Eine generische Methode zur optimierten Absicherung und Freigabe von Fahrwerkfunktionen

Von der Fakultät für Ingenieurwissenschaften, Abteilung Maschinenbau und Verfahrenstechnik
der
Universität Duisburg-Essen

zur Erlangung des akademischen Grades
eines
Doktors der Ingenieurswissenschaften
Dr.-Ing.

genehmigte Dissertation

von
Benedikt Sven Jooß
aus
Villingen-Schwenningen

Gutachter: Univ.-Prof. Dr.-Ing. Dr. h.c. Dieter Schramm
Prof. Dr.-Ing. Frank Lobeck

Tag der mündlichen Prüfung: 23.05.2023

Vorwort

Die vorliegende Arbeit wäre ohne die Unterstützung vieler nicht möglich gewesen, daher möchte ich mich an dieser Stelle bei allen Bedanken, die zum Erfolg dieser Arbeit beigetragen haben.

An erster Stelle möchte ich mich bei Prof. Dr.-Ing. Dr. h.c. Dieter Schramm für die wissenschaftliche Betreuung und die Unterstützung bereits im Vorfeld bedanken. So haben seine Ratschläge, die Durchsicht meiner Veröffentlichungen und die gemeinsamen Gespräche zum Erfolg dieser Arbeit geführt. Ebenso gilt mein Dank Prof. Dr.-Ing. Frank Lobeck für die Ausfertigung des Zweitgutachtens und die Unterstützung im Rahmen der mündlichen Prüfung.

Natürlich möchte ich mich auch beim gesamten Team vom Lehrstuhl Mechatronik der Universität Duisburg-Essen für die Unterstützung in allen universitären Fragen und die großartigen Doktorandenseminare bedanken.

Kolleginnen und Kollegen sind bei Industriepromotionen ebenso wichtig wie das Team des Lehrstuhles. Daher gilt mein Dank allen, mit denen ich bei der Dr. Ing. h.c. F. Porsche AG, der Audi AG und der CARIAD SE zusammengearbeitet habe.

Ein großer Dank gilt dabei: Marc Petit, der mir als mein Vorgesetzter bei der PAG ausreichend vertrauen schenkte, um diese Arbeit möglich zu machen. Dr.-Ing. Jan-Maximilian Montenbruck für die wissenschaftliche Beratung und die Übernahme bei CARIAD. Ruža Gröbe für die vielen fachlichen und überfachlichen Gespräche sowie die gemeinsamen Dienstreisen nach Ingolstadt. Und Thomas Satzinger für sein breites Wissen zum Thema Softwaretest. Aber auch allen Kollegen der ehemaligen Abteilung „EFM“ möchte ich danken – ohne euch wäre diese Arbeit nicht möglich gewesen.

Lernen konnte ich mindestens genauso viel außerhalb meiner direkten Arbeit. Daher möchte ich mich bei allen Kollegen vom Porsche Doktoranden Netzwerk, insbesondere der AG-B für den überfachlichen Austausch, die vielen Erinnerungen und gemeinsamen Erlebnisse z. B. beim Hüttenwochenende oder der Exkursion zu Bentley bedanken.

In schwierigen Momenten konnte ich mich immer an meine Familie und Freunde wenden. Daher danke ich euch dafür, dass ihr mich in meinen Vorhaben stets unterstützt und motiviert habt. Vor allem möchte ich meiner Mutter, Dagmar Jooß, für die Durchsicht danken.

Schließlich gilt mein Dank Dr. rer. nat. Julian Schöllkopf für die wissenschaftlichen Ratschläge, den Mut, die Aufmunterungen und die Geduld an schlechten Tagen.

Auch allen, die ich nicht namentlich erwähnen konnte, gilt selbstverständlich mein Dank!

Kurzfassung

Der Wandel, den die Automobilindustrie momentan durchlebt, führt zu vielen Veränderungen in der Fahrzeugentwicklung. Die meisten neuen Fahrzeugfunktionen basieren auf Software, was die Etablierung von neuen agilen Vorgehensweisen nach sich zieht. Davon ist auch der Prozessschritt der Absicherung und Freigabe betroffen. Hinzu kommt eine gestiegene Komplexität durch neue technologische Innovationen sowie durch neue Normen und Gesetze, die die Entwicklung beeinflussen.

