; MÜLLER-LUSCHNAT, Günther: Begriffliche Grundlagen für Vorgehensmodelle. Version: 1998. http://dx.doi.org/10.1007/978-3-663-05994-3_1. In: *Vorgehensmodelle für die betriebliche Anwendungsentwicklung*. Vieweg und Teubner Verlag, 1998. – DOI 10.1007/978–3–663–05994–3_1, 13–31

[Flo08] FLOHR, Thomas: Defining Suitable Criteria for Quality Gates. In: *Proceedings of the International Conferences on Software Process and Product Measurement*, Springer-Verlag, 2008 (IWSM/Metrikon/Mensura '08), S. 245–256

[FP98] FENTON, Norman E. ; PFLEEGER, Shari L.: *Software Metrics: A Rigorous and Practical Approach*. 2nd. PWS Publishing Co., 1998

[Gab12] GABLER, Wirtschaftslexikon: *Definition Simulation*. http://Wirtschaftslexikon.gabler.de/definition/simulation.html. Version: Mai 2012. – zuletzt abgerufen am 16. Mai 2012

[GBK12] GARATE, Virginia R. ; BOURS, Roy ; KIETLINSKI, Kajetan: Numerical modeling of ADA system for vulnerable road users protection based on radar and vision sensing. In: *2012 IEEE Intelligent Vehicles Symposium*, IEEE, Juni 2012

[GDN+12] GRUYER, Dominique ; DEMMEL, Sebastien ; NOVEL, Brigitte d'Andrea ; LAMBERT, Alain ; RAKOTONIRAINY, Andry: Simulation architecture for the design of

Cooperative Collision Warning systems. In: *2012 15th International IEEE Conference on Intelligent Transportation Systems*, IEEE, September 2012

[GKB12] GANSLMEIER, Thomas ; KELLNER, Martin ; BRUEGGE, Bernd: An approach to develop location-based efficiency systems without real test drives. In: *2012 IEEE International Electric Vehicle Conference*, IEEE, März 2012

[GKG13] GORMAN, Thomas I. ; KUSANO, Kristofer D. ; GABLER, Hampton C.: Model of fleet-wide safety benefits of Lane Departure Warning systems. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, IEEE, Oktober 2013

[GKR+08] GRÖNNIGER, Hans ; KRAHN, Holger ; RUMPE, Bernhard ; SCHINDLER, Martin ; VÖLKEL, Steven: MontiCore: a framework for the development of textual domain specific languages. In: SCHÄFER, Wilhelm (Hrsg.) ; DWYER, Matthew B. (Hrsg.) ; GRUHN, Volker (Hrsg.): *ICSE Companion*, ACM, 2008. – ISBN 978–1–60558–079–1, 925-926

[GKRS06] GRÖNNIGER, Hans ; KRAHN, Holger ; RUMPE, Bernhard ; SCHINDLER, Martin: Integration von Modellen in einen codebasierten Softwareentwicklungsprozess. In: MAYER, H.C. (Hrsg.) ; BREU, R. (Hrsg.): *Proceedings of the Modellierung 2006* Bd. 82. Innsbruck, Tirol, Österreich : Gesellschaft für Informatik, März 2006 (LNI). – ISBN 3–88579–176–5, 67–81

[GLB+13] GHAFOOR, Kayhan Z. ; LLORET, Jaime ; BAKAR, Kamalrulnizam A. ; SADIQ, Ali S. ; MUSSA, Sofian A.: Beaconing Approaches in Vehicular Ad Hoc Networks: A Survey. In: *Wirel. Pers. Commun.* 73 (2013), Dezember, Nr. 3, 885–912. http://dx.doi.org/10.1007/s11277-013-1222-9. – DOI 10.1007/s11277–013–1222–9. – ISSN 0929–6212

[Glo05] GLOBKE, Wolfgang: *Software-Metriken*. http://www.math.kit.edu/iag2/globke/seite/semi/nar/media/metriken.pdf. Version: Juni 2005

[Gmb14] GMBH, Vires S. ; DUPUIS, Marcus (Hrsg.): *Virtual Test Drive*. online. http://vires.com/products.html. Version: 2014. – Accessed at 03.06.2014

[GPR06] GRUHN, Volker ; PIEPER, Daniel ; RÖTTGERS, Carsten: *MDA*. Springer, 2006. – ISBN 978–3–540–28744–5

[GPSV09] GIETELINK, O.J. ; PLOEG, J. ; SCHUTTER, B. D. ; VERHAEGEN, M.: Deve-

lopment of a driver information and warning system with vehicle hardware-in-the-loop simulations. In: *Mechatronics* 19 (2009), Nr. 7, 1091 - 1104. http://dx.doi.org/10.1016/j.mechatronics.2009.04.012. – DOI 10.1016/j.mechatronics.2009.04.012. – ISSN 0957–4158. – Special Issue on Hardware-in-the-loop simulation

[GPW+13] GENEDER, S. ; PFISTER, F. ; WILHELM, C. ; ARNOLD, A. ; SCHERRMANN, P. ; DOHMEN, H.-P.: Energy Flux Simulation on a Vehicle Test Bed for Validating the Efficiency of Different Driving and Assistance Systems. Version: September 2013. http://dx.doi.org/10.1007/978-3-319-01884-3_1. In: *Sustainable Automotive Technologies 2013*. Springer International Publishing, September 2013. – DOI 10.1007/978–3–319–01884–3_1, 3–13

[GSSZ13] *Kapitel* -. In: GONTER, Mark ; SCHWARZ, Thomas ; SEIFFERT, Ulrich ; ZOBEL, Robert: *Fahrzeugsicherheit*. Springer Fachmedien Wiesbaden, 2013. – ISBN 978–3–658–01690–6

[Han08] HANNAWALD, Lars: *Multivariate Bewertung zukünftiger Fahrzeugsicherheit*, Diss., 2008

[HBQS13] HASSAN, Bassem ; BERSSENBRUGGE, Jan ; QAISI, Imad A. ; STOCKLEIN, Jorg: Reconfigurable driving simulator for testing and training of advanced driver assistance systems. In: *2013 IEEE International Symposium on Assembly and Manufacturing (ISAM)*, IEEE, Juli 2013

[HE11] HEISSING, Bernd ; ERSOY, Metin: Chassis Components. Version: 2011. http://dx.doi.org/10.1007/978-3-8348-9789-3_3. In: *Chassis Handbook*. Vieweg und Teubner, 2011. – DOI 10.1007/978–3–8348–9789–3_3, 149–381

[HH10] HAAS, Jason J. ; HU, Yih-Chun: Communication requirements for crash avoidance. In: *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking - VANET '10*, ACM Press, 2010

[HKE+13] HULSHOF, Wesley ; KNIGHT, Iain ; EDWARDS, Alix ; AVERY, Matthew ; GROVER, Colin: Autonomous emergency braking test results. In: *Proceedings of the 23rd International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2013

[HMF92] HESSE, W. ; MERBETH, G. ; FRÖLICH, R.: *Software-Entwicklung - Vorgehensmodelle, Projektführung,Produktverwaltung*. Bd. Band 5.3. Oldenbourg, 1992

[HO00] HIRAMATSU, Machiko ; OBARA, Hideo: Rear-end collision scenarios categorized by type of human error. In: *{JSAE} Review* 21 (2000), Nr. 4, 535 - 541. http://dx.doi.org/10.1016/S0389-4304(00)00067-9. – DOI 10.1016/S0389–4304(00)00067–9. – ISSN 0389–4304

[HR17] HÖLLDOBLER, Katrin ; RUMPE, Bernhard: *MontiCore 5 Language Workbench Edition 2017*. Shaker Verlag, 2017 (Aachener Informatik-Berichte, Software Engineering, Band 32). http://www.se-rwth.de/phdtheses/MontiCore-5-Language-Workbench-Edition-2017.pdf. – ISBN 978–3–8440–5713–3

[HRR12] HABER, Arne ; RINGERT, Jan O. ; RUMPE, Bernhard: MontiArc - Architectural Modeling of Interactive Distributed and Cyber-Physical Systems / RWTH Aachen University. Version: February 2012. http://www.se-rwth.de/publications/. 2012 (AIB-2012-03). – Technical Report

[HSS+11] HELMER, Thomas ; SCULLION, Paul ; SAMAHA, Randa R. ; EBNER, Adrian ; KATES, Ronald: Predicting the Injury Severity of Pedestrians in Frontal Vehicle Crashes based on Empirical, In-depth Accident Data. In: *International Journal of Intelligent Transportation Systems Research* 9 (2011), Juli, Nr. 3, 139–151. http://dx.doi.org/10.1007/s13177-011-0036-y. – DOI 10.1007/s13177–011–0036–y

[HTP+10] HENDRIKS, F. ; TIDEMAN, M. ; PELDERS, R. ; BOURS, R. ; LIU, X.: Development tools for active safety systems: Prescan and VeHIL. In: *Proceedings of 2010 IEEE International Conference on Vehicular Electronics and Safety*, IEEE, Juli 2010

[HY06] HAN, Donghoon ; YI, Kyongsu: Evaluation of adaptive cruise control algorithms on a virtual test track. In: *2006 American Control Conference*, IEEE, 2006

[HYRS09] HARB, Rami ; YAN, Xuedong ; RADWAN, Essam ; SU, Xiaogang: Exploring precrash maneuvers using classification trees and random forests. In: *Accident Analysis & Prevention* 41 (2009), Nr. 1, 98 - 107. http://dx.doi.org/10.1016/j.aap.2008.09.009. – DOI 10.1016/j.aap.2008.09.009. – ISSN 0001–4575

[HyY06] HAN, Donghoon ; YI, Kyongsu ; YI, Seungjong: Evaluation of Integrated ACC(Adaptive Cruise Control)/CA(Collision Avoidance) on a Virtual Test Track. In: *2006 SICE-ICASE International Joint Conference*, IEEE, 2006

[HZHW11] HULSEN, Michael ; ZOLLNER, J. M. ; HAEBERLEN, Nino ; WEISS, Christian: Asynchronous real-time framework for knowledge-based intersection assistance.

In: *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, IEEE, Oktober 2011

[IAY13] ITOH, Makoto ; ABE, Genya ; YAMAMURA, Tomohiro: Effects of arousing attention on distracted driver's following behaviour under uncertainty. In: *Cognition, Technology & Work* 16 (2013), Mai, Nr. 2, 271–280. http://dx.doi.org/10.1007/s10111-013-0264-9. – DOI 10.1007/s10111–013–0264–9

[ITW12] ITWISSEN: *Definition Simulation*. http://www.itwissen.info/definition/lexikon/Simulation-simulation.html. Version: Mai 2012. – zuletzt abgerufen am 20. Juli 2012

[ITW18] ITWISSEN: *Objektorientierte Software-Metrik*. https://www.itwissen.info/Objektorientierte-Software-Metrik.html. Version: 2018

[Jen09] JENNY, Heinz: System capability. In: *MTZ worldwide* 70 (2009), Oktober, Nr. 10, 30–34. http://dx.doi.org/10.1007/bf03227978. – DOI 10.1007/bf03227978

[JFWA13] JENSEN, Matthew ; FREEMAN, Paul ; WAGNER, John ; ALEXANDER, Kim: Controller design and evaluation for vehicle run-off-the-road and recovery. In: *2013 European Control Conference (ECC)*, IEEE, Juli 2013

[JLC+13] JEONG, C. ; LEE, Y. ; CHOI, S. ; JUNG, D. ; LEE, K.: Comparison of driving characteristics between drivers in Korea and in the united states of America based on driver-vehicle interaction field database. In: *International Journal of Automotive Technology* 14 (2013), Januar, Nr. 1, 123–132. http://dx.doi.org/10.1007/s12239-013-0014-2. – DOI 10.1007/s12239–013–0014–2

[Jor06] JORDAN, Rüdiger: *Objekthypothesen für Sicherheitsfunktionen auf Basis eines Radar-Sensors*, Universitüt Fridericiana Karlsruhe, Diss., 2006

[KAB13] KAISER, Bernhard ; AUGUSTIN, Berhnard ; BAUMANN, Christoph: Von der Komponenten- zur Funktionsorientierten Entwicklung in der Funktionalen Sicherheit. In: *VDI-Berichte* Bd. 2188, 2013

[KC07] KITCHENHAM, Barbara ; CHARTERS, Stuart: Guidelines for performing Systematic Literature Review in Software Engineering / Keele University and Durham University. 2007. – techreport

[KEB07] KOBER, Werner ; EHMANN, Martin ; BUTZ, Torsten: Model-based development of an integrated vehicle dynamics control. In: *ATZ worldwide* 109 (2007), Okto-

ber, Nr. 10, 13–16. http://dx.doi.org/10.1007/bf03224961. – DOI 10.1007/bf03224961

[KG12] KUSANO, Kristofer D. ; GABLER, Hampton C.: Model of collision avoidance with lane departure warning in real-world departure collisions with fixed roadside objects. In: *2012 15th International IEEE Conference on Intelligent Transportation Systems*, IEEE, September 2012

[KHP+15] KÜHNEL, Stefan ; HABER, Arne ; PLOTNIKOV, Dimitri ; RUMPE, Bernhard ; CHRISTIAN, Berger: *Projektbericht Fahrsimulation*. 2015

[KO03] KASNAKOGLU, C. ; OZGUNER, U.: Recent advances on the OSU virtual environment system test-bed and its applications. In: *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings (Cat. No.03TH8683)*, IEEE, 2003

[KPS14] KRÜGER, Dirk ; PARCHMANN, Ilka ; SCHECKER, Horst ; KRÜGER, Dirk (Hrsg.) ; PARCHMANN, Ilka (Hrsg.) ; SCHECKER, Horst (Hrsg.): *Methoden in der naturwissenschaftsdidaktischen Forschung*. Springer Berlin Heidelberg, 2014. http://dx.doi.org/10.1007/978-3-642-37827-0. http://dx.doi.org/10.1007/978-3-642-37827-0

[KQW15] KUANG, Yan ; QU, Xiaobo ; WANG, Shuaian: A tree-structured crash surrogate measure for freeways. In: *Accident Analysis & Prevention* 77 (2015), Nr. 0, 137 - 148. http://dx.doi.org/10.1016/j.aap.2015.02.007. – DOI 10.1016/j.aap.2015.02.007. – ISSN 0001–4575

[Kra10] KRAHN, Holger ; RUMPE, Bernhard (Hrsg.): *MontiCore: Agile Entwicklung von domänenspezifischen Sprachen im Software-Engineering*. Bd. 1. Shaker Verlag, 2010 www.se-rwth.de/publications. – ISBN 978–3–8322–8948–5

[KRV07] KRAHN, Holger ; RUMPE, Bernhard ; VÖLKEL, Steven: Integrated Definition of Abstract and Concrete Syntax for Textual Languages. In: ENGELS, G. (Hrsg.) ; OPDYKE, B. (Hrsg.) ; SCHMIDT, D.C. (Hrsg.) ; WEIL, F. (Hrsg.): *Proceedings of Model Driven Engineering Languages and Systems (MODELS'11)* Bd. 4735. Nashville, TN, USA : Springer, Germany, October 2007 (LNCS). – ISBN 978–3–540–75208–0, 286–300

[KRV08] KRAHN, Holger ; RUMPE, Bernhard ; VÖLKEL, Steven: Monticore: Modular Development of Textual Domain Specific Languages. In: PAIGE, R.F. (Hrsg.) ; MEYER, B. (Hrsg.): *Proceedings of the 46th International Conference Objects, Models,*

Components, Patterns (TOOLS-Europe) Bd. 11. Zurich, Switzerland : Springer, Germany, July 2008 (LNBIP). – ISBN 978–3540698234, 297–315

[KS97] KHALIL, T. B. ; SHEH, M. Y.: Vehicle Crashworthiness and Occupant Protection in Frontal Impact by FE Analysis — An Integrated Approach. Version: 1997. http://dx.doi.org/10.1007/978-94-011-5796-4_15. In: *Crashworthiness of Transportation Systems: Structural Impact and Occupant Protection*. Springer Netherlands, 1997. – DOI 10.1007/978–94–011–5796–4_15, 363–399

[KS15] KECHER, C. ; SALVANOS, A.: *UML 2.5: das umfassende Handbuch ; [UML lernen und effektiv in Projekten einsetzen; alle Diagramme und Notationselemente im Überblick; mit zahlreichen Praxisbeispielen in Java und C#; inkl. Poster mit allen Diagrammtypen]*. Rheinwerk Verlag, 2015 (Galileo Computing). https://books.google.de/books?id=65IIogEACAAJ. – ISBN 9783836229777

[KSL12] KUNZ, Andreas ; SCHICK, Bernhard ; LANGE, Steffen: Predictive Energy Management Strategies in Virtual Driving Tests: Early Evaluation of Networked Controller Functions in Realistic Use Cases. Version: November 2012. http://dx.doi.org/10.1007/978-3-642-33738-3_35. In: *Lecture Notes in Electrical Engineering*. Springer Berlin Heidelberg, November 2012. – DOI 10.1007/978–3–642–33738–3_35, 1351–1361

[KZR13] KAMALANATHSHARMA, Raj K. ; ZOHDY, Ismail H. ; RAKHA, Hesham A.: Public perception on increasing use of technology in automobiles: Survey findings. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, IEEE, Oktober 2013

[LCY+14] LEE, Junyung ; CHOI, Jaewoong ; YI, Kyongsu ; SHIN, Minyong ; KO, Bongchul: Lane-keeping assistance control algorithm using differential braking to prevent unintended lane departures. In: *Control Engineering Practice* 23 (2014), Nr. 0, 1 - 13. http://dx.doi.org/10.1016/j.conengprac.2013.10.008. – DOI 10.1016/j.conengprac.2013.10.008. – ISSN 0967–0661

[LDA97] LEHNER, Franz ; DUMKE, Reiner ; ABRAN, Alain: *Software-Metrics: Research and Practice in Software Measurement*. Dt. Univ.Verlag, 1997

[Lee07] LEE, Edward A.: Computing Foundations and Practice for Cyber-Physical Systems: A Preliminary Report / University of California, Berkeley. Version: 2007.

http://chess.eecs.berkeley.edu/pubs/306.html. Berkeley, CA, USA, 2007 (UCB/EECS-2007-72). – Forschungsbericht. – 1–27 S.

[Lee08] LEE, Edward A.: Cyber Physical System: Design Challenges / Eletrical Engineering and Computer Sciences, University of California at Berkeley. 2008 (UCB/EECS-2008-8). – Forschungsbericht

[Lig09] LIGGESMEYER, Peter: *Software-Qualität - Testen, Analysieren, Verifizieren von Software*. Spektrum-Verlag, 2009

[LK19] LANGENFELD, Marc ; KULAWIK, Jonas ; AG, Volkswagen (Hrsg.): *Volkswagen stärkt neue Software-Organisation*. 2019

[LKS+13] *Kapitel* 8. In: LACHMAYER, Roland ; KUNKEL, Bernd ; SCHARNHORST, Thomas ; SCHWAB, Gabriel ; BRABETZ, Ludwig ; LEOHOLD, Jürgen ; DUDENBOSTEL, Dirk ; SCHNEIDER, Klaus ; VOLK, Thomas ; PFAFF, Wolfgang ; ABEL, Heinz-Bernhard ; BLUME, Heinrich-Jochen ; HEYEN, Gerhard ; KREYE, Markus ; SCHNEIDER, Guido ; KNOLL, Peter ; KASTIES, Günther ; LEMMER, Karsten ; BROY, Manfred ; HELBIG, Jörg ; GANZELMEIER, Lothar: *Elektrik/Elektronik/-Software*. Springer Fachmedien Wiesbaden, 2013. – ISBN 978–3–658–01690–6

[LW11] LIN, Bin ; WU, Changxu: Mathematical Modeling of the Human Cognitive System in Two Serial Processing Stages With Its Applications in Adaptive Workload-Management Systems. In: *IEEE Transactions on Intelligent Transportation Systems* 12 (2011), März, Nr. 1, 221–231. http://dx.doi.org/10.1109/tits.2010.2081359. – DOI 10.1109/tits.2010.2081359

[Mar11] MARWEDEL, Peter ; DUTT, Nikil D. (Hrsg.) ; MARWEDEL, Manfred (Hrsg.) ; MARTIN, Grant (Hrsg.): *Embedded System Design*. 2. Springer Dordrecht Heidelberg London New York, 2011

[May19] MAYER, Bettina: *Warum schon Level 3 solange dauert*. https://www.auto-motor-und-sport.de/tech-zukunft/autonomes-fahren-leitfaden-fuer-behoerden/. Version: 2019

[MKD08] MA, Jianming ; KOCKELMAN, Kara M. ; DAMIEN, Paul: A multivariate Poisson-lognormal regression model for prediction of crash counts by severity, using Bayesian methods. In: *Accident Analysis & Prevention* 40 (2008), Nr. 3, 964 - 975. http://dx.doi.org/10.1016/j.aap.2007.11.002. – DOI 10.1016/j.aap.2007.11.002. – ISSN 0001–4575

[MKS13] MERCEP, L. ; KNOLL, A. ; SPIEGELBERG, G.: Context processing for automotive human-machine interfaces. In: *2013 Science and Information Conference*, 2013. – ISSN null, S. 979–984

[MPH+97] MCMILLAN, N.J. ; PANE, D.B. ; HADDEN, J.A. ; NARENDRAN, V.K. ; EVERSON, J.H.: Statistics-based simulation methodology for evaluating collision countermeasure systems performance. In: *Proceedings of Conference on Intelligent Transportation Systems*, IEEE, 1997

[MS15] MENG, Fanxing ; SPENCE, Charles: Tactile warning signals for in-vehicle systems. In: *Accident Analysis & Prevention* 75 (2015), Nr. 0, 333 - 346. http://dx.doi.org/10.1016/j.aap.2014.12.013. – DOI 10.1016/j.aap.2014.12.013. – ISSN 0001–4575

[MSCK13] MIN, Kyoungwon ; SON, Haengseon ; CHOE, Yoonsik ; KIM, Yong-Goo: Real-time pedestrian detection based on A hierarchical two-stage Support Vector Machine. In: *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, IEEE, Juni 2013

[MSK+09] MIEGLER, Maximilian ; SCHIEBER, Reinhard ; KERN, Andreas ; GANSLMEIER, Thomas ; NENTWIG, Mirko: Hardware-in-the-loop test of advanced driver assistance systems. In: *ATZelektronik worldwide* 4 (2009), September, Nr. 5, 4–9. http://dx.doi.org/10.1007/bf03242234. – DOI 10.1007/bf03242234

[Mul19] MULLER, Joann ; AXIOS (Hrsg.): *What Tesla Knows About You*. https://bit.ly/3GTnQ9Y. Version: 2019

[NC14] NEUMANN-COSEL, Kilian v.: *Virtual Test Drive*. München, Technische Universität München, Dissertation, 2014

[NCDW09] NEUMANN-COSEL, Kilian v. ; DUPUIS, Marius ; WEISS, Christian: Virtual Test Drive - Provision Of a Consistent Tool-Set For [D,H,S,V]-In-The-Loop. In: *Proceedings on Driving Simulation Conference Europe, 2009, Monaco*, 2009

[NEI12] NOTH, Sebastian ; EDELBRUNNER, Johann ; IOSSIFIDIS, Ioannis: An integrated architecture for the development and assessment of ADAS. In: *2012 15th International IEEE Conference on Intelligent Transportation Systems*, IEEE, September 2012

[NHTSA14] NATIONAL HIGHWAY TRAFFIC SAFETY ADMINISTRATION, The: Crash Immi-

nent Braking System Performance Evaluation (Draft) / NHTSA. 2014. – Test Protocol

[Nie05] NIEDERMAYR, Christoph: *Hauptseminar Automotive Software Engineering - Testen,*
Rapid Prototyping und X-in-the-Loop. http://www4.in.tum.de/lehre/seminare/hs/WS0405/\automotive/Ausarbeitung_Christoph_Niedermayr.pdf. Version: Januar 2005. – Zusammenstellung

[NMS12] NENTWIG, Mirko ; MIEGLER, Maximilian ; STAMMINGER, Marc: Concerning the applicability of computer graphics for the evaluation of image processing algorithms. In: *2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012)*, IEEE, Juli 2012

[NS10] NENTWIG, Mirko ; STAMMINGER, Marc: A method for the reproduction of vehicle test drives for the simulation based evaluation of image processing algorithms. In: *13th International IEEE Conference on Intelligent Transportation Systems*, IEEE, September 2010

[NS11] NENTWIG, Mirko ; STAMMINGER, Marc: Hardware-in-the-loop testing of computer vision based driver assistance systems. In: *2011 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, Juni 2011

[NSM11] NENTWIG, Mirko ; SCHIEBER, Reinhard ; MIEGLER, Maximilian: Hardware-in-the-Loop Testing of Advanced Driver Assistance Systems. In: *ATZelektronik worldwide* 6 (2011), August, Nr. 4, 10–15. http://dx.doi.org/10.1365/s38314-011-0034-5. – DOI 10.1365/s38314–011–0034–5

[NWJ+11] NORFLEET, Dionne ; WAGNER, John R. ; JENSEN, Matthew ; ALEXANDER, Kim ; PIDGEON, Philip: Automotive Simulator Based Novice Driver Training with Assessment. In: *SAE Technical Paper Series*, SAE International, April 2011

[OMG11] OMG: Semantics of a Foundational Subset for Executable UML Models (fUML), v1.0 / Object Management Group. Version: 2011. https://www.omg.org/. 2011. – Forschungsbericht

[OMG12] OMG: Unified Modeling Language Specification, Version 2.4.1 / Object Management Group. Version: 2012. https://www.omg.org/spec/UML/2.5/About-UML/. 2012. – Forschungsbericht

[OMG15] OMG: Unified Modeling Language Specification, Version 2.5 / Object Mana-

gement Group. Version: 2015. https://www.omg.org/spec/UML/2.5/About-UML/. 2015. – Forschungsbericht

[Ora] ORACLE: *VirtualBox*. https://www.virtualbox.org/. – Accessed at 10.06.2014

[Par19] PARTNER, Be: *Entwicklung als Kollektivleistung: Wie agile Projektmanagement Methoden die Automobilindustrie revolutionierten*. http://blog.bepartner.de/agile-projektmanagement-methoden/. Version: 2019

[PBT10] PANOU, Maria C. ; BEKIARIS, Evangelos D. ; TOULIOU, Aikaterini A.: ADAS module in driving simulation for training young drivers. In: *13th International IEEE Conference on Intelligent Transportation Systems*, IEEE, September 2010

[PGN+10] PEREZ, Antonio ; GARCIA, M I. ; NIETO, Manuel ; PEDRAZA, Jose L. ; RODRIGUEZ, Santiago ; ZAMORANO, Juan: Argos: An Advanced In-Vehicle Data Recorder on a Massively Sensorized Vehicle for Car Driver Behavior Experimentation. In: *IEEE Transactions on Intelligent Transportation Systems* 11 (2010), Juni, Nr. 2, 463–473. http://dx.doi.org/10.1109/tits.2010.2046323. – DOI 10.1109/tits.2010.2046323

[PHHJ10] PING, Em P. ; HUDHA, Khisbullah ; HARUN, Mohd Hanif B. ; JAMALUDDIN, Hishamuddin: Hardware-in-the-loop simulation of automatic steering control: Outer-loop and inner-loop control design. In: *2010 11th International Conference on Control Automation Robotics & Vision*, IEEE, Dezember 2010

[PPA+10] PENNA, Mauro D. ; PAASSEN, Marinus M. ; ABBINK, David A. ; MULDER, Mark ; MULDER, Max: Reducing steering wheel stiffness is beneficial in supporting evasive maneuvers. In: *2010 IEEE International Conference on Systems, Man and Cybernetics*, IEEE, Oktober 2010

[RAB+12] RAKOUTH, Heri ; ALEXANDER, Paul ; BROWN, Andrew ; KOSIAK, Walter ; FUKUSHIMA, Masao ; GHOSH, Lali ; HEDGES, Chris ; KONG, Henry ; KOPETZKI, Sven ; SIRIPURAPU, Ramesh ; SHEN, Junqiang: V2X Communication Technology: Field Experience and Comparative Analysis. Version: Oktober 2012. http://dx.doi.org/10.1007/978-3-642-33838-0_10. In: *Lecture Notes in Electrical Engineering*. Springer Berlin Heidelberg, Oktober 2012. – DOI 10.1007/978–3–642–33838–0_10, 113–129

[Rae19] RAESE, Nina ; INGENIEUR.DE (Hrsg.): *Ein Assistent,*

der Radfahrer und Fußgänger schützt. https://www.ingenieur.de/technik/fachbereiche/fahrzeugbau/ein-assistent-der-radfahrer-und-fussgaenger-schuetzt/. Version: 2019

[RAY97] RAY, LAURA R.: Nonlinear Tire Force Estimation and Road Friction Identification: Simulation and Experiments1,2. In: *Automatica* 33 (1997), Nr. 10, 1819 - 1833. http://dx.doi.org/10.1016/S0005-1098(97)00093-9. – DOI 10.1016/S0005–1098(97)00093–9. – ISSN 0005–1098

[RBL+09] RAUSKOLB, Fred W. ; BERGER, Kai ; LIPSKI, Christian ; MAGNOR, Marcus ; CORNELSEN, Karsten ; EFFERTZ, Jan ; FORM, Thomas ; GRAEFE, Fabian ; OHL, Sebastian ; SCHUMACHER, Walter ; WILLE, Jörn-Marten ; HECKER, Peter ; NOTHDURFT, Tobias ; DOERING, Michael ; HOMEIER, Kai ; MORGENROTH, Johannes ; WOLF, Lars ; BASARKE, Christian ; BERGER, Christian ; GÜLKE, Tim ; KLOSE, Felix ; RUMPE, Bernhard: Caroline: An Autonomously Driving Vehicle for Urban Environments. Version: 2009. http://dx.doi.org/10.1007/978-3-642-03991-1_11. In: *Springer Tracts in Advanced Robotics*. Springer Berlin Heidelberg, 2009. – DOI 10.1007/978–3–642–03991–1_11, 441–508

[RH08] RUNESON, Per ; HÖST, Martin: Guidelines for conducting and reporting case study research in software engineering. In: *Empirical Software Engineering* 14 (2008), Dezember, Nr. 2, 131–164. http://dx.doi.org/10.1007/s10664-008-9102-8. – DOI 10.1007/s10664–008–9102–8. – ISSN 1382–3256

[RHZE15] RAUDSZUS, Dominik ; HAMACHER, Michael ; ZLOCKI, Adrian ; ECKSTEIN, Lutz: Integrated Assessment of Active Pedestrian Protection Systems. In: *ATZ worldwide* 117 (2015), Januar, Nr. 1, 4–9. http://dx.doi.org/10.1007/s38311-015-0145-3. – DOI 10.1007/s38311–015–0145–3

[RRW14] RINGERT, Jan O. ; RUMPE, Bernhard ; WORTMANN, Andreas: *Architecture and Behavior Modeling of Cyber-Physical Systems with MontiArcAutomaton*. Shaker Verlag, 2014 (Aachener Informatik-Berichte, Software Engineering, Band 20). http://www.se-rwth.de/publications/

[RSMT14] RÖDEL, Christina ; STADLER, Susanne ; MESCHTSCHERJAKOV, Alexander ; TSCHELIGI, Manfred: Towards Autonomous Cars: The Effect of Autonomy Levels on Acceptance and User Experience. In: *Proceedings of the 6th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*.