Um diesen Herausforderungen zu begegnen, wurde in dieser Arbeit untersucht, wie eine generische Methode zur Absicherung und Freigabe von Fahrwerkfunktionen dargestellt werden kann. Mit der standardisierten Methode können Funktionen mit verschiedenen technischen Merkmalen getestet und freigegeben werden, um eine einheitliche Qualität zu gewährleisten. Fundament der Methode sind Ergebnisse hermeneutischer Untersuchungen gesetzlicher und normativer Anforderungen. Zunächst wurden besondere Merkmale definiert, welche eine technische Eigenschaft von SW beschreiben oder durch die Software gewährleistet werden. Parallel wurden Testmodule entwickelt. Jedes Testmodul entspricht einer Testaktivität und beschreibt generisch u. a. Testziele, Testendekriterien und Testentwurfsverfahren. Mittels hermeneutischer Untersuchungen von Gesetzen und Normen wurden für jedes besondere Merkmal Testmodule bestimmt, welche auf verschiedenen Ebenen durchzuführen sind. Daraus ergibt sich ein modularer Testbaukasten, welcher während der Testplanung zum Einsatz kommt und im Vergleich zu anderen Methoden für eine Vielzahl unterschiedlicher Funktionen anwendbar ist.

Weiterhin lag ein besonderes Augenmerk der vorliegenden Arbeit auf der Entwicklung eines generischen Freigabeverfahrens mit breitem Anwendungsspektrum innerhalb der Fahrwerkfunktionsentwicklung. Um effizient Freigaben erteilen zu können, basiert die Methode auf der Erreichung von Testendekriterien für verschiedene Freigabelevel. Dabei werden Risiken identifiziert, gegeneinander gewichtet und mit Eintrittswahrscheinlichkeiten pro Freigabelevel in Verbindung gebracht, um einen Eintrittsfaktor zu erhalten. Für jedes besondere Merkmal wird ein Gefahrenniveau systematisch ermittelt, um aus diesem in Zusammenhang mit dem Eintrittsfaktor das Testendekriterium zu bestimmen.

Da es in der Praxis häufig zu Terminkonflikten kommt, wird die Methode mit Vorschlägen zur Testfallpriorisierung abgerundet. Dazu werden verschiedene Kriterien herausgearbeitet, anhand derer die Priorisierung von Testfällen hinsichtlich Zeitmangels abgeleitet werden kann.

Abstract

The transformation that the automotive industry is currently undergoing is leading to many changes in vehicle development. Most new vehicle functions are based on software, which leads to the establishment of agile development procedures. This also affects the development steps of validation and release. In addition, there is an increased complexity due to new technological innovations as well as new standards and laws that influence the development.

In order to meet these challenges, in this thesis an investigation of a generic method for validation and release of chassis functions is presented. With a standardised method the quality of test an release processes for functions with different technical characteristics is guaranteed. The method is based on the results of hermeneutic investigations of legal and normative requirements. First, special features were defined that describe a technical characteristic of software or that must be guaranteed by the software. In parallel, test modules were developed. Each test module corresponds to a test activity and generically describes, among other things, test objectives, test end criteria and test design procedures. By means of hermeneutic studies of laws and standards, test modules were determined for each particular special feature. Foe each feature the modules have to be carried out at different test levels. This results in a modular test kit, which is used during test planning. Compared to other test strategies this method is designed to be applied to a large number of different technical functions of the vehicle chassis.

Furthermore, a special focus of the present work was on the development of a generic release procedure with a broad application spectrum within the chassis function development. In order to be able to issue releases efficiently, the method is based on the achievement of test exit criteria for different release levels. Risks are identified, weighted against each other and associated with probabilities of occurrence per release level to obtain an occurrence factor. For each particular characteristic, a hazard level is systematically determined in order to determine the test exit criterion from this in connection with the occurrence factor.