New York, NY, USA : ACM, 2014 (AutomotiveUI '14). – ISBN 978–1–4503–3212–5, 11:1–11:8

[RSSW10] RUMPE, Bernhard ; SCHINDLER, Martin ; STEVEN, Völkel; ; WEISEMÜLLER, Ingo: Generative Software Developement. In: *Proceedings of the 32nd International Conference on Software Engineering* Bd. 2, 2010, S. 473–474

[Rum04] RUMPE, Bernhard: *Modellierung mit UML - Sprache, Konzepte und Methodik*. Springer Verlag, 2004

[Rum05] RUMPE, Bernhard: *Agile Modellierung mit UML - Codegenerierung, Testfälle, Refactoring*. Springer Verlag, 2005

[Rum11] RUMPE, Bernhard: *Modellierung mit UML*. 2nd Edition. Springer Berlin, 2011 (Xpert.press)

[Rum12] RUMPE, Bernhard: *Agile Modellierung mit UML*. Springer Berlin Heidelberg, 2012. http://dx.doi.org/10.1007/978-3-642-22430-0. http://dx.doi.org/10.1007/978-3-642-22430-0

[Rum16] RUMPE, Bernhard: *Modeling with UML: Language, Concepts, Methods*. Springer International, 2016 http://www.se-rwth.de/mbse/

[Rum17] RUMPE, Bernhard: *Agile Modeling with UML: Code Generation, Testing, Refactoring*. Springer International, 2017 http://www.se-rwth.de/mbse/

[SA07] SATO, Toshihisa ; AKAMATSU, Motoyuki: Influence of traffic conditions on driver behavior before making a right turn at an intersection: Analysis of driver behavior based on measured data on an actual road. In: *Transportation Research Part F: Traffic Psychology and Behaviour* 10 (2007), Nr. 5, 397 - 413. http://dx.doi.org/http://dx.doi.org/10.1016/j.trf.2007.03.001. – DOI http://dx.doi.org/10.1016/j.trf.2007.03.001. – ISSN 1369–8478

[Saz14] SAZAMA, Frank: *Agile in Automotive - State of the Practice*. https://fdocuments.in/document/agile-in-automotive-state-of-the-practice-2014.html. Version: 2014

[SBBM07] SCHICK, Bernhard ; BÜTTNER, Rolf ; BALTRUSCHAT, Klaus ; MEIER, Günther: Bewertung der Funktion und Güte von Fahrerassistenzsystemen bei aktivem Brem-

seingriff. In: *ATZ* 05 (2007), S. 414–425. http://dx.doi.org/10.1007/BF03221887. – DOI 10.1007/BF03221887

[SBHK14] SCHMIDT, Kim ; BEGGIATO, Matthias ; HOFFMANN, Karl H. ; KREMS, Josef F.: A mathematical model for predicting lane changes using the steering wheel angle. In: *Journal of Safety Research* 49 (2014), Nr. 0, 85.e1 - 90. http://dx.doi.org/http://dx.doi.org/10.1016/j.jsr.2014.02.014. – DOI http://dx.doi.org/10.1016/j.jsr.2014.02.014. – ISSN 0022–4375. – Proceedings of the International Conference on Road Safety (RSS2013) International Conference on Road Safety

[SC11] SEGATA, Michele ; CIGNO, Renato L.: Emergency braking. In: *Proceedings of the Eighth ACM international workshop on Vehicular inter-networking - VANET '11*, ACM Press, 2011

[SC13] SEGATA, Michele ; CIGNO, Renato L.: Automatic Emergency Braking: Realistic Analysis of Car Dynamics and Network Performance. In: *IEEE Transactions on Vehicular Technology* 62 (2013), November, Nr. 9, 4150–4161. http://dx.doi.org/10.1109/tvt.2013.2277802. – DOI 10.1109/tvt.2013.2277802

[Sch88] SCHUMANN, Gerisch: *Software-Entwurf*. Rudolf Müller, 1988

[Sch06] *Kapitel* 1. In: SCHLINGLOFF, Bernd-Holger: *Softwarequalität - Geschichte und Trends*. Springer, Berlin, Heidelberg, 2006. – ISBN 978–3–540–32742–4, S. 329 – 345

[Sch12] SCHINDLER, Martin ; RUMPE, Bernhard (Hrsg.): *Eine Werkzeuginfrastruktur zur agilen EEntwicklung mit der UML/P*. Bd. 11. Shaker Verlag, 2012 www.se-rwth.de/publications. – ISBN 978–3–8440–0864–7

[SFF14] SCHNEIDER, Stefan-Alexander ; FRIMBERGER, Johannes ; FOLIE, Michael: Reduced Validation Effort for Dynamic Light Functions. In: *ATZelektronik worldwide* 9 (2014), März, Nr. 2, 16–21. http://dx.doi.org/10.1365/s38314-014-0233-y. – DOI 10.1365/s38314–014–0233–y

[SHWK15] SCHMIDT, Steffen ; HENNING, Josef ; WAMBERA, Thomas ; KRONIMUS, Silke: Early PC-based Validation of ECU Software Using Virtual Test Driving. In: *ATZelektronik worldwide* 10 (2015), Februar, Nr. 1, 14–17. http://dx.doi.org/10.1007/s38314-015-0508-y. – DOI 10.1007/s38314–015–0508–y

[Sie19] SIEMERS, Sebastian Christian; G. Christian; Gerstl: *Was ist ein Embedded System?* https://www.embedded-software-engineering.de/was-ist-ein-embedded-system-a-665424/. Version: 2019

[SL07] SPILLNER, Andreas ; LINZ, Tilo: *Basiswissen Softwaretest - Aus- und Weiterbildung zum Certified Tester*. dpunkt.verlag, 2007

[SLL12a] SCHICK, Bernhard ; LEONHARD, Volker ; LANGE, Steffen: Predictive energy management in virtual driving tests. In: *ATZ worldwide* 114 (2012), April, Nr. 4, 28–32. http://dx.doi.org/10.1007/s38311-012-0166-0. – DOI 10.1007/s38311–012–0166–0

[SLL12b] SCHICK, Bernhard ; LEONHARD, Volker ; LANGE, Steffen: Predictive Energy Management in Virtual Driving Tests. In: *ATZautotechnology* 12 (2012), April, Nr. 2, 48–53. http://dx.doi.org/10.1365/s35595-012-0108-x. – DOI 10.1365/s35595–012–0108–x

[SLZB14] SCHWAB, Sebastian ; LEICHSENRING, Tobias ; ZOFKA, Marc R. ; BÄR, Tobias: Consistent Test Method for Assistance Systems. In: *ATZ worldwide* 116 (2014), August, Nr. 9, 38–43. http://dx.doi.org/10.1007/s38311-014-0216-x. – DOI 10.1007/s38311–014–0216–x

[Som01] SOMMERVILLE, Ian: *Software Engineering*. Peason Studium, 2001. – ISBN 3827370019

[SPM13] STAUDER, Steffen ; PLÖGER, Markus ; MÜLLER, Steffen: Model-Based Controller and Function Development for Mechatronic Steering Systems. In: *ATZ worldwide* 115 (2013), April, Nr. 5, 36–40. http://dx.doi.org/10.1007/s38311-013-0060-4. – DOI 10.1007/s38311–013–0060–4

[SR10] SARSHAR, Mohammadreza ; REZAEI, Mahdi: A novel system for Advanced Driver Assistance Systems. In: *2010 IEEE International Systems Conference*, IEEE, April 2010

[SS12] SCHICK, Bernhard ; SCHMIDT, Steffen: Evaluation of Video-Based Driver Assistance Systems with Sensor Data Fusion by Using Virtual Test Driving. Version: November 2012. http://dx.doi.org/10.1007/978-3-642-33738-3_36. In: *Lecture Notes in Electrical Engineering*. Springer Berlin Heidelberg, November 2012. – DOI 10.1007/978–3–642–33738–3_36, 1363–1375

[SSB10] SNEED, Harry M. ; SEIDL, Richard ; BAUMGARTNER, Manfred: *Software in Zahlen - Die Vermessung von Applikationen*. Hanser-Verlag, 2010

[ST13] SIVARAMAN, Sayanan ; TRIVEDI, Mohan M.: Towards cooperative, predictive driver assistance. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, IEEE, Oktober 2013

[SWB+97] STAPLEFORD, R.L. ; WEIR, D.H. ; BROEN, N.L. ; CHIANG, D.P. ; IGARASHI, R.: The DRI Driving Simulator 1997, description and applications to the study of intelligent transportation systems. In: *Proceedings of Conference on Intelligent Transportation Systems*, IEEE, 1997

[SWR13] SCHRAM, Richard ; WILLIAMS, Aled ; RATINGEN, Michiel van: Implementation of Autonomous Emergency Braking (AEB), the Next Step in Euro NCAP's Safety Assessment. In: *EuroNCAP Webseite*, 2013

[SX09] STEFANI, A. ; XENOS, M.: Meta-Metric Evaluation of E-Commerce-related Metrics. In: *Electronic Notes in Theoretical Computer Science* 233 (2009), mar, S. 59–72. http://dx.doi.org/10.1016/j.entcs.2009.02.061. – DOI 10.1016/j.entcs.2009.02.061

[SZ10] SCHAEUFELE, Joerg ; ZURAWKA, Thomas: *Automotive Software Engineering: Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen*. ATZ/MTZ-Fabuvh. Vieweg+Teubner Verlag, 2010. – ISBN 978–3834803641

[TBL+12] TIDEMAN, Martijn ; BOURS, Roy ; LI, Hao ; SCHULZE, Tino ; NAKANO, Takahito: Integrated Simulation Toolset for ADA System Development. Version: Oktober 2012. http://dx.doi.org/10.1007/978-3-642-33838-0_3. In: *Lecture Notes in Electrical Engineering*. Springer Berlin Heidelberg, Oktober 2012. – DOI 10.1007/978–3–642–33838–0_3, 25–36

[TFT+12] TANG, Chi-Wai ; FENG, Kai-Ten ; TSENG, Po-Hsuan ; CHEN, Chien-Hua ; GUO, Jing-Wei: A pitch-aided lane tracking algorithm for driver assistance system with insufficient observations. In: *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, IEEE, April 2012

[TGW+05] TORKKOLA, K. ; GARDNER, M. ; WOOD, C. ; SCHREINER, C. ; MASSEY, N. ; LEIVIAN, B. ; SUMMERS, J. ; VENKATESAN, S.: Toward modeling and classification of naturalistic driving. In: *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, IEEE, 2005, -

[TH13] TAN, Yin ; HASSAN, Bassem: A method for testing camera based advanced driving assistance systems. In: *2013 IEEE International Symposium on Assembly and Manufacturing (ISAM)*, IEEE, Juli 2013

[Tid10] TIDEMAN, M.: Scenario-Based Simulation Environment for Assistance Systems. 10 (2010). http://dx.doi.org/10.1007/BF03247153. – DOI 10.1007/BF03247153. – ISSN 2192–886X

[TRPP04] TRICOT, N. ; RAJAONAH, B. ; PACAUX, M. ; POPIEUL, J.-C.: Driver's behaviors and human-machine interactions characterization for the design of an advanced driving assistance system. In: *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, IEEE, 2004

[TSGZ06] TORKKOLA, K. ; SCHREINER, C. ; GARDNER, M. ; ZHANG, Keshu: Development of a Semi-Automatic Data Annotation Tool for Driving Data. In: *2006 IEEE Intelligent Transportation Systems Conference*, IEEE, 2006, -

[TVA10] TIDEMAN, Martijn ; VOORT, Mascha C. d. ; AREM, Bart van: A new scenario based approach for designing driver support systems applied to the design of a lane change support system. In: *Transportation Research Part C: Emerging Technologies* 18 (2010), Nr. 2, 247 - 258. http://dx.doi.org/http://dx.doi.org/10.1016/j.trc.2009.09.001. – DOI http://dx.doi.org/10.1016/j.trc.2009.09.001. – ISSN 0968–090X

[unk] UNKNOWN: Introduction. Version: -. http://dx.doi.org/10.1007/978-0-387-44409-3_1. In: *Advanced Motion Control and Sensing for Intelligent Vehicles*. Springer US, -. – DOI 10.1007/978–0–387–44409–3_1, 1–31

[VAK10] VANKAYALAPATI, H. D. ; ANNE, K. R. ; KYAMAKYA, K.: Extraction of Visual and Acoustic Features of the Driver for Monitoring Driver Ergonomics Applied to Extended Driver Assistance Systems. Version: 2010. http://dx.doi.org/10.1007/978-3-642-15503-1_8. In: *Advances in Intelligent and Soft Computing*. Springer Berlin Heidelberg, 2010. – DOI 10.1007/978–3–642–15503–1_8, 83–94

[Voe11] VOELKEL, Steven ; RUMPE, Bernhard (Hrsg.): *Kompositionale Entwicklung domänenspezifischer Sprachen*. Bd. 9. Shaker Verlag, 2011 www.se-rwth.de/publications. – ISBN 978–3–8440–0328–4

[Vol19a] VOLKSWAGEN, AG: *Fahrprofilauswahl*. https://www.volkswagen.de/de/technologie/technik-lexikon/f-j.html. Version: 2019

[Vol19b] VOLKSWAGEN, AG: *So funktioniert das Proaktive Insassenschutzsystem*. https://www.volkswagen.de/de/technologie/assistenzsysteme/gut-ankommen.html#item=1&powerLayer=insassenschutzsystem.display. Version: 2019

[Vol19c] VOLKSWAGEN, AG: *So funktioniert das Umfeldbeobachtungsystem Front Assist mit City-Notbremsfunktion*. https://www.volkswagen.de/de/technologie/assistenzsysteme/gut-ankommen.html#item=1&powerLayer=t.display. Version: 2019

[Vol19d] VOLVO: *Radfahrererkennung*. https://www.volvocars.com/de/support/manuals/s60/2016-late/fahrerunterstutzung/unfallwarnsystem/unfallwarnsystem---radfahrererkennung. Version: 2019

[WN10] WEBER, Edzard ; NIMMICH, Andre: Meta-Kennzahlen fuer die Bewertung von Dienstleistungen. In: *Diskussionsbeitraege des 2. Workshops Dienstleistungsmodellierung (DLM 2010)*, 2010, S. 65–88

[Woo99] WOODINGS, Terry L.: *Meta-Metrics for the Accuracy of Software Project Estimation*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.118.2390, 1999

[WPH+14] WU, Chaozhong ; PENG, Liqun ; HUANG, Zhen ; ZHONG, Ming ; CHU, Duanfeng: A method of vehicle motion prediction and collision risk assessment with a simulated vehicular cyber physical system. In: *Transportation Research Part C: Emerging Technologies* 47, Part 2 (2014), Nr. 0, 179 - 191. http://dx.doi.org/10.1016/j.trc.2014.07.002. – DOI 10.1016/j.trc.2014.07.002. – ISSN 0968–090X. – Special Issue: Emerging Technologies Special Issue of {ICTIS} 2013 – Guest Editors: Liping Fu and Ming Zhong and Special Issue: Visualization & Visual Analytics in Transportation – Guest Editors: Patricia S. Hu and Michael L. Pack

[WR06] WALLENTOWITZ, Henning ; REIF, Konrad: *Handbuch Kraftfahrzeugelektronik*. Bd. 1. Vieweg+Teubner Verlag, 2006. http://dx.doi.org/10.1007/978-3-8348-9121-1. http://dx.doi.org/10.1007/978-3-8348-9121-1. – ISBN 978–3–8348–9121–1

[WR10] WALLENTOWITZ, H. ; REIF, K.: *Handbuch Kraftfahrzeugelektronik: Grundlagen - Komponenten - Systeme - Anwendungen*. Vieweg+Teubner Verlag,

2010 (ATZ/MTZ-Fachbuch). https://books.google.de/books?id=yMjcMIdGceoC. – ISBN 9783834807007

[WRH+12] WOHLIN, Claes ; RUNESON, Per ; HÖST, Martin ; OHLSSON, Magnus C. ; REGNELL, Björn ; WESSLÉN, Anders: *Experimentation in Software Engineering*. Springer Berlin Heidelberg, 2012. http://dx.doi.org/10.1007/978-3-642-29044-2. http://dx.doi.org/10.1007/978-3-642-29044-2

[WRL05] WOLF, Henning ; ROOCK, Stefan ; LIPPERT, Martin: *eXtreme Programming*. 2. erweiterte und überarbeitete Auflage. dpunkt.verlag, 2005. – ISBN 978–3898643399

[WWW10] WINNER, H. ; WOLF, G. ; WEITZEL, A.: Freigabefalle des Autonomen Fahrens. In: *VDI-Berichte* 2107 (2010), S. 17–29

[XMKS07] XU, Qing ; MAK, T. ; KO, Jeff ; SENGUPTA, R.: Medium Access Control Protocol Design for Vehicle–Vehicle Safety Messages. In: *IEEE Transactions on Vehicular Technology* 56 (2007), Nr. 2, 499–518. http://dx.doi.org/10.1109/tvt.2007.891482. – DOI 10.1109/tvt.2007.891482

[XSZC00] XENOS, M. ; STAVRINOUDIS, D. ; ZIKOULI, K. ; CHRISTODOULAKIS, D.: Object-Oriented Metrics - A Survey. In: *Proceedings of the FESMA Conference (FESMA'2000)*, 2000

[YS15] YU, Shaowei ; SHI, Zhongke: Dynamics of connected cruise control systems considering velocity changes with memory feedback. In: *Measurement* 64 (2015), Nr. 0, 34 - 48. http://dx.doi.org/10.1016/j.measurement.2014.12.036. – DOI 10.1016/j.measurement.2014.12.036. – ISSN 0263–2241

[YSLS14] YOUNG, William ; SOBHANI, Amir ; LENNï¿½, Michael G. ; SARVI, Majid: Simulation of safety: A review of the state of the art in road safety simulation modelling. In: *Accident Analysis & Prevention* 66 (2014), Nr. 0, 89 - 103. http://dx.doi.org/10.1016/j.aap.2014.01.008. – DOI 10.1016/j.aap.2014.01.008. – ISSN 0001–4575

[YYW+10] YAN, Gongjun ; YANG, Weiming ; WEIGLE, Michele C. ; OLARIU, Stephan ; RAWAT, Danda: Cooperative Collision Warning through mobility and probability prediction. In: *2010 IEEE Intelligent Vehicles Symposium*, IEEE, Juni 2010, -

[ZAC+09] ZELLNER, John W. ; AUKEN, R. Michael V. ; CHIANG, Dean P. ; BROEN, Peter C.