As there are often scheduling conflicts in practice, the method is rounded off with suggestions for test case prioritisation. For this purpose, various criteria are worked out. On the basis of these the prioritisation of test cases can be derived with regard to lack of time.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Abkürzung	Bedeutung
ABS	Antiblockiersystem
AHP	Analytic Hierachy Process
ALM	Application Lifecycle Management
ASIL	Automotive Safety Integrity Level
ASPICE	Automotive Software Process Improvement and Capability Determination
ASTM	Automotive Security Testing Methodik
BEV	Battery Electric Vehicle; dt. Elektroauto
BM F	Besonderes Merkmal funktionale Anforderungen
BM S	Besonderes Merkmal Sicherheitsanforderungen
BM Z	Besonderes Merkmal Zulassungsrelevanz
BsM	Besonderes Merkmal
BST	Baustufe
CARB	California Air Resources Board
CS	Cybersecurity
DF	Design Freeze
ESP	Elektronisches Stabilitätsprogramm; elektronische Stabilitätskontrolle
EWG	Europäische Wirtschaftsgemeinschaft; heute: Europäische Union
EU	Europäische Union
FAS	Fahrerassistenzsysteme
FaaS	Function as a Service
FMS	Freigabemeilenstein
FuSi	Funktionale Sicherheit
HCP	High Computing Platform
ISTQB	International Software Testing Qualification Board
MC/DC	Modifizierter Bedingungs-/Entscheidungstest
MTB	Modularer Testbaukasten
NHTSA	National Highway Traffic Safety Administration Cybersecurity Best Practices for the Safety of Modern Vehicles
OEM	Original Equipment Manufacturer
PI	Product Increment
PKW	Personenkraftwagen
PO	Product Owner

Abkürzung	Bedeutung
PVS	Produktionsvorserie
QS	Qualitätssicherung
RTE	Run Time Environment
RUT	Ressourcennutzungstest
SAFe	Scaled Agile Framework
SG	Steuergerät
SM	Scrum Master
SOP	Start of Production
SOTIF	Safety of the Intended Function
SW	Software
SWC	Softwarekomponente
SWS	Softwaresystem
SWU	Softwareunit; Synonym auch Softwaremodul
TS	Testspezifikation
USP	Unique Selling Proposition; dt. Alleinstellungsmerkmal
VDA	Verband der Automobilindustrie

1 Softwaretesting in der Fahrwerkentwicklung

Fahrzeuge sind heutzutage digitale Produkte ähnlich wie Smartphones und Computer. Die Zunahme von Software (SW) basierten Funktionen in Fahrzeugen führt zu Veränderungen in der Fahrzeugentwicklung. In diesem Kapitel wird deutlich gemacht, vor welchen Herausforderungen die Entwicklungsbereiche hinsichtlich der Absicherung von SW stehen und welche Zielsetzung diese Arbeit verfolgt. Es folgt ein Überblick über den Aufbau der Arbeit.

1.1 Motivation

Mechatronische Regelsysteme zählen spätestens seit der Einführung der elektronischen Stabilitätskontrolle (ESP) in der Mercedes-Benz A-Klasse Mitte der 1990er Jahre zum Standard in der Fahrwerktechnik. Heutzutage werden solche Funktionen häufig dafür genutzt, einen Wettbewerbsvorteil durch Differenzierung und Innovationen zu erzielen. Viele der SW basierten Funktionen, die der Domäne „Fahrwerk“ zugeordnet werden, sind sowohl für ein sicheres Fahrverhalten als auch für komfortables oder sportliches Fahrempfinden verantwortlich. Einen wesentlichen Anteil an der Funktionalität solcher mechatronischen Systeme hat die Software. Die Software ermöglicht es mit exakt demselben Hardwarelayout unterschiedliche Fahrverhalten darzustellen und so Fahrzeuge voneinander zu differenzieren. Die Kommunikation zwischen den Systemen ist in modernen Fahrzeugen essenziell. Eine fahrzeugweite Vernetzung zwischen allen Steuergeräten muss gewährleistet sein.

Da die SW einen wesentlichen Einfluss auf das Fahrzeug, die Fahrzeugsicherheit und letztendlich auch die Qualität hat, sind SW-Tests unabdingbar. Ziel ist es, so früh wie möglich Fehler in der Software zu finden. Allerdings ist es unter SW-Testern allgemein akzeptiert, dass es unmöglich ist, alle Fehler in einer komplexen Software zu identifizieren und zu eliminieren (ATB et al. 2020). Trotzdem wird das Ziel verfolgt, eine hohe SW-Qualität zu erlangen, ohne dabei Kosten und Projektpläne aus den Augen zu verlieren. Hinzu kommen vielseitige nicht funktionale Anforderungen an die Software und die Entwicklungsprozesse. So müssen beispielsweise verschiedene Gesetze oder Normen erfüllt, funktionale Anforderungen an Softwarekomponenten (SWC) und Vernetzung berücksichtigt und neue Technologien entwickelt werden.

Um all diesen Anforderungen gerecht zu werden, setzt sich die SW-Entwicklung aus verschiedenen Prozessen zusammen. Im Bereich des SW-Tests sind diese ebenso notwendig wie beispielsweise im Anforderungs- oder Fehlermanagement. Die Strategie, auf die die Testdurchführung gestützt ist, muss eine notwendige Flexibilität ermöglichen, ohne zu große Abweichungen zuzulassen. In der Wissenschaft und Industrie gibt es unzählige Strategien und

Methoden mit unterschiedlichsten Vor- und Nachteilen. Ein Großteil dieser Strategien ist jedoch für einen sehr spezifischen Teil der SW, beispielsweise funktionale Sicherheit (FuSi), konzipiert. Wenige dieser Strategien betrachten ein größeres, vollumfängliches Bild der SW-Entwicklung. In dieser Arbeit wird ein vollumfänglicher Ansatz einer generischen Methode für die Absicherung und Freigabe von Fahrwerkfunktionen mit verschiedenen technischen Merkmalen erarbeitet.

1.1.1 Betrachtungsgegenstand

Eine Teststrategie kann nur effizient gestaltet werden, wenn das Testobjekt bekannt ist. Da in dieser Arbeit eine generische Methode für die Absicherung und Freigabe von Fahrwerkfunktionen erarbeitet werden soll, wird an dieser Stelle erörtert, was unter einer Fahrwerkfunktion im Zusammenhang mit der Arbeit verstanden wird.

Prinzipiell wird unter einer Funktion der Softwareanteil eines mechatronischen Systems, welches dem Fahrwerk zugeordnet ist, verstanden. Diese Fahrwerkfunktionen können sowohl die Längs-, Vertikal- und/oder die Querdynamik beeinflussen. Software kann verschiedene Bedeutungen haben. Für diese Arbeit wird darunter der Anteil verstanden, der aus modellbasierter Entwicklung hervorgeht. Dies bedeutet, dass sowohl das Modell (i. S. v. Logik) als auch der daraus generierte Code Objekt der Untersuchung sind. Für höhere Testebenen werden auch Basissoftware und Run-Time-Environment benötigt. Diese werden jedoch der Testumgebung einzelner Prüfinstanzen zugezählt und bilden damit nicht das Untersuchungsobjekt ab.

Ein mechatronisches System besteht zusätzlich zur Software auch aus Hardwareelementen und Elektronik. Der Begriff „Mechatronik“ (bzw. die englische Übersetzung „mechatronics“) wurde Ende der 1960er Jahr durch den Japaner Ko Kikuchi geprägt und seither mehrmals auf unterschiedliche Weise definiert. Die Definitionen sind unterschiedlich weit gefasst. Allen gemein ist jedoch der Standpunkt, dass mechatronische Systeme das Zusammenspiel mehrerer klassischer Ingenieurwissenschaften sind. Es handelt sich um die Disziplinen Maschinenbau, Elektrotechnik und der Informationstechnik wie in Abbildung 1.1 dargestellt. (Isermann 2008; Krause et al. 2007; VDI Richtlinie VDI 2206)

Die Definition nach Isermann (2008) ist die gebräuchlichste und dient auch als Grundlage für die VDI-Richtlinie VDI 2206 (VDI Richtlinie VDI 2206). Durch die Kombination der drei Fachdisziplinen entstehen oftmals komplexe Systeme. Ziel ist meist die Optimierung eines bestehenden mechanischen oder elektromechanischen Systems. Die Aufgabe eines solchen Systems wird nicht ausschließlich mechanisch, sondern zusätzlich digital-elektronisch gelöst. Zur Erfüllung der Aufgabe müssen die Lösungswege aufeinander abgestimmt sein. Nur durch eine optimale und permanente Interaktion der einzelnen Komponenten kann ein System funktionieren. Um dies zu erreichen, gibt es unterschiedliche Ansätze. Alle Ansätze stützen sich auf mathematische Modelle. Durch ein Zusammenspiel aus Bauteilen (im weiteren Sinne) aus

allen drei Fachdisziplinen kann ein solches mathematisches Modell in ein funktionierendes System überführt werden. Die Kraftübertragung oder die Verstellung von Aktuatoren wird von der Mechanik übernommen. Die Elektrik ist für die Energieflüsse zuständig und die Informationstechnik für die Steuerung oder Regelung (abhängig von den Systemanforderungen). (Isermann 2008; Schäppi et al. 2005)