; KELLY, Joseph ; SUGIMOTO, Yoichi: Extension of the Honda-DRI "Safety Impact Methodology" (SIM) for the NHTSA Advanced Crash Avoidance Technology (ACAT) Program and Application to a Prototype Advanced Collision Mitigation Braking System. In: *SAE International Journal of Passenger Cars - Mechanical Systems* 2 (2009), April, Nr. 1, 875–894. http://dx.doi.org/10.4271/2009-01-0781. – DOI 10.4271/2009–01–0781

[ZAS+12] ZELLNER, John ; AUKEN, R. Michael V. ; SILBERLING, Jordan ; KELLY, Joseph ; HAGOSKI, Brad ; SUGIMOTO, Yoichi ; URAI, Yoshihiro: Extension of the Honda-DRI Safety Impact Methodology (SIM) for the (NHTSA) Advanced Crash Avoidance Technology (ACAT) II Program and Application to the Evaluation of a Pre-Production Head-On Crash Avoidance Assist System - Progress Report. In: *SAE International Journal of Passenger Cars - Electronic and Electrical Systems* 5 (2012), April, Nr. 2, 385–402. http://dx.doi.org/10.4271/2012-01-0291. – DOI 10.4271/2012–01–0291

[ZCLL14] ZHANG, Qiang ; CHEN, Daxing ; LI, Yusheng ; LI, Keqiang: Research on Performance Test Method of Lane Departure Warning System with PreScan. Version: Dezember 2014. http://dx.doi.org/10.1007/978-3-662-45043-7_45. In: *Lecture Notes in Electrical Engineering*. Springer Berlin Heidelberg, Dezember 2014. – DOI 10.1007/978–3–662–45043–7_45, 445–453

[ZFB+14] ZLOCKI, Adrian ; FAHRENKROG, Felix ; BENMIMOUN, Mohamed ; JOSTEN, Johanna ; ECKSTEIN, Lutz: Evaluation of Automated Road Vehicles. Version: 2014. http://dx.doi.org/10.1007/978-3-319-05990-7_17. In: *Road Vehicle Automation*. Springer International Publishing, 2014. – DOI 10.1007/978–3–319–05990–7_17, 197–208

[ZS06] ZIMMERMANN, Werner ; SCHMIDGALL, Ralf: *Bussysteme in der Fahrzeugtechnik*. Bd. 3. 2006

[ZTWL13] ZINCHENKO, Tetiana ; TCHOUANKEM, Hugues ; WOLF, Lars ; LESCHKE, André: Reliability analysis of vehicle-to-vehicle applications based on real world measurements. In: *Proceeding of the tenth ACM international workshop on Vehicular inter-networking, systems, and applications - VANET 13*, (ACM) Press, 2013, -

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ABS	Anti-Blockier-System
ACC	Adaptive Cruise Control
AEB	Automated Emergency Breaking
AEB	Autonomous Emergency Braking
ASR	Anti-Schlupf-Regelung
AUTOSAR	AUTomotive Open System ARchitecture
CCRb	Car-to-Car Rear: braking
CCRm	Car-to-Car Rear: moving
CCRs	Car-to-Car Rear: stationary
CMS	Collision Mitigation System
CPS	Cyber-Physical-System
CTO	Consumer Test Organization
EBNF	Extended-Backus-Naur-Form
ESC	Electronic Stability Control
ESP	Elektronisches Stablitäts-programm
EuroNCAP	European New Car Assessment Programme
EVT	EuroNCAP-Vehicle-Target
FCW	Forward Collision Warning
HBA	Hydraulischer Brems-Assistent
HiL	Hardware-in-the-Loop
ISM	IntegrationsStufenManagement
ISTQB	International Software Testing Qualifications Board
LKA	Lane Keeping Assist
MDA	Model Driven Architecture
MiL	Model-in-the-Loop
PiL	Processor-in-the-Loop

RQ ..... Research Question
RUP ..... Rational Unified Process
SiL ..... Software-in-the-Loop
SimAQuAss4ConTest . Simulative Agile Quality Assurance for Consumer Tests
SLR ..... Systematic Literature Review
SLR ..... Systematic Literature Review
SOP ..... Start of Production
TTC ..... Time-To-Collision
UML ..... Unified Modeling Language
USNCAP ..... United States New Car Assessment Programme
VUT ..... Vehicle-Under-Test

Related Interesting Work from the SE Group, RWTH Aachen

The following section gives an overview on related work done at the SE Group, RWTH Aachen. More details can be found on the website `https://www.se-rwth.de/topics/` or in [HMR+19]. The work presented here mainly has been guided by our mission statement:

Our mission is to define, improve, and industrially apply *techniques*, *concepts*, and *methods* for *innovative and efficient development* of software and software-intensive systems, such that *high-quality* products can be developed in a *shorter period of time* and with *flexible integration* of *changing requirements*. Furthermore, we *demonstrate the applicability* of our results in various domains and potentially refine these results in a domain specific form.

Agile Model Based Software Engineering

Agility and modeling in the same project? This question was raised in [Rum04]: "Using an executable, yet abstract and multi-view modeling language for modeling, designing and programming still allows to use an agile development process.", [JWCR18] addresses the question how digital and organizational techniques help to cope with physical distance of developers and [RRSW17] addresses how to teach agile modeling. Modeling will increasingly be used in development projects, if the benefits become evident early, e.g with executable UML [Rum02] and tests [Rum03]. In [GKRS06], for example, we concentrate on the integration of models and ordinary programming code. In [Rum12] and [Rum16], the UML/P, a variant of the UML especially designed for programming, refactoring and evolution, is defined. The language workbench MontiCore [GKR+06, GKR+08, HR17] is used to realize the UML/P [Sch12]. Links to further research, e.g., include a general discussion of how to manage and evolve models [LRSS10], a precise definition for model composition as well as model languages [HKR+09] and refactoring in various modeling and programming languages [PR03]. In [FHR08] we describe a set of general requirements for model quality. Finally, [KRV06] discusses the additional roles and activities necessary in a DSL-based software development project. In [CEG+14] we discuss how to improve the reliability of adaptivity through models at runtime, which will allow developers to delay design decisions to runtime adaptation.

Artifacts in Complex Development Projects

Developing modern software solutions has become an increasingly complex and time consuming process. Managing the complexity, size, and number of the artifacts developed and used during a project together with their complex relationships is not trivial [BGRW17]. To keep track of relevant structures, artifacts, and their relations in order to be able e.g. to evolve or adapt models and their implementing code, the *artifact model* [GHR17] was introduced. [BGRW18] explains its applicability in systems engineering based on MDSE projects.

An artifact model basically is a meta-data structure that explains which kinds of artifacts, namely code files, models, requirements files, etc. exist and how these artifacts are related to each other. The artifact model therefore covers the wide range of human activities during the development down to fully automated, repeatable build scripts. The artifact model can be used to optimize parallelization during the development and building, but also to identify deviations of the real architecture and dependencies from the desired, idealistic architecture, for cost estimations, for requirements and bug tracing, etc. Results can be measured using metrics or visualized as graphs.

Artificial Intelligence in Software Engineering

MontiAnna is a family of explicit domain specific languages for the concise description of the architecture of (1) a neural network, (2) its training, and (3) the training data [KNP+19]. We have developed a compositional technique to integrate neural networks into larger software architectures [KRRvW17] as standardized machine learning components [KPRS19]. This enables the compiler to support the systems engineer by automating the lifecycle of such components including multiple learning approaches such as supervised learning, reinforcement learning, or generative adversarial networks. According to [MRR11g] the semantic difference between two models are the elements contained in the semantics of the one model that are not elements in the semantics of the other model. A smart semantic differencing operator is an automatic procedure for computing diff witnesses for two given models. Smart semantic differencing operators have been defined for Activity Diagrams [MRR11a], Class Diagrams [MRR11d], Feature Models [DKMR19], Statecharts [DEKR19], and Message-Driven Component and Connector Architectures [BKRW17, BKRW19]. We also developed a modeling language-independent method for determining syntactic changes that are responsible for the existence of semantic differences [KR18].

We apply logic, knowledge representation and intelligent reasoning to software engineering to perform correctness proofs, execute symbolic tests or find counterexamples using a theorem prover. And we have applied it to challenges in intelligent flight control systems and assistance systems for air or road traffic management [KRRS19, HRR12] and based it on the core ideas of Broy's Focus theory [RR11, BR07]. Intelligent testing strategies have been applied to automotive software engineering [EJK+19, DGH+19, KMS+18], or more generally in systems engineering [DGH+18]. These methods are realized for a variant of SysML Activity Diagrams and Statecharts.

Machine Learning has been applied to the massive amount of observable data in energy management for buildings [FLP+11a, KLPR12] and city quarters [GLPR15] to optimize the operation efficiency and prevent unneeded CO2 emissions or reduce costs. This creates a structural and behavioral system theoretical view on cyber-physical systems understandable as essential parts of digital twins [RW18, BDH+20].

Generative Software Engineering

The UML/P language family [Rum12, Rum11, Rum16] is a simplified and semantically sound derivate of the UML designed for product and test code generation. [Sch12] describes a flexible generator

for the UML/P based on the MontiCore language workbench [KRV10, GKR+06, GKR+08, HR17]. In [KRV06], we discuss additional roles necessary in a model-based software development project. [GKRS06, GHK+15a] discuss mechanisms to keep generated and handwritten code separated. In [Wei12], we demonstrate how to systematically derive a transformation language in concrete syntax. [HMSNRW16] presents how to generate extensible and statically type-safe visitors. In [MSNRR16], we propose the use of symbols for ensuring the validity of generated source code. [GMR+16] discusses product lines of template-based code generators. We also developed an approach for engineering reusable language components [HLMSN+15b, HLMSN+15a]. To understand the implications of executability for UML, we discuss needs and advantages of executable modeling with UML in agile projects in [Rum04], how to apply UML for testing in [Rum03], and the advantages and perils of using modeling languages for programming in [Rum02].

Unified Modeling Language (UML)

Starting with an early identification of challenges for the standardization of the UML in [KER99] many of our contributions build on the UML/P variant, which is described in the books [Rum16, Rum17] respectively [Rum12, Rum13] and is implemented in [Sch12]. Semantic variation points of the UML are discussed in [GR11]. We discuss formal semantics for UML [BHP+98] and describe UML semantics using the "System Model" [BCGR09a], [BCGR09b], [BCR07b] and [BCR07a]. Semantic variation points have, e.g., been applied to define class diagram semantics [CGR08]. A precisely defined semantics for variations is applied, when checking variants of class diagrams [MRR11c] and objects diagrams [MRR11e] or the consistency of both kinds of diagrams [MRR11f]. We also apply these concepts to activity diagrams [MRR11b] which allows us to check for semantic differences of activity diagrams [MRR11a]. The basic semantics for ADs and their semantic variation points is given in [GRR10]. We also discuss how to ensure and identify model quality [FHR08], how models, views and the system under development correlate to each other [BGH+98], and how to use modeling in agile development projects [Rum04], [Rum02]. The question how to adapt and extend the UML is discussed in [PFR02] describing product line annotations for UML and more general discussions and insights on how to use meta-modeling for defining and adapting the UML are included in [EFLR99], [FELR98] and [SRVK10].

Domain Specific Languages (DSLs)

Computer science is about languages. Domain Specific Languages (DSLs) are better to use, but need appropriate tooling. The MontiCore language workbench [GKR+06, KRV10, Kra10, GKR+08, HR17] allows the specification of an integrated abstract and concrete syntax format [KRV07b, HR17] for easy development. New languages and tools can be defined in modular forms [KRV08, GKR+07, Völ11, HLMSN+15b, HLMSN+15a, HRW18, BEK+18a, BEK+18b, BEK+19] and can, thus, easily be reused. We discuss the roles in software development using domain specific languages in [KRV14]. [Wei12] presents a tool that allows to create transformation rules tailored to an underlying DSL. Variability in DSL definitions has been examined in [GR11, GMR+16]. [BDL+18] presents a method to derive internal

DSLs from grammars. In [BJRW18], we discuss the translation from grammars to accurate metamodels. Successful applications have been carried out in the Air Traffic Management [ZPK+11] and television [DHH+20] domains. Based on the concepts described above, meta modeling, model analyses and model evolution have been discussed in [LRSS10] and [SRVK10]. DSL quality [FHR08], instructions for defining views [GHK+07], guidelines to define DSLs [KKP+09] and Eclipse-based tooling for DSLs [KRV07a] complete the collection.

Software Language Engineering

For a systematic definition of languages using composition of reusable and adaptable language components, we adopt an engineering viewpoint on these techniques. General ideas on how to engineer a language can be found in the GeMoC initiative [CBCR15, CCF+15] and the concern-oriented language development approach [CKM+18]. As said, the MontiCore language workbench provides techniques for an integrated definition of languages [KRV07b, Kra10, KRV10, HR17, HRW18, BEK+19]. In [SRVK10] we discuss the possibilities and the challenges using metamodels for language definition. Modular composition, however, is a core concept to reuse language components like in MontiCore for the frontend [Völ11, KRV08, HLMSN+15b, HLMSN+15a, HMSNRW16, HR17, BEK+18a, BEK+18b, BEK+19] and the backend [RRRW15, MSNRR16, GMR+16, HR17, BEK+18b]. In [GHK+15b, GHK+15a], we discuss the integration of handwritten and generated object-oriented code. [KRV14] describes the roles in software development using domain specific languages. Language derivation is to our believe a promising technique to develop new languages for a specific purpose that rely on existing basic languages [HRW18]. How to automatically derive such a transformation language using concrete syntax of the base language is described in [HRW15, Wei12] and successfully applied to various DSLs. We also applied the language derivation technique to tagging languages that decorate a base language [GLRR15] and delta languages [HHK+15a, HHK+13], where a delta language is derived from a base language to be able to constructively describe differences between model variants usable to build feature sets. The derivation of internal DSLs from grammars is discussd in [BDL+18] and a translation of grammars to accurate metamodels in [BJRW18].

Modeling Software Architecture & the MontiArc Tool

Distributed interactive systems communicate via messages on a bus, discrete event signals, streams of telephone or video data, method invocation, or data structures passed between software services. We use streams, statemachines and components [BR07] as well as expressive forms of composition and refinement [PR99, RW18] for semantics. Furthermore, we built a concrete tooling infrastructure called MontiArc [HRR12] for architecture design and extensions for states [RRW13b]. In [RRW13a], we introduce a code generation framework for MontiArc. MontiArc was extended to describe variability [HRR+11] using deltas [HRRS11, HKR+11] and evolution on deltas [HRRS12]. Other extensions are concerned with modeling cloud architectures [NPR13] and with the robotics domain [AHRW17a, AHRW17b]. [GHK+07] and [GHK+08a] close the gap between the requirements and the logical architecture and

[GKPR08] extends it to model variants. [MRR14b] provides a precise technique to verify consistency of architectural views [Rin14, MRR13] against a complete architecture in order to increase reusability. We discuss the synthesis problem for these views in [MRR14a]. Co-evolution of architecture is discussed in [MMR10] and modeling techniques to describe dynamic architectures are shown in [HRR98, BHK+17, KKR19].