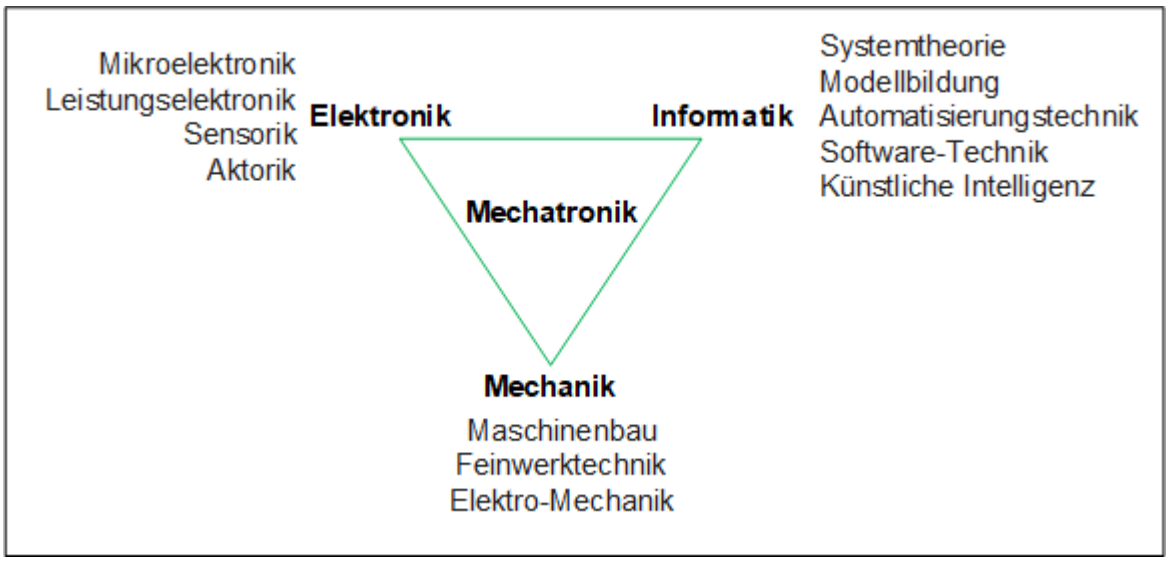

Abbildung 1.1: Mechatronik als interdisziplinäres System (modifiziert nach Isermann 2008)

Ein weiteres wesentliches Merkmal mechatronischer Systeme im Vergleich zu konventionellen Systemen besteht in der Entwicklungsarbeit. In konventionellen Systemen werden elektronische Anteile, falls vorhanden, in das mechanische System räumlich integriert und weitestgehend getrennt voneinander entwickelt. Bei mechatronischen Systemen sind die Disziplinen hingegen „[…]von Anfang an als räumlich und funktionell integriertes Gesamtsystem zu betrachten […] (Isermann 2008). Daher werden die einzelnen Bestandteile parallel entwickelt, was als „simultaneous engineering“ bezeichnet wird. Nur so können laut Isermann die notwendigen Synergieeffekte erzielt werden.

Der Fokus dieser Arbeit liegt auf dem Informatik-Anteil mechatronischer Systeme für Fahrzeuge. Im gängigen Sprachgebrauch wird dies meist als Softwareentwicklung bezeichnet. Der Schwerpunkt liegt auf den Entwicklungsteilaufgaben „Absicherung“ und „Freigabe“ von Fahrwerkfunktionen. Diese Systeme haben speziell in sportlichen Fahrzeugen eine zweigeteilte Ausprägung. Einerseits dienen sie der Fahrsicherheit und sind darauf ausgelegt Fahrzeuge zu stabilisieren, andererseits gibt es Regelsysteme, die der Agilisierung von Fahrzeugen dienen. Aus diesen unterschiedlichen Ansprüchen leitet sich ein sehr breites Spektrum an Anforderungen an die einzelnen Funktionen ab. Diese Anforderungen dienen als Grundlage für Tests, beispielsweise auf verschiedenen Ebenen im V-Modell (Kapitel 3).

Ein Softwaretest ist ein Werkzeug zur Verifikation oder Validierung von Software und bestätigt, dass diese für den bestimmungsgemäßen Gebrauch anwendbar ist. D. h. dass Feh-

lerzustände aufgedeckt und Erkenntnisse über das Testobjekt gesammelt werden. Die gewonnenen Erkenntnisse dienen dazu, die Fehler zu beseitigen und den Reifegrad der SW zu bestätigen. Darüber hinaus können die Informationen der Entscheidungsfindung im Unternehmen dienen. (ATB et al. 2020)

Unter Testing wird grundsätzlich jedes Mittel verstanden, welches dem Nachweis von Fehlern oder dem Nachweis der Einhaltung von Anforderungen dient. In der Softwareentwicklung wird davon ausgegangen, dass die Software zu komplex ist, um die Fehlerfreiheit zu beweisen. Aus diesem Grund wird durch Testing nachgewiesen, dass der bestimmungsgemäße Gebrauch der Systeme frei von Gefahren für den Anwender oder Dritte ist (ATB et al. 2020). Neben dem bestimmungsgemäßen Gebrauch werden auch fahrdynamische Extremsituationen, beispielsweise ein doppelter Spurwechsel oder Misuse, beispielsweise schnelles Überfahren von Schwellen berücksichtigt. Um einen solchen Nachweis erbringen zu können, wird ein vielschichtiger Testprozess durchlaufen. Der Prozess ist meist so gestaltet, dass die Software auf Basis der bestehenden Entwicklungsprozesse und Vorgehensmodelle in möglichst kleinen Einheiten sukzessive getestet wird, um anschließend in immer größere Konstrukte integriert und getestet zu werden. In der modellbasierten Entwicklung von Software wird zunächst das Modell per Model-in-the-Loop (MiL) Test getestet. Sobald keine Auffälligkeiten im Modell vorhanden sind, wird aus dem Modell die Software kompiliert und mittels Software-in-the-Loop (SiL) getestet. Häufig wird anschließend die Software auf dem Target, also der Zielhardware getestet. Es wird zwischen Processor-in-the-Loop (PiL) Tests, Hardware-in-the-Loop (HiL) Test und Fahrzeugtests unterschieden. Ein wesentlicher Unterschied zwischen den einzelnen Testebenen ist der Grad der Simulation. Während beim PiL-Test nur der Zielprozessor verwendet wird und die restliche Umgebung simuliert ist, wird bei HiL-Tests das gesamte Steuergerät (SG) in eine ansonsten simulierte Umgebung integriert. Die Testziele der einzelnen Ebenen unterscheiden sich ebenfalls. Um diese Komplexität zu beherrschen, werden Strategien erarbeitet, die Testziele, Testebenen und den Einsatz von Testinstanzen zu planen und dokumentieren. Solche Strategien müssen für jeden Anwendungsfall bzw. jedes Unternehmen oder eine Unternehmenseinheit individuell angepasst werden. Dadurch gibt es eine Vielzahl an Strategien, die so gestaltet sind, dass ein bestimmtes Testziel mit einem Minimum an Testaufwand erreicht wird. Die meisten dieser Strategien verfolgen das Ziel, eine bestimmte Anforderung effizient zu testen, beispielsweise für Funktionen mit hoher Sicherheitskritikalität (engl. Safety) oder besonderen Anforderungen hinsichtlich IT-Sicherheit (engl. Security).