Compositionality & Modularity of Models

[HKR+09] motivates the basic mechanisms for modularity and compositionality for modeling. The mechanisms for distributed systems are shown in [BR07, RW18] and algebraically underpinned in [HKR+07]. Semantic and methodical aspects of model composition [KRV08] led to the language workbench MontiCore [KRV10, HR17] that can even be used to develop modeling tools in a compositional form [HR17, HLMSN+15b, HLMSN+15a, HMSNRW16, MSNRR16, HRW18, BEK+18a, BEK+18b, BEK+19]. A set of DSL design guidelines incorporates reuse through this form of composition [KKP+09]. [Völ11] examines the composition of context conditions respectively the underlying infrastructure of the symbol table. Modular editor generation is discussed in [KRV07a]. [RRRW15] applies compositionality to Robotics control. [CBCR15] (published in [CCF+15]) summarizes our approach to composition and remaining challenges in form of a conceptual model of the "globalized" use of DSLs. As a new form of decomposition of model information we have developed the concept of tagging languages in [GLRR15]. It allows to describe additional information for model elements in separated documents, facilitates reuse, and allows to type tags.

Semantics of Modeling Languages

The meaning of semantics and its principles like underspecification, language precision and detailedness is discussed in [HR04]. We defined a semantic domain called "System Model" by using mathematical theory in [RKB95, BHP+98] and [GKR96, KRB96]. An extended version especially suited for the UML is given in [BCGR09b] and in [BCGR09a] its rationale is discussed. [BCR07a, BCR07b] contain detailed versions that are applied to class diagrams in [CGR08]. To better understand the effect of an evolved design, detection of semantic differencing as opposed to pure syntactical differences is needed [MRR10]. [MRR11a, MRR11b] encode a part of the semantics to handle semantic differences of activity diagrams and [MRR11f, MRR11f] compare class and object diagrams with regard to their semantics. In [BR07], a simplified mathematical model for distributed systems based on black-box behaviors of components is defined. Meta-modeling semantics is discussed in [EFLR99]. [BGH+97] discusses potential modeling languages for the description of an exemplary object interaction, today called sequence diagram. [BGH+98] discusses the relationships between a system, a view and a complete model in the context of the UML. [GR11] and [CGR09] discuss general requirements for a framework to describe semantic and syntactic variations of a modeling language. We apply these on class and object diagrams in [MRR11f] as well as activity diagrams in [GRR10]. [Rum12] defines the semantics in a variety of code and test case generation, refactoring and evolution techniques. [LRSS10] discusses evolution and related issues in

greater detail. [RW18] discusses an elaborated theory for the modeling of underspecification, hierarchical composition, and refinement that can be practically applied for the development of CPS.

Evolution and Transformation of Models

Models are the central artifacts in model driven development, but as code they are not initially correct and need to be changed, evolved and maintained over time. Model transformation is therefore essential to effectively deal with models. Many concrete model transformation problems are discussed: evolution [LRSS10, MMR10, Rum04], refinement [PR99, KPR97, PR94], decomposition [PR99, KRW20], synthesis [MRR14a], refactoring [Rum12, PR03], translating models from one language into another [MRR11c, Rum12], and systematic model transformation language development [Wei12]. [Rum04] describes how comprehensible sets of such transformations support software development and maintenance [LRSS10], technologies for evolving models within a language and across languages, and mapping architecture descriptions to their implementation [MMR10]. Automaton refinement is discussed in [PR94, KPR97], refining pipe-and-filter architectures is explained in [PR99]. Refactorings of models are important for model driven engineering as discussed in [PR01, PR03, Rum12]. Translation between languages, e.g., from class diagrams into Alloy [MRR11c] allows for comparing class diagrams on a semantic level.

Variability and Software Product Lines (SPL)

Products often exist in various variants, for example cars or mobile phones, where one manufacturer develops several products with many similarities but also many variations. Variants are managed in a Software Product Line (SPL) that captures product commonalities as well as differences. Feature diagrams describe variability in a top down fashion, e.g., in the automotive domain [GHK+08a] using 150% models. Reducing overhead and associated costs is discussed in [GRJA12]. Delta modeling is a bottom up technique starting with a small, but complete base variant. Features are additive, but also can modify the core. A set of commonly applicable deltas configures a system variant. We discuss the application of this technique to Delta-MontiArc [HRR+11, HRR+11] and to Delta-Simulink [HKM+13]. Deltas can not only describe spacial variability but also temporal variability which allows for using them for software product line evolution [HRRS12]. [HHK+13] and [HRW15] describe an approach to systematically derive delta languages. We also apply variability modeling languages in order to describe syntactic and semantic variation points, e.g., in UML for frameworks [PFR02] and generators [GMR+16]. Furthermore, we specified a systematic way to define variants of modeling languages [CGR09], leverage features for compositional reuse [BEK+18b], and applied it as a semantic language refinement on Statecharts in [GR11].

Modeling for Cyber-Physical Systems (CPS)

Cyber-Physical Systems (CPS) [KRS12] are complex, distributed systems which control physical entities. In [RW18], we discuss how an elaborated theory can be practically applied for the development of CPS. Contributions for individual aspects range from requirements [GRJA12], complete product lines [HRRW12], the improvement of engineering for distributed automotive systems [HRR12], autonomous driving [BR12a, KKR19], and digital twin development [BDH+20] to processes and tools to improve the development as well as the product itself [BBR07]. In the aviation domain, a modeling language for uncertainty and safety events was developed, which is of interest for the European airspace [ZPK+11]. A component and connector architecture description language suitable for the specific challenges in robotics is discussed in [RRW13b, RRW14]. In [RRW13a], we describe a code generation framework for this language. Monitoring for smart and energy efficient buildings is developed as Energy Navigator toolset [KPR12, FPPR12, KLPR12].

Model-Driven Systems Engineering (MDSysE)

Applying models during Systems Engineering activities is based on the long tradition on contributing to systems engineering in automotive [GHK+08b], which culminated in a new comprehensive model-driven development process for automotive software [KMS+18, DGH+19]. We leveraged SysML to enable the integrated flow from requirements to implementation to integration. To facilitate modeling of products, resources, and processes in the context of Industry 4.0, we also conceived a multi-level framework for machining based on these concepts [BKL+18]. Research within the excellence cluster Internet of Production considers fast decision making at production time with low latencies using contextual data traces of production systems, also known as Digital Shadows (DS) [SHH+20]. We have investigated how to derive Digital Twins (DTs) for injection molding [BDH+20], how to generate interfaces between a cyber-physical system and its DT [KMR+20] and have proposed model-driven architectures for DT cockpit engineering [DMR+20].

State Based Modeling (Automata)

Today, many computer science theories are based on statemachines in various forms including Petri nets or temporal logics. Software engineering is particularly interested in using statemachines for modeling systems. Our contributions to state based modeling can currently be split into three parts: (1) understanding how to model object-oriented and distributed software using statemachines resp. Statecharts [GKR96, BCR07b, BCGR09b, BCGR09a], (2) understanding the refinement [PR94, RK96, Rum96, RW18] and composition [GR95, GKR96, RW18] of statemachines, and (3) applying statemachines for modeling systems. In [Rum96, RW18] constructive transformation rules for refining automata behavior are given and proven correct. This theory is applied to features in [KPR97]. Statemachines are embedded in the composition and behavioral specification concepts of Focus [GKR96, BR07]. We apply these

techniques, e.g., in MontiArcAutomaton [RRW13a, RRW14, RRW13a, RW18] as well as in building management systems [FLP+11b].

Model-Based Assistance and Information Services (MBAIS)

Assistive systems are a special type of information system: they (1) provide situational support for human behaviour (2) based on information from previously stored and real-time monitored structural context and behaviour data (3) at the time the person needs or asks for it [HMR+19]. To create them, we follow a model centered architecture approach [MMR+17] which defines systems as a compound of various connected models. Used languages for their definition include DSLs for behavior and structure such as the human cognitive modeling language [MM13], goal modeling languages [MRV20] or UML/P based languages [MNRV19]. [MM15] describes a process how languages for assistive systems can be created.

We have designed a system included in a sensor floor able to monitor elderlies and analyze impact patterns for emergency events [LMK+11]. We have investigated the modeling of human contexts for the active assisted living and smart home domain [MS17] and user-centered privacy-driven systems in the IoT domain in combination with process mining systems [MKM+19], differential privacy on event logs of handling and treatment of patients at a hospital [MKB+19], the mark-up of online manuals for devices [SM18] and websites [SM20], and solutions for privacy-aware environments for cloud services [ELR+17] and in IoT manufacturing [MNRV19]. The user-centered view on the system design allows to track who does what, when, why, where and how with personal data, makes information about it available via information services and provides support using assistive services.

Modelling Robotics Architectures and Tasks

Robotics can be considered a special field within Cyber-Physical Systems which is defined by an inherent heterogeneity of involved domains, relevant platforms, and challenges. The engineering of robotics applications requires composition and interaction of diverse distributed software modules. This usually leads to complex monolithic software solutions hardly reusable, maintainable, and comprehensible, which hampers broad propagation of robotics applications. The MontiArcAutomaton language [RRW13a] extends the ADL MontiArc and integrates various implemented behavior modeling languages using MontiCore [RRW13b, RRW14, RRRW15, HR17] that perfectly fit robotic architectural modeling. The LightRocks [THR+13] framework allows robotics experts and laymen to model robotic assembly tasks. In [AHRW17a, AHRW17b], we define a modular architecture modeling method for translating architecture models into modules compatible to different robotics middleware platforms.

Automotive, Autonomic Driving & Intelligent Driver Assistance

Introducing and connecting sophisticated driver assistance, infotainment and communication systems as well as advanced active and passive safety-systems result in complex embedded systems. As these

feature-driven subsystems may be arbitrarily combined by the customer, a huge amount of distinct variants needs to be managed, developed and tested. A consistent requirements management that connects requirements with features in all phases of the development for the automotive domain is described in [GRJA12]. The conceptual gap between requirements and the logical architecture of a car is closed in [GHK+07, GHK+08a]. [HKM+13] describes a tool for delta modeling for Simulink [HKM+13]. [HRRW12] discusses means to extract a well-defined Software Product Line from a set of copy and paste variants. [RSW+15] describes an approach to use model checking techniques to identify behavioral differences of Simulink models. In [KKR19], we introduce a framework for modeling the dynamic reconfiguration of component and connector architectures and apply it to the domain of cooperating vehicles. Quality assurance, especially of safety-related functions, is a highly important task. In the Carolo project [BR12a, BR12b], we developed a rigorous test infrastructure for intelligent, sensor-based functions through fully-automatic simulation [BBR07]. This technique allows a dramatic speedup in development and evolution of autonomous car functionality, and thus enables us to develop software in an agile way [BR12a]. [MMR10] gives an overview of the current state-of-the-art in development and evolution on a more general level by considering any kind of critical system that relies on architectural descriptions. As tooling infrastructure, the SSElab storage, versioning and management services [HKR12] are essential for many projects.

Smart Energy Management

In the past years, it became more and more evident that saving energy and reducing CO2 emissions is an important challenge. Thus, energy management in buildings as well as in neighbourhoods becomes equally important to efficiently use the generated energy. Within several research projects, we developed methodologies and solutions for integrating heterogeneous systems at different scales. During the design phase, the Energy Navigators Active Functional Specification (AFS) [FPPR12, KPR12] is used for technical specification of building services already. We adapted the well-known concept of statemachines to be able to describe different states of a facility and to validate it against the monitored values [FLP+11b]. We show how our data model, the constraint rules, and the evaluation approach to compare sensor data can be applied [KLPR12].

Cloud Computing & Enterprise Information Systems

The paradigm of Cloud Computing is arising out of a convergence of existing technologies for web-based application and service architectures with high complexity, criticality, and new application domains. It promises to enable new business models, to lower the barrier for web-based innovations and to increase the efficiency and cost-effectiveness of web development [KRR14]. Application classes like Cyber-Physical Systems and their privacy [HHK+14, HHK+15b], Big Data, App, and Service Ecosystems bring attention to aspects like responsiveness, privacy and open platforms. Regardless of the application domain, developers of such systems are in need for robust methods and efficient, easy-to-use languages and tools [KRS12]. We tackle these challenges by perusing a model-based, generative approach [NPR13]. The

core of this approach are different modeling languages that describe different aspects of a cloud-based system in a concise and technology-agnostic way. Software architecture and infrastructure models describe the system and its physical distribution on a large scale. We apply cloud technology for the services we develop, e.g., the SSELab [HKR12] and the Energy Navigator [FPPR12, KPR12] but also for our tool demonstrators and our own development platforms. New services, e.g., collecting data from temperature, cars etc. can now easily be developed.

Model-Driven Engineering of Information Systems

Information Systems provide information to different user groups as main system goal. Using our experiences in the model-based generation of code with MontiCore [KRV10, HR17], we developed several generators for such data-centric information systems. *MontiGem* [AMN+20] is a specific generator framework for data-centric business applications that uses standard models from UML/P optionally extended by GUI description models as sources [GMN+20]. While the standard semantics of these modeling languages remains untouched, the generator produces a lot of additional functionality around these models. The generator is designed flexible, modular and incremental, handwritten and generated code pieces are well integrated [GHK+15a], tagging of existing models is possible [GLRR15], e.g., for the definition of roles and rights or for testing [DGH+18].

Literaturverzeichnis

[AHRW17a] Kai Adam, Katrin Hölldobler, Bernhard Rumpe, and Andreas Wortmann. Engineering Robotics Software Architectures with Exchangeable Model Transformations. In *International Conference on Robotic Computing (IRC'17)*, pages 172–179. IEEE, April 2017.

[AHRW17b] Kai Adam, Katrin Hölldobler, Bernhard Rumpe, and Andreas Wortmann. Modeling Robotics Software Architectures with Modular Model Transformations. *Journal of Software Engineering for Robotics (JOSER)*, 8(1):3–16, 2017.

[AMN+20] Kai Adam, Judith Michael, Lukas Netz, Bernhard Rumpe, and Simon Varga. Enterprise Information Systems in Academia and Practice: Lessons learned from a MBSE Project. In *40 Years EMISA: Digital Ecosystems of the Future: Methodology, Techniques and Applications (EMISA'19)*, LNI P-304, pages 59–66. Gesellschaft für Informatik e.V., May 2020.

[BBR07] Christian Basarke, Christian Berger, and Bernhard Rumpe. Software & Systems Engineering Process and Tools for the Development of Autonomous Driving Intelligence. *Journal of Aerospace Computing, Information, and Communication (JACIC)*, 4(12):1158–1174, 2007.

[BCGR09a] Manfred Broy, María Victoria Cengarle, Hans Grönniger, and Bernhard Rumpe. Considerations and Rationale for a UML System Model. In K. Lano, editor, *UML 2 Semantics and Applications*, pages 43–61. John Wiley & Sons, November 2009.

[BCGR09b] Manfred Broy, María Victoria Cengarle, Hans Grönniger, and Bernhard Rumpe. Definition of the UML System Model. In K. Lano, editor, *UML 2 Semantics and Applications*, pages 63–93. John Wiley & Sons, November 2009.

[BCR07a] Manfred Broy, María Victoria Cengarle, and Bernhard Rumpe. Towards a System Model for UML. Part 2: The Control Model. Technical Report TUM-I0710, TU Munich, Germany, February 2007.

[BCR07b] Manfred Broy, María Victoria Cengarle, and Bernhard Rumpe. Towards a System Model for UML. Part 3: The State Machine Model. Technical Report TUM-I0711, TU Munich, Germany, February 2007.

[BDH+20] Pascal Bibow, Manuela Dalibor, Christian Hopmann, Ben Mainz, Bernhard Rumpe, David Schmalzing, Mauritius Schmitz, and Andreas Wortmann. Model-Driven Development of a Digital Twin for Injection Molding. In Schahram Dustdar, Eric Yu, Camille Salinesi,

Dominique Rieu, and Vik Pant, editors, *International Conference on Advanced Information Systems Engineering (CAiSE'20)*, Lecture Notes in Computer Science 12127, pages 85–100. Springer International Publishing, June 2020.

[BDL+18] Arvid Butting, Manuela Dalibor, Gerrit Leonhardt, Bernhard Rumpe, and Andreas Wortmann. Deriving Fluent Internal Domain-specific Languages from Grammars. In *International Conference on Software Language Engineering (SLE'18)*, pages 187–199. ACM, 2018.

[BEK+18a] Arvid Butting, Robert Eikermann, Oliver Kautz, Bernhard Rumpe, and Andreas Wortmann. Controlled and Extensible Variability of Concrete and Abstract Syntax with Independent Language Features. In *Proceedings of the 12th International Workshop on Variability Modelling of Software-Intensive Systems (VAMOS'18)*, pages 75–82. ACM, January 2018.

[BEK+18b] Arvid Butting, Robert Eikermann, Oliver Kautz, Bernhard Rumpe, and Andreas Wortmann. Modeling Language Variability with Reusable Language Components. In *International Conference on Systems and Software Product Line (SPLC'18)*. ACM, September 2018.

[BEK+19] Arvid Butting, Robert Eikermann, Oliver Kautz, Bernhard Rumpe, and Andreas Wortmann. Systematic Composition of Independent Language Features. *Journal of Systems and Software*, 152:50–69, June 2019.

[BGH+97] Ruth Breu, Radu Grosu, Christoph Hofmann, Franz Huber, Ingolf Krüger, Bernhard Rumpe, Monika Schmidt, and Wolfgang Schwerin. Exemplary and Complete Object Interaction Descriptions. In *Object-oriented Behavioral Semantics Workshop (OOPSLA'97)*, Technical Report TUM-I9737, Germany, 1997. TU Munich.

[BGH+98] Ruth Breu, Radu Grosu, Franz Huber, Bernhard Rumpe, and Wolfgang Schwerin. Systems, Views and Models of UML. In *Proceedings of the Unified Modeling Language, Technical Aspects and Applications*, pages 93–109. Physica Verlag, Heidelberg, Germany, 1998.

[BGRW17] Arvid Butting, Timo Greifenberg, Bernhard Rumpe, and Andreas Wortmann. Taming the Complexity of Model-Driven Systems Engineering Projects. Part of the Grand Challenges in Modeling (GRAND'17) Workshop, July 2017.

[BGRW18] Arvid Butting, Timo Greifenberg, Bernhard Rumpe, and Andreas Wortmann. On the Need for Artifact Models in Model-Driven Systems Engineering Projects. In Martina Seidl and Steffen Zschaler, editors, *Software Technologies: Applications and Foundations*, LNCS 10748, pages 146–153. Springer, January 2018.

[BHK+17] Arvid Butting, Robert Heim, Oliver Kautz, Jan Oliver Ringert, Bernhard Rumpe, and Andreas Wortmann. A Classification of Dynamic Reconfiguration in Component and Connector Architecture Description Languages. In *Proceedings of MODELS 2017. Workshop ModComp*, CEUR 2019, September 2017.

[BHP+98] Manfred Broy, Franz Huber, Barbara Paech, Bernhard Rumpe, and Katharina Spies. Software and System Modeling Based on a Unified Formal Semantics. In *Workshop on Requirements Targeting Software and Systems Engineering (RTSE'97)*, LNCS 1526, pages 43–68. Springer, 1998.

[BJRW18] Arvid Butting, Nico Jansen, Bernhard Rumpe, and Andreas Wortmann. Translating Grammars to Accurate Metamodels. In *International Conference on Software Language Engineering (SLE'18)*, pages 174–186. ACM, 2018.

[BKL+18] Christian Brecher, Evgeny Kusmenko, Achim Lindt, Bernhard Rumpe, Simon Storms, Stephan Wein, Michael von Wenckstern, and Andreas Wortmann. Multi-Level Modeling Framework for Machine as a Service Applications Based on Product Process Resource Models. In *Proceedings of the 2nd International Symposium on Computer Science and Intelligent Control (ISCSIC'18)*. ACM, September 2018.

[BKRW17] Arvid Butting, Oliver Kautz, Bernhard Rumpe, and Andreas Wortmann. Semantic Differencing for Message-Driven Component & Connector Architectures. In *International Conference on Software Architecture (ICSA'17)*, pages 145–154. IEEE, April 2017.

[BKRW19] Arvid Butting, Oliver Kautz, Bernhard Rumpe, and Andreas Wortmann. Continuously Analyzing Finite, Message-Driven, Time-Synchronous Component & Connector Systems During Architecture Evolution. *Journal of Systems and Software*, 149:437–461, March 2019.

[BR07] Manfred Broy and Bernhard Rumpe. Modulare hierarchische Modellierung als Grundlage der Software- und Systementwicklung. *Informatik-Spektrum*, 30(1):3–18, Februar 2007.

[BR12a] Christian Berger and Bernhard Rumpe. Autonomous Driving - 5 Years after the Urban Challenge: The Anticipatory Vehicle as a Cyber-Physical System. In *Automotive Software Engineering Workshop (ASE'12)*, pages 789–798, 2012.

[BR12b] Christian Berger and Bernhard Rumpe. Engineering Autonomous Driving Software. In C. Rouff and M. Hinchey, editors, *Experience from the DARPA Urban Challenge*, pages 243–271. Springer, Germany, 2012.

[CBCR15] Tony Clark, Mark van den Brand, Benoit Combemale, and Bernhard Rumpe. Conceptual Model of the Globalization for Domain-Specific Languages. In *Globalizing Domain-Specific Languages*, LNCS 9400, pages 7–20. Springer, 2015.

[CCF+15] Betty H. C. Cheng, Benoit Combemale, Robert B. France, Jean-Marc Jézéquel, and Bernhard Rumpe, editors. *Globalizing Domain-Specific Languages*, LNCS 9400. Springer, 2015.

[CEG+14] Betty Cheng, Kerstin Eder, Martin Gogolla, Lars Grunske, Marin Litoiu, Hausi Müller, Patrizio Pelliccione, Anna Perini, Nauman Qureshi, Bernhard Rumpe, Daniel Schneider, Frank Trollmann, and Norha Villegas. Using Models at Runtime to Address Assurance

for Self-Adaptive Systems. In *Models@run.time*, LNCS 8378, pages 101–136. Springer, Germany, 2014.

[CGR08] María Victoria Cengarle, Hans Grönniger, and Bernhard Rumpe. System Model Semantics of Class Diagrams. Informatik-Bericht 2008-05, TU Braunschweig, Germany, 2008.

[CGR09] María Victoria Cengarle, Hans Grönniger, and Bernhard Rumpe. Variability within Modeling Language Definitions. In *Conference on Model Driven Engineering Languages and Systems (MODELS'09)*, LNCS 5795, pages 670–684. Springer, 2009.

[CKM+18] Benoit Combemale, Jörg Kienzle, Gunter Mussbacher, Olivier Barais, Erwan Bousse, Walter Cazzola, Philippe Collet, Thomas Degueule, Robert Heinrich, Jean-Marc Jézéquel, Manuel Leduc, Tanja Mayerhofer, Sébastien Mosser, Matthias Schöttle, Misha Strittmatter, and Andreas Wortmann. Concern-Oriented Language Development (COLD): Fostering Reuse in Language Engineering. *Computer Languages, Systems & Structures*, 54:139 – 155, 2018.

[DEKR19] Imke Drave, Robert Eikermann, Oliver Kautz, and Bernhard Rumpe. Semantic Differencing of Statecharts for Object-oriented Systems. In Slimane Hammoudi, Luis Ferreira Pires, and Bran Selić, editors, *Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development (MODELSWARD'19)*, pages 274–282. SciTePress, February 2019.

[DGH+18] Imke Drave, Timo Greifenberg, Steffen Hillemacher, Stefan Kriebel, Matthias Markthaler, Bernhard Rumpe, and Andreas Wortmann. Model-Based Testing of Software-Based System Functions. In *Conference on Software Engineering and Advanced Applications (SEAA'18)*, pages 146–153, August 2018.

[DGH+19] Imke Drave, Timo Greifenberg, Steffen Hillemacher, Stefan Kriebel, Evgeny Kusmenko, Matthias Markthaler, Philipp Orth, Karin Samira Salman, Johannes Richenhagen, Bernhard Rumpe, Christoph Schulze, Michael Wenckstern, and Andreas Wortmann. SMArDT modeling for automotive software testing. *Software: Practice and Experience*, 49(2):301–328, February 2019.

[DHH+20] Imke Drave, Timo Henrich, Katrin Hölldobler, Oliver Kautz, Judith Michael, and Bernhard Rumpe. Modellierung, Verifikation und Synthese von validen Planungszuständen fÃ¼r Fernsehausstrahlungen. In Dominik Bork, Dimitris Karagiannis, and Heinrich C. Mayr, editors, *Modellierung 2020*, pages 173–188. Gesellschaft fÃ¼r Informatik e.V., February 2020.

[DKMR19] Imke Drave, Oliver Kautz, Judith Michael, and Bernhard Rumpe. Semantic Evolution Analysis of Feature Models. In Thorsten Berger, Philippe Collet, Laurence Duchien, Thomas Fogdal, Patrick Heymans, Timo Kehrer, Jabier Martinez, Raúl Mazo, Leticia Montalvillo, Camille Salinesi, Xhevahire Tërnava, Thomas Thüm, and Tewfik Ziadi, editors, *International Systems and Software Product Line Conference (SPLC'19)*, pages 245–255. ACM, September 2019.

[DMR+20] Manuela Dalibor, Judith Michael, Bernhard Rumpe, Simon Varga, and Andreas Wortmann. Towards a Model-Driven Architecture for Interactive Digital Twin Cockpits. In Gillian Dobbie, Ulrich Frank, Gerti Kappel, Stephen W. Liddle, and Heinrich C. Mayr, editors, *Conceptual Modeling*, pages 377–387. Springer International Publishing, October 2020.

[EFLR99] Andy Evans, Robert France, Kevin Lano, and Bernhard Rumpe. Meta-Modelling Semantics of UML. In H. Kilov, B. Rumpe, and I. Simmonds, editors, *Behavioral Specifications of Businesses and Systems*, pages 45–60. Kluver Academic Publisher, 1999.

[EJK+19] Rolf Ebert, Jahir Jolianis, Stefan Kriebel, Matthias Markthaler, Benjamin Pruenster, Bernhard Rumpe, and Karin Samira Salman. Applying Product Line Testing for the Electric Drive System. In Thorsten Berger, Philippe Collet, Laurence Duchien, Thomas Fogdal, Patrick Heymans, Timo Kehrer, Jabier Martinez, Raúl Mazo, Leticia Montalvillo, Camille Salinesi, Xhevahire Tërnava, Thomas Thüm, and Tewfik Ziadi, editors, *International Systems and Software Product Line Conference (SPLC'19)*, pages 14–24. ACM, September 2019.

[ELR+17] Robert Eikermann, Markus Look, Alexander Roth, Bernhard Rumpe, and Andreas Wortmann. Architecting Cloud Services for the Digital me in a Privacy-Aware Environment. In Ivan Mistrik, Rami Bahsoon, Nour Ali, Maritta Heisel, and Bruce Maxim, editors, *Software Architecture for Big Data and the Cloud*, chapter 12, pages 207–226. Elsevier Science & Technology, June 2017.

[FELR98] Robert France, Andy Evans, Kevin Lano, and Bernhard Rumpe. The UML as a formal modeling notation. *Computer Standards & Interfaces*, 19(7):325–334, November 1998.

[FHR08] Florian Fieber, Michaela Huhn, and Bernhard Rumpe. Modellqualität als Indikator für Softwarequalität: eine Taxonomie. *Informatik-Spektrum*, 31(5):408–424, Oktober 2008.

[FLP+11a] M. Norbert Fisch, Markus Look, Claas Pinkernell, Stefan Plesser, and Bernhard Rumpe. Der Energie-Navigator - Performance-Controlling für Gebäude und Anlagen. *Technik am Bau (TAB) - Fachzeitschrift für Technische Gebäudeausrüstung*, Seiten 36-41, März 2011.

[FLP+11b] M. Norbert Fisch, Markus Look, Claas Pinkernell, Stefan Plesser, and Bernhard Rumpe. State-based Modeling of Buildings and Facilities. In *Enhanced Building Operations Conference (ICEBO'11)*, 2011.

[FPPR12] M. Norbert Fisch, Claas Pinkernell, Stefan Plesser, and Bernhard Rumpe. The Energy Navigator - A Web-Platform for Performance Design and Management. In *Energy Efficiency in Commercial Buildings Conference(IEECB'12)*, 2012.

[GHK+07] Hans Grönniger, Jochen Hartmann, Holger Krahn, Stefan Kriebel, and Bernhard Rumpe. View-based Modeling of Function Nets. In *Object-oriented Modelling of Embedded Real-Time Systems Workshop (OMER4'07)*, 2007.

[GHK+08a] Hans Grönniger, Jochen Hartmann, Holger Krahn, Stefan Kriebel, Lutz Rothhardt, and Bernhard Rumpe. Modelling Automotive Function Nets with Views for Features, Variants, and Modes. In *Proceedings of 4th European Congress ERTS - Embedded Real Time Software*, 2008.

[GHK+08b] Hans Grönniger, Jochen Hartmann, Holger Krahn, Stefan Kriebel, Lutz Rothhardt, and Bernhard Rumpe. View-Centric Modeling of Automotive Logical Architectures. In *Tagungsband des Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteter Systeme IV*, Informatik Bericht 2008-02. TU Braunschweig, 2008.

[GHK+15a] Timo Greifenberg, Katrin Hölldobler, Carsten Kolassa, Markus Look, Pedram Mir Seyed Nazari, Klaus Müller, Antonio Navarro Perez, Dimitri Plotnikov, Dirk Reiß, Alexander Roth, Bernhard Rumpe, Martin Schindler, and Andreas Wortmann. Integration of Handwritten and Generated Object-Oriented Code. In *Model-Driven Engineering and Software Development*, Communications in Computer and Information Science 580, pages 112–132. Springer, 2015.

[GHK+15b] Timo Greifenberg, Katrin Hölldobler, Carsten Kolassa, Markus Look, Pedram Mir Seyed Nazari, Klaus Müller, Antonio Navarro Perez, Dimitri Plotnikov, Dirk Reiß, Alexander Roth, Bernhard Rumpe, Martin Schindler, and Andreas Wortmann. A Comparison of Mechanisms for Integrating Handwritten and Generated Code for Object-Oriented Programming Languages. In *Model-Driven Engineering and Software Development Conference (MODELSWARD'15)*, pages 74–85. SciTePress, 2015.

[GHR17] Timo Greifenberg, Steffen Hillemacher, and Bernhard Rumpe. *Towards a Sustainable Artifact Model: Artifacts in Generator-Based Model-Driven Projects*. Aachener Informatik-Berichte, Software Engineering, Band 30. Shaker Verlag, December 2017.

[GKPR08] Hans Grönniger, Holger Krahn, Claas Pinkernell, and Bernhard Rumpe. Modeling Variants of Automotive Systems using Views. In *Modellbasierte Entwicklung von eingebetteten Fahrzeugfunktionen*, Informatik Bericht 2008-01, pages 76–89. TU Braunschweig, 2008.

[GKR96] Radu Grosu, Cornel Klein, and Bernhard Rumpe. Enhancing the SysLab System Model with State. Technical Report TUM-I9631, TU Munich, Germany, July 1996.

[GKR+06] Hans Grönniger, Holger Krahn, Bernhard Rumpe, Martin Schindler, and Steven Völkel. MontiCore 1.0 - Ein Framework zur Erstellung und Verarbeitung domänspezifischer Sprachen. Informatik-Bericht 2006-04, CFG-Fakultät, TU Braunschweig, August 2006.

[GKR+07] Hans Grönniger, Holger Krahn, Bernhard Rumpe, Martin Schindler, and Steven Völkel. Textbased Modeling. In *4th International Workshop on Software Language Engineering, Nashville*, Informatik-Bericht 4/2007. Johannes-Gutenberg-Universität Mainz, 2007.

[GKR+08] Hans Grönniger, Holger Krahn, Bernhard Rumpe, Martin Schindler, and Steven Völkel. MontiCore: A Framework for the Development of Textual Domain Specific Languages. In

30th International Conference on Software Engineering (ICSE 2008), Leipzig, Germany, May 10-18, 2008, Companion Volume, pages 925–926, 2008.

[GKRS06] Hans Grönniger, Holger Krahn, Bernhard Rumpe, and Martin Schindler. Integration von Modellen in einen codebasierten Softwareentwicklungsprozess. In *Modellierung 2006 Conference*, LNI 82, Seiten 67-81, 2006.