Im Anschluss an erfolgreich abgeschlossene Tests werden Softwarefunktionen freigegeben. Dies geschieht auf der Grundlage der durchgeführten Tests. Mit einer Freigabe wird bestätigt, dass das entwickelte Objekt (SW, Bauteile oder Systeme) den Anforderungen entspricht und für einen im Vorfeld geplanten Verwendungszweck eingesetzt werden kann. Mit einer Freigabe wird dies durch den Entwickler bestätigt. Bereits während der Planung werden

Meilensteine des Entwicklungsprozesses festgelegt, zu denen definierte Entwicklungsziele erreicht werden sollen. Zu diesen Entwicklungszielen werden zudem Freigabelevel definiert. Diese Level geben Auskunft darüber, wie die Softwareeinheit eingesetzt werden kann und welche Einschränkungen bestehen. Am Ende des Entwicklungsprozesses muss die Software für den Endkunden gefahrlos und ohne Einschränkungen verwendbar sein. Die Bestätigung der einzelnen Freigabelevel erfolgt über die Erreichung der zuvor definierten Testziele.

1.1.2 Gesetzliche Rahmenbedingungen und Normen

Um die Sicherheit und die notwendige Funktionalität von Software basierten Fahrwerksystemen für den Endverbraucher zu garantieren, ist die Entwicklung von Automobilen auch durch den Gesetzgeber reguliert. Durch den Einzug von Elektronik und Software wird der Entwicklungsprozess durch neue Normen und Gesetze geprägt und reglementiert. Zudem haben Umweltschutz, Fahrzeugsicherheit und assistiertes Fahren eine immer größere Bedeutung, die zu steigenden Anzahlen unterschiedlichster Vorschriften führen. Längst sind es nicht nur die EU und die USA, welche strenge Vorschriften z. B. für Grenzwerte des Abgasausstoßes fordern, sondern auch Länder wie China, Süd-Korea oder Mexiko. All diese Veränderung führen zu einer Vielzahl an weltweiten Normen und Gesetzen, die im Entwicklungsprozess berücksichtig werden müssen.

Am Beispiel der europäischen Verordnungen zum Ausstoß von Emissionen, umgangssprachlich Abgasnorm genannt, lässt sich zudem erkennen, dass nicht nur die Anzahl der Gesetze zunimmt, sondern ebenfalls deren Komplexität und Umfang steigt. 1977 wurde durch die Europäische Wirtschaftsgemeinschaft (EWG; heute Europäische Union (EU)) die Richtlinie 70/220/EWG mit „Maßnahmen gegen die Verunreinigung der Luft durch Emissionen von Kraftfahrzeugen“ veröffentlicht. Bei deren Einführung umfasste die Norm 22 Seiten. Wie in Abbildung 1.2 zu sehen, umfasst ihre letzte Ausbaustufe (in Kraft von Januar 2007 bis 2013) 194 Seiten. Mitte 2007 wurde die Richtlinie 70/220/EWG durch die VERORDNUNG (EG) Nr. 715/2007 mit 16 Seiten abgelöst. Im Jahr 2020 umfasst diese Verordnung durch Erweiterungen und mehr als zehn ergänzende Verordnungen über 2.000 Seiten allein für die Zulassung von PKW. (Jooß und Schramm 2022)

Die Gründe für die steigende Anzahl an Normen und deren zunehmende Komplexität sind vielfältig. Einerseits müssen eine steigende Anzahl länderspezifische Normen berücksichtigt werden, da frühere Schwellenländer sich weiter entwickeln und daher eigene Gesetze zu Kraftfahrzeugsicherheit, Umweltschutz oder Entwicklungsarbeit veröffentlichen. Andererseits kann man die Begründung im technischen Fortschritt sehen. Dies soll mittels zweier Beispiel verdeutlicht werden.