[GLPR15] Timo Greifenberg, Markus Look, Claas Pinkernell, and Bernhard Rumpe. Energieeffiziente Städte - Herausforderungen und Lösungen aus Sicht des Software Engineerings. In Linnhoff-Popien, Claudia and Zaddach, Michael and Grahl, Andreas, Editor, *Marktplätze im Umbruch: Digitale Strategien für Services im Mobilen Internet*, Xpert.press, Kapitel 56, Seiten 511-520. Springer Berlin Heidelberg, April 2015.

[GLRR15] Timo Greifenberg, Markus Look, Sebastian Roidl, and Bernhard Rumpe. Engineering Tagging Languages for DSLs. In *Conference on Model Driven Engineering Languages and Systems (MODELS'15)*, pages 34–43. ACM/IEEE, 2015.

[GMN+20] Arkadii Gerasimov, Judith Michael, Lukas Netz, Bernhard Rumpe, and Simon Varga. Continuous Transition from Model-Driven Prototype to Full-Size Real-World Enterprise Information Systems. In Bonnie Anderson, Jason Thatcher, and Rayman Meservy, editors, *25th Americas Conference on Information Systems (AMCIS 2020)*, AIS Electronic Library (AISeL), pages 1–10. Association for Information Systems (AIS), August 2020.

[GMR+16] Timo Greifenberg, Klaus Müller, Alexander Roth, Bernhard Rumpe, Christoph Schulze, and Andreas Wortmann. Modeling Variability in Template-based Code Generators for Product Line Engineering. In *Modellierung 2016 Conference*, LNI 254, pages 141–156. Bonner Köllen Verlag, March 2016.

[GR95] Radu Grosu and Bernhard Rumpe. Concurrent Timed Port Automata. Technical Report TUM-I9533, TU Munich, Germany, October 1995.

[GR11] Hans Grönniger and Bernhard Rumpe. Modeling Language Variability. In *Workshop on Modeling, Development and Verification of Adaptive Systems*, LNCS 6662, pages 17–32. Springer, 2011.

[GRJA12] Tim Gülke, Bernhard Rumpe, Martin Jansen, and Joachim Axmann. High-Level Requirements Management and Complexity Costs in Automotive Development Projects: A Problem Statement. In *Requirements Engineering: Foundation for Software Quality (REFSQ'12)*, 2012.

[GRR10] Hans Grönniger, Dirk Reiß, and Bernhard Rumpe. Towards a Semantics of Activity Diagrams with Semantic Variation Points. In *Conference on Model Driven Engineering Languages and Systems (MODELS'10)*, LNCS 6394, pages 331–345. Springer, 2010.

[HHK+13] Arne Haber, Katrin Hölldobler, Carsten Kolassa, Markus Look, Klaus Müller, Bernhard

Rumpe, and Ina Schaefer. Engineering Delta Modeling Languages. In *Software Product Line Conference (SPLC'13)*, pages 22–31. ACM, 2013.

[HHK+14] Martin Henze, Lars Hermerschmidt, Daniel Kerpen, Roger Häußling, Bernhard Rumpe, and Klaus Wehrle. User-driven Privacy Enforcement for Cloud-based Services in the Internet of Things. In *Conference on Future Internet of Things and Cloud (FiCloud'14)*. IEEE, 2014.

[HHK+15a] Arne Haber, Katrin Hölldobler, Carsten Kolassa, Markus Look, Klaus Müller, Bernhard Rumpe, Ina Schaefer, and Christoph Schulze. Systematic Synthesis of Delta Modeling Languages. *Journal on Software Tools for Technology Transfer (STTT)*, 17(5):601–626, October 2015.

[HHK+15b] Martin Henze, Lars Hermerschmidt, Daniel Kerpen, Roger Häußling, Bernhard Rumpe, and Klaus Wehrle. A comprehensive approach to privacy in the cloud-based Internet of Things. *Future Generation Computer Systems*, 56:701–718, 2015.

[HKM+13] Arne Haber, Carsten Kolassa, Peter Manhart, Pedram Mir Seyed Nazari, Bernhard Rumpe, and Ina Schaefer. First-Class Variability Modeling in Matlab/Simulink. In *Variability Modelling of Software-intensive Systems Workshop (VaMoS'13)*, pages 11–18. ACM, 2013.

[HKR+07] Christoph Herrmann, Holger Krahn, Bernhard Rumpe, Martin Schindler, and Steven Völkel. An Algebraic View on the Semantics of Model Composition. In *Conference on Model Driven Architecture - Foundations and Applications (ECMDA-FA'07)*, LNCS 4530, pages 99–113. Springer, Germany, 2007.

[HKR+09] Christoph Herrmann, Holger Krahn, Bernhard Rumpe, Martin Schindler, and Steven Völkel. Scaling-Up Model-Based-Development for Large Heterogeneous Systems with Compositional Modeling. In *Conference on Software Engineeering in Research and Practice (SERP'09)*, pages 172–176, July 2009.

[HKR+11] Arne Haber, Thomas Kutz, Holger Rendel, Bernhard Rumpe, and Ina Schaefer. Delta-oriented Architectural Variability Using MontiCore. In *Software Architecture Conference (ECSA'11)*, pages 6:1–6:10. ACM, 2011.

[HKR12] Christoph Herrmann, Thomas Kurpick, and Bernhard Rumpe. SSELab: A Plug-In-Based Framework for Web-Based Project Portals. In *Developing Tools as Plug-Ins Workshop (TOPI'12)*, pages 61–66. IEEE, 2012.

[HLMSN+15a] Arne Haber, Markus Look, Pedram Mir Seyed Nazari, Antonio Navarro Perez, Bernhard Rumpe, Steven Völkel, and Andreas Wortmann. Composition of Heterogeneous Modeling Languages. In *Model-Driven Engineering and Software Development*, Communications in Computer and Information Science 580, pages 45–66. Springer, 2015.

[HLMSN+15b] Arne Haber, Markus Look, Pedram Mir Seyed Nazari, Antonio Navarro Perez, Bernhard Rumpe, Steven Völkel, and Andreas Wortmann. Integration of Heterogeneous Modeling Languages via Extensible and Composable Language Components. In *Model-Driven*

Engineering and Software Development Conference (MODELSWARD'15), pages 19–31. SciTePress, 2015.

[HMR+19] Katrin Hölldobler, Judith Michael, Jan Oliver Ringert, Bernhard Rumpe, and Andreas Wortmann. Innovations in Model-based Software and Systems Engineering. *The Journal of Object Technology*, 18(1):1–60, July 2019.

[HMSNRW16] Robert Heim, Pedram Mir Seyed Nazari, Bernhard Rumpe, and Andreas Wortmann. Compositional Language Engineering using Generated, Extensible, Static Type Safe Visitors. In *Conference on Modelling Foundations and Applications (ECMFA)*, LNCS 9764, pages 67–82. Springer, July 2016.

[HR04] David Harel and Bernhard Rumpe. Meaningful Modeling: What's the Semantics of "Semantics"? *IEEE Computer*, 37(10):64–72, October 2004.

[HR17] Katrin Hölldobler and Bernhard Rumpe. *MontiCore 5 Language Workbench Edition 2017*. Aachener Informatik-Berichte, Software Engineering, Band 32. Shaker Verlag, December 2017.

[HRR98] Franz Huber, Andreas Rausch, and Bernhard Rumpe. Modeling Dynamic Component Interfaces. In *Technology of Object-Oriented Languages and Systems (TOOLS 26)*, pages 58–70. IEEE, 1998.

[HRR+11] Arne Haber, Holger Rendel, Bernhard Rumpe, Ina Schaefer, and Frank van der Linden. Hierarchical Variability Modeling for Software Architectures. In *Software Product Lines Conference (SPLC'11)*, pages 150–159. IEEE, 2011.

[HRR12] Arne Haber, Jan Oliver Ringert, and Bernhard Rumpe. MontiArc - Architectural Modeling of Interactive Distributed and Cyber-Physical Systems. Technical Report AIB-2012-03, RWTH Aachen University, February 2012.

[HRRS11] Arne Haber, Holger Rendel, Bernhard Rumpe, and Ina Schaefer. Delta Modeling for Software Architectures. In *Tagungsband des Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteterSysteme VII*, pages 1 – 10. fortiss GmbH, 2011.

[HRRS12] Arne Haber, Holger Rendel, Bernhard Rumpe, and Ina Schaefer. Evolving Delta-oriented Software Product Line Architectures. In *Large-Scale Complex IT Systems. Development, Operation and Management, 17th Monterey Workshop 2012*, LNCS 7539, pages 183–208. Springer, 2012.

[HRRW12] Christian Hopp, Holger Rendel, Bernhard Rumpe, and Fabian Wolf. Einführung eines Produktlinienansatzes in die automotive Softwareentwicklung am Beispiel von Steuergerätesoftware. In *Software Engineering Conference (SE'12)*, LNI 198, Seiten 181-192, 2012.

[HRW15] Katrin Hölldobler, Bernhard Rumpe, and Ingo Weisemöller. Systematically Deriving Domain-Specific Transformation Languages. In *Conference on Model Driven Engineering Languages and Systems (MODELS'15)*, pages 136–145. ACM/IEEE, 2015.

[HRW18] Katrin Hölldobler, Bernhard Rumpe, and Andreas Wortmann. Software Language Engineering in the Large: Towards Composing and Deriving Languages. *Computer Languages, Systems & Structures*, 54:386–405, 2018.

[JWCR18] Rodi Jolak, Andreas Wortmann, Michel Chaudron, and Bernhard Rumpe. Does Distance Still Matter? Revisiting Collaborative Distributed Software Design. *IEEE Software*, 35(6):40–47, 2018.

[KER99] Stuart Kent, Andy Evans, and Bernhard Rumpe. UML Semantics FAQ. In A. Moreira and S. Demeyer, editors, *Object-Oriented Technology, ECOOP'99 Workshop Reader*, LNCS 1743, Berlin, 1999. Springer Verlag.

[KKP+09] Gabor Karsai, Holger Krahn, Claas Pinkernell, Bernhard Rumpe, Martin Schindler, and Steven Völkel. Design Guidelines for Domain Specific Languages. In *Domain-Specific Modeling Workshop (DSM'09)*, Techreport B-108, pages 7–13. Helsinki School of Economics, October 2009.

[KKR19] Nils Kaminski, Evgeny Kusmenko, and Bernhard Rumpe. Modeling Dynamic Architectures of Self-Adaptive Cooperative Systems. *The Journal of Object Technology*, 18(2):1–20, July 2019. The 15th European Conference on Modelling Foundations and Applications.

[KLPR12] Thomas Kurpick, Markus Look, Claas Pinkernell, and Bernhard Rumpe. Modeling Cyber-Physical Systems: Model-Driven Specification of Energy Efficient Buildings. In *Modelling of the Physical World Workshop (MOTPW'12)*, pages 2:1–2:6. ACM, October 2012.

[KMR+20] Jörg Christian Kirchhof, Judith Michael, Bernhard Rumpe, Simon Varga, and Andreas Wortmann. Model-driven Digital Twin Construction: Synthesizing the Integration of Cyber-Physical Systems with Their Information Systems. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, pages 90–101. ACM, October 2020.

[KMS+18] Stefan Kriebel, Matthias Markthaler, Karin Samira Salman, Timo Greifenberg, Steffen Hillemacher, Bernhard Rumpe, Christoph Schulze, Andreas Wortmann, Philipp Orth, and Johannes Richenhagen. Improving Model-based Testing in Automotive Software Engineering. In *International Conference on Software Engineering: Software Engineering in Practice (ICSE'18)*, pages 172–180. ACM, June 2018.

[KNP+19] Evgeny Kusmenko, Sebastian Nickels, Svetlana Pavlitskaya, Bernhard Rumpe, and Thomas Timmermanns. Modeling and Training of Neural Processing Systems. In Marouane Kessentini, Tao Yue, Alexander Pretschner, Sebastian Voss, and Loli Burgueño, editors, *Conference on Model Driven Engineering Languages and Systems (MODELS'19)*, pages 283–293. IEEE, September 2019.

[KPR97] Cornel Klein, Christian Prehofer, and Bernhard Rumpe. Feature Specification and Refinement with State Transition Diagrams. In *Workshop on Feature Interactions in Telecommunications Networks and Distributed Systems*, pages 284–297. IOS-Press, 1997.

[KPR12] Thomas Kurpick, Claas Pinkernell, and Bernhard Rumpe. Der Energie Navigator. In H. Lichter and B. Rumpe, Editoren, *Entwicklung und Evolution von Forschungssoftware. Tagungsband, Rolduc, 10.-11.11.2011*, Aachener Informatik-Berichte, Software Engineering, Band 14. Shaker Verlag, Aachen, Deutschland, 2012.

[KPRS19] Evgeny Kusmenko, Svetlana Pavlitskaya, Bernhard Rumpe, and Sebastian Stüber. On the Engineering of AI-Powered Systems. In Lisa Oâ€™Conner, editor, *ASEâ€™19. Software Engineering Intelligence Workshop (SEIâ€™19)*, pages 126–133. IEEE, November 2019.

[KR18] Oliver Kautz and Bernhard Rumpe. On Computing Instructions to Repair Failed Model Refinements. In *Conference on Model Driven Engineering Languages and Systems (MODELS'18)*, pages 289–299. ACM, October 2018.

[Kra10] Holger Krahn. *MontiCore: Agile Entwicklung von domänenspezifischen Sprachen im Software-Engineering*. Aachener Informatik-Berichte, Software Engineering, Band 1. Shaker Verlag, März 2010.

[KRB96] Cornel Klein, Bernhard Rumpe, and Manfred Broy. A stream-based mathematical model for distributed information processing systems - SysLab system model. In *Workshop on Formal Methods for Open Object-based Distributed Systems*, IFIP Advances in Information and Communication Technology, pages 323–338. Chapmann & Hall, 1996.

[KRR14] Helmut Krcmar, Ralf Reussner, and Bernhard Rumpe. *Trusted Cloud Computing*. Springer, Schweiz, December 2014.

[KRRS19] Stefan Kriebel, Deni Raco, Bernhard Rumpe, and Sebastian Stüber. Model-Based Engineering for Avionics: Will Specification and Formal Verification e.g. Based on Broyâ€™s Streams Become Feasible? In Stephan Krusche, Kurt Schneider, Marco Kuhrmann, Robert Heinrich, Reiner Jung, Marco Konersmann, Eric Schmieders, Steffen Helke, Ina Schaefer, Andreas Vogelsang, Björn Annighöfer, Andreas Schweiger, Marina Reich, and André van Hoorn, editors, *Proceedings of the Workshops of the Software Engineering Conference. Workshop on Avionics Systems and Software Engineering (AvioSE'19)*, CEUR Workshop Proceedings 2308, pages 87–94. CEUR Workshop Proceedings, February 2019.

[KRRvW17] Evgeny Kusmenko, Alexander Roth, Bernhard Rumpe, and Michael von Wenckstern. Modeling Architectures of Cyber-Physical Systems. In *European Conference on Modelling Foundations and Applications (ECMFA'17)*, LNCS 10376, pages 34–50. Springer, July 2017.

[KRS12] Stefan Kowalewski, Bernhard Rumpe, and Andre Stollenwerk. Cyber-Physical Systems - eine Herausforderung für die Automatisierungstechnik? In *Proceedings of Automation 2012, VDI Berichte 2012*, Seiten 113-116. VDI Verlag, 2012.

[KRV06] Holger Krahn, Bernhard Rumpe, and Steven Völkel. Roles in Software Development using Domain Specific Modelling Languages. In *Domain-Specific Modeling Workshop (DSM'06)*, Technical Report TR-37, pages 150–158. Jyväskylä University, Finland, 2006.

[KRV07a] Holger Krahn, Bernhard Rumpe, and Steven Völkel. Efficient Editor Generation for Compositional DSLs in Eclipse. In *Domain-Specific Modeling Workshop (DSM'07)*, Technical Reports TR-38. Jyväskylä University, Finland, 2007.

[KRV07b] Holger Krahn, Bernhard Rumpe, and Steven Völkel. Integrated Definition of Abstract and Concrete Syntax for Textual Languages. In *Conference on Model Driven Engineering Languages and Systems (MODELS'07)*, LNCS 4735, pages 286–300. Springer, 2007.