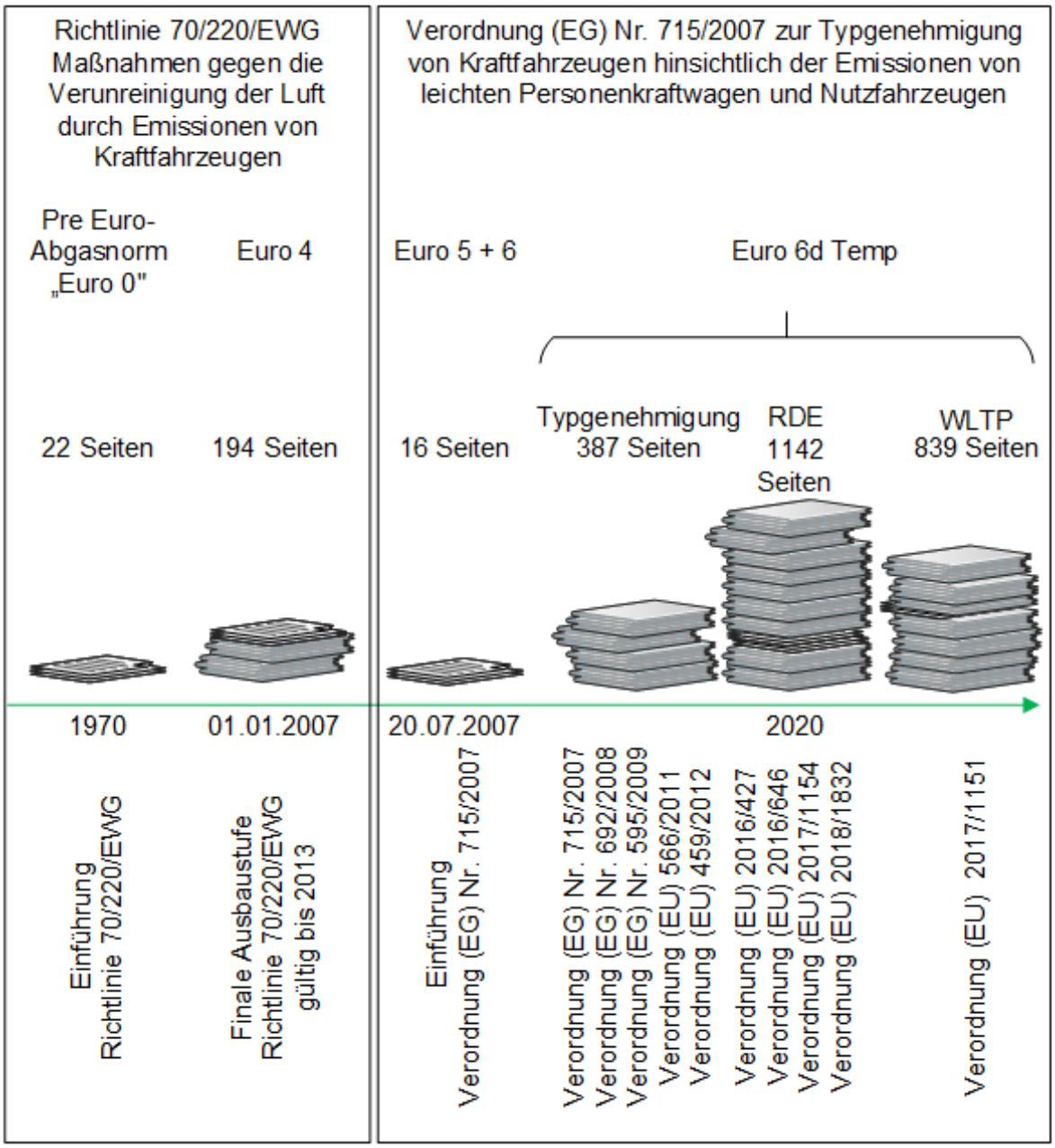

Abbildung 1.2: Steigende Komplexität von Vorschriften am Beispiel der EU "Abgasnorm" (modifiziert nach Jooß und Schramm 2022)

Erstens: Die Wiener Straßenverkehrskonvention vom 8. November 1968

Die Wiener Straßenverkehrskonvention schreibt vor, dass ein fahrendes Fahrzeug jederzeit einen Fahrzeugführer haben muss (Art. 8 Abs. 1), der während der Fahrt die Kontrolle über das Fahrzeug innehaben muss (Art. 8 Abs. 5). Gemäß Art. 13 Abs. 1 ist der Fahrzeugführer dazu verpflichtet, „unter allen Umständen sein Fahrzeug beherrschen, um den Sorgfaltspflichten genügen zu können und um ständig in der Lage zu sein, alle ihm obliegenden Fahrbewegungen auszuführen“ (Bundesgesetzblatt Teil II 1977 Nr.39).

Zur Zeit des Inkrafttretens der Konvention waren autonom fahrende Fahrzeuge technisch undenkbar. Über 50 Jahre später sind die technischen Möglichkeiten jedoch so weit, dass Fahrzeuge zumindest teilweise ohne Eingreifen eines Fahrzeugführers fahren können. Um