[KRV08] Holger Krahn, Bernhard Rumpe, and Steven Völkel. Monticore: Modular Development of Textual Domain Specific Languages. In *Conference on Objects, Models, Components, Patterns (TOOLS-Europe'08)*, LNBIP 11, pages 297–315. Springer, 2008.

[KRV10] Holger Krahn, Bernhard Rumpe, and Stefen Völkel. MontiCore: a Framework for Compositional Development of Domain Specific Languages. *International Journal on Software Tools for Technology Transfer (STTT)*, 12(5):353–372, September 2010.

[KRV14] Holger Krahn, Bernhard Rumpe, and Steven Völkel. Roles in Software Development using Domain Specific Modeling Languages. In *Proceedings of the 6th OOPSLA Workshop on Domain-Specific Modeling (DSMâ€˜06)*. CoRR arXiv, 2014.

[KRW20] Oliver Kautz, Bernhard Rumpe, and Andreas Wortmann. Automated semantics-preserving parallel decomposition of finite component and connector architectures. *Automated Software Engineering*, 27:119–151, April 2020.

[LMK+11] Philipp Leusmann, Christian MÃ¶llering, Lars Klack, Kai Kasugai, Bernhard Rumpe, and Martina Ziefle. Your Floor Knows Where You Are: Sensing and Acquisition of Movement Data. In Arkady Zaslavsky, Panos K. Chrysanthis, Dik Lun Lee, Dipanjan Chakraborty, Vana Kalogeraki, Mohamed F. Mokbel, and Chi-Yin Chow, editors, *12th IEEE International Conference on Mobile Data Management (Volume 2)*, pages 61–66. IEEE, June 2011.

[LRSS10] Tihamer Levendovszky, Bernhard Rumpe, Bernhard Schätz, and Jonathan Sprinkle. Model Evolution and Management. In *Model-Based Engineering of Embedded Real-Time Systems Workshop (MBEERTS'10)*, LNCS 6100, pages 241–270. Springer, 2010.

[MKB+19] Felix Mannhardt, Agnes Koschmider, Nathalie Baracaldo, Matthias Weidlich, and Judith Michael. Privacy-Preserving Process Mining: Differential Privacy for Event Logs. *Business & Information Systems Engineering*, 61(5):1–20, October 2019.

[MKM+19] Judith Michael, Agnes Koschmider, Felix Mannhardt, Nathalie Baracaldo, and Bernhard Rumpe. User-Centered and Privacy-Driven Process Mining System Design for IoT. In Cinzia Cappiello and Marcela Ruiz, editors, *Proceedings of CAiSE Forum 2019: Information Systems Engineering in Responsible Information Systems*, pages 194–206. Springer, June 2019.

[MM13] Judith Michael and Heinrich C. Mayr. Conceptual modeling for ambient assistance. In *Conceptual Modeling - ER 2013*, LNCS 8217, pages 403–413. Springer, 2013.

[MM15] Judith Michael and Heinrich C. Mayr. Creating a domain specific modelling method for ambient assistance. In *International Conference on Advances in ICT for Emerging Regions (ICTer2015)*, pages 119–124. IEEE, 2015.

[MMR10] Tom Mens, Jeff Magee, and Bernhard Rumpe. Evolving Software Architecture Descriptions of Critical Systems. *IEEE Computer*, 43(5):42–48, May 2010.

[MMR+17] Heinrich C. Mayr, Judith Michael, Suneth Ranasinghe, Vladimir A. Shekhovtsov, and Claudia Steinberger. *Model Centered Architecture*, pages 85–104. Springer International Publishing, 2017.

[MNRV19] Judith Michael, Lukas Netz, Bernhard Rumpe, and Simon Varga. Towards Privacy-Preserving IoT Systems Using Model Driven Engineering. In Nicolas Ferry, Antonio Cicchetti, Federico Ciccozzi, Arnor Solberg, Manuel Wimmer, and Andreas Wortmann, editors, *Proceedings of MODELS 2019. Workshop MDE4IoT*, pages 595–614. CEUR Workshop Proceedings, September 2019.

[MRR10] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. A Manifesto for Semantic Model Differencing. In *Proceedings Int. Workshop on Models and Evolution (ME'10)*, LNCS 6627, pages 194–203. Springer, 2010.

[MRR11a] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. ADDiff: Semantic Differencing for Activity Diagrams. In *Conference on Foundations of Software Engineering (ESEC/FSE '11)*, pages 179–189. ACM, 2011.

[MRR11b] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. An Operational Semantics for Activity Diagrams using SMV. Technical Report AIB-2011-07, RWTH Aachen University, Aachen, Germany, July 2011.

[MRR11c] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. CD2Alloy: Class Diagrams Analysis Using Alloy Revisited. In *Conference on Model Driven Engineering Languages and Systems (MODELS'11)*, LNCS 6981, pages 592–607. Springer, 2011.

[MRR11d] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. CDDiff: Semantic Differencing for Class Diagrams. In Mira Mezini, editor, *ECOOP 2011 - Object-Oriented Programming*, pages 230–254. Springer Berlin Heidelberg, 2011.

[MRR11e] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. Modal Object Diagrams. In *Object-Oriented Programming Conference (ECOOP'11)*, LNCS 6813, pages 281–305. Springer, 2011.

[MRR11f] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. Semantically Configurable Consistency Analysis for Class and Object Diagrams. In *Conference on Model Driven Engineering Languages and Systems (MODELS'11)*, LNCS 6981, pages 153–167. Springer, 2011.

[MRR11g] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. Summarizing Semantic Model

Differences. In Bernhard Schätz, Dirk Deridder, Alfonso Pierantonio, Jonathan Sprinkle, and Dalila Tamzalit, editors, *ME 2011 - Models and Evolution*, October 2011.

[MRR13] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. Synthesis of Component and Connector Models from Crosscutting Structural Views. In Meyer, B. and Baresi, L. and Mezini, M., editor, *Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE'13)*, pages 444–454. ACM New York, 2013.

[MRR14a] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. Synthesis of Component and Connector Models from Crosscutting Structural Views (extended abstract). In Wilhelm Hasselbring and Nils Christian Ehmke, editors, *Software Engineering 2014*, LNI 227, pages 63–64. Gesellschaft für Informatik, Köllen Druck+Verlag GmbH, 2014.

[MRR14b] Shahar Maoz, Jan Oliver Ringert, and Bernhard Rumpe. Verifying Component and Connector Models against Crosscutting Structural Views. In *Software Engineering Conference (ICSE'14)*, pages 95–105. ACM, 2014.

[MRV20] Judith Michael, Bernhard Rumpe, and Simon Varga. Human behavior, goals and model-driven software engineering for assistive systems. In Agnes Koschmider, Judith Michael, and Bernhard Thalheim, editors, *Enterprise Modeling and Information Systems Architectures (EMSIA 2020)*, pages 11–18. CEUR Workshop Proceedings, June 2020.

[MS17] Judith Michael and Claudia Steinberger. Context modeling for active assistance. In Cristina Cabanillas, Sergio España, and Siamak Farshidi, editors, *Proc. of the ER Forum 2017 and the ER 2017 Demo Track co-located with the 36th Int. Conference on Conceptual Modelling (ER 2017)*, pages 221–234, 2017.

[MSNRR16] Pedram Mir Seyed Nazari, Alexander Roth, and Bernhard Rumpe. An Extended Symbol Table Infrastructure to Manage the Composition of Output-Specific Generator Information. In *Modellierung 2016 Conference*, LNI 254, pages 133–140. Bonner Köllen Verlag, March 2016.

[NPR13] Antonio Navarro Pérez and Bernhard Rumpe. Modeling Cloud Architectures as Interactive Systems. In *Model-Driven Engineering for High Performance and Cloud Computing Workshop*, CEUR Workshop Proceedings 1118, pages 15–24, 2013.

[PFR02] Wolfgang Pree, Marcus Fontoura, and Bernhard Rumpe. Product Line Annotations with UML-F. In *Software Product Lines Conference (SPLC'02)*, LNCS 2379, pages 188–197. Springer, 2002.

[PR94] Barbara Paech and Bernhard Rumpe. A new Concept of Refinement used for Behaviour Modelling with Automata. In *Proceedings of the Industrial Benefit of Formal Methods (FME'94)*, LNCS 873, pages 154–174. Springer, 1994.

[PR99] Jan Philipps and Bernhard Rumpe. Refinement of Pipe-and-Filter Architectures. In *Congress on Formal Methods in the Development of Computing System (FM'99)*, LNCS 1708, pages 96–115. Springer, 1999.

[PR01] Jan Philipps and Bernhard Rumpe. Roots of Refactoring. In Kilov, H. and Baclavski, K., editor, *Tenth OOPSLA Workshop on Behavioral Semantics. Tampa Bay, Florida, USA, October 15*. Northeastern University, 2001.

[PR03] Jan Philipps and Bernhard Rumpe. Refactoring of Programs and Specifications. In Kilov, H. and Baclavski, K., editor, *Practical Foundations of Business and System Specifications*, pages 281–297. Kluwer Academic Publishers, 2003.

[Rin14] Jan Oliver Ringert. *Analysis and Synthesis of Interactive Component and Connector Systems*. Aachener Informatik-Berichte, Software Engineering, Band 19. Shaker Verlag, 2014.

[RK96] Bernhard Rumpe and Cornel Klein. Automata Describing Object Behavior. In B. Harvey and H. Kilov, editors, *Object-Oriented Behavioral Specifications*, pages 265–286. Kluwer Academic Publishers, 1996.

[RKB95] Bernhard Rumpe, Cornel Klein, and Manfred Broy. Ein strombasiertes mathematisches Modell verteilter informationsverarbeitender Systeme - Syslab-Systemmodell. Technischer Bericht TUM-I9510, TU München, Deutschland, März 1995.

[RR11] Jan Oliver Ringert and Bernhard Rumpe. A Little Synopsis on Streams, Stream Processing Functions, and State-Based Stream Processing. *International Journal of Software and Informatics*, 2011.

[RRRW15] Jan Oliver Ringert, Alexander Roth, Bernhard Rumpe, and Andreas Wortmann. Language and Code Generator Composition for Model-Driven Engineering of Robotics Component & Connector Systems. *Journal of Software Engineering for Robotics (JOSER)*, 6(1):33–57, 2015.

[RRSW17] Jan Oliver Ringert, Bernhard Rumpe, Christoph Schulze, and Andreas Wortmann. Teaching Agile Model-Driven Engineering for Cyber-Physical Systems. In *International Conference on Software Engineering: Software Engineering and Education Track (ICSE'17)*, pages 127–136. IEEE, May 2017.

[RRW13a] Jan Oliver Ringert, Bernhard Rumpe, and Andreas Wortmann. From Software Architecture Structure and Behavior Modeling to Implementations of Cyber-Physical Systems. In *Software Engineering Workshopband (SE'13)*, LNI 215, pages 155–170, 2013.

[RRW13b] Jan Oliver Ringert, Bernhard Rumpe, and Andreas Wortmann. MontiArcAutomaton: Modeling Architecture and Behavior of Robotic Systems. In *Conference on Robotics and Automation (ICRA'13)*, pages 10–12. IEEE, 2013.

[RRW14] Jan Oliver Ringert, Bernhard Rumpe, and Andreas Wortmann. *Architecture and Behavior Modeling of Cyber-Physical Systems with MontiArcAutomaton*. Aachener Informatik-Berichte, Software Engineering, Band 20. Shaker Verlag, December 2014.

[RSW+15] Bernhard Rumpe, Christoph Schulze, Michael von Wenckstern, Jan Oliver Ringert, and Peter Manhart. Behavioral Compatibility of Simulink Models for Product Line Maintenance and Evolution. In *Software Product Line Conference (SPLC'15)*, pages 141–150. ACM, 2015.

[Rum96] Bernhard Rumpe. *Formale Methodik des Entwurfs verteilter objektorientierter Systeme*. Herbert Utz Verlag Wissenschaft, München, Deutschland, 1996.

[Rum02] Bernhard Rumpe. Executable Modeling with UML - A Vision or a Nightmare? In T. Clark and J. Warmer, editors, *Issues & Trends of Information Technology Management in Contemporary Associations, Seattle*, pages 697–701. Idea Group Publishing, London, 2002.

[Rum03] Bernhard Rumpe. Model-Based Testing of Object-Oriented Systems. In *Symposium on Formal Methods for Components and Objects (FMCO'02)*, LNCS 2852, pages 380–402. Springer, November 2003.

[Rum04] Bernhard Rumpe. Agile Modeling with the UML. In *Workshop on Radical Innovations of Software and Systems Engineering in the Future (RISSEF'02)*, LNCS 2941, pages 297–309. Springer, October 2004.

[Rum11] Bernhard Rumpe. *Modellierung mit UML, 2te Auflage*. Springer Berlin, September 2011.

[Rum12] Bernhard Rumpe. *Agile Modellierung mit UML: Codegenerierung, Testfälle, Refactoring, 2te Auflage*. Springer Berlin, Juni 2012.

[Rum13] Bernhard Rumpe. Towards Model and Language Composition. In Benoit Combemale, Walter Cazzola, and Robert Bertrand France, editors, *Proceedings of the First Workshop on the Globalization of Domain Specific Languages*, pages 4–7. ACM, 2013.

[Rum16] Bernhard Rumpe. *Modeling with UML: Language, Concepts, Methods*. Springer International, July 2016.

[Rum17] Bernhard Rumpe. *Agile Modeling with UML: Code Generation, Testing, Refactoring*. Springer International, May 2017.

[RW18] Bernhard Rumpe and Andreas Wortmann. Abstraction and Refinement in Hierarchically Decomposable and Underspecified CPS-Architectures. In Lohstroh, Marten and Derler, Patricia Sirjani, Marjan, editor, *Principles of Modeling: Essays Dedicated to Edward A. Lee on the Occasion of His 60th Birthday*, LNCS 10760, pages 383–406. Springer, 2018.

[Sch12] Martin Schindler. *Eine Werkzeuginfrastruktur zur agilen Entwicklung mit der UML/P*. Aachener Informatik-Berichte, Software Engineering, Band 11. Shaker Verlag, 2012.

[SHH+20] GÃ¼nther Schuh, Constantin Häfner, Christian Hopmann, Bernhard Rumpe, Matthias Brockmann, Andreas Wortmann, Judith Maibaum, Manuela Dalibor, Pascal Bibow, Patrick Sapel, and Moritz Kröger. Effizientere Produktion mit Digitalen Schatten. *ZWF Zeitschrift für wirtschaftlichen Fabrikbetrieb*, 115(special):105–107, April 2020.

[SM18] Claudia Steinberger and Judith Michael. Towards Cognitive Assisted Living 3.0 (Extended Abstract): Integration of non-smart resources into cognitive assistance systems. *EMISA Forum*, 38(1):35–36, Nov 2018.

[SM20] Claudia Steinberger and Judith Michael. *Using Semantic Markup to Boost Context Awareness for Assistive Systems*, pages 227–246. Computer Communications and Networks. Springer International Publishing, 2020.

[SRVK10] Jonathan Sprinkle, Bernhard Rumpe, Hans Vangheluwe, and Gabor Karsai. Metamodelling: State of the Art and Research Challenges. In *Model-Based Engineering of Embedded Real-Time Systems Workshop (MBEERTS'10)*, LNCS 6100, pages 57–76. Springer, 2010.

[THR+13] Ulrike Thomas, Gerd Hirzinger, Bernhard Rumpe, Christoph Schulze, and Andreas Wortmann. A New Skill Based Robot Programming Language Using UML/P Statecharts. In *Conference on Robotics and Automation (ICRA'13)*, pages 461–466. IEEE, 2013.

[Völ11] Steven Völkel. *Kompositionale Entwicklung domänenspezifischer Sprachen*. Aachener Informatik-Berichte, Software Engineering, Band 9. Shaker Verlag, 2011.

[Wei12] Ingo Weisemöller. *Generierung domänenspezifischer Transformationssprachen*. Aachener Informatik-Berichte, Software Engineering, Band 12. Shaker Verlag, 2012.

[ZPK+11] Massimiliano Zanin, David Perez, Dimitrios S Kolovos, Richard F Paige, Kumardev Chatterjee, Andreas Horst, and Bernhard Rumpe. On Demand Data Analysis and Filtering for Inaccurate Flight Trajectories. In *Proceedings of the SESAR Innovation Days*. EUROCONTROL, 2011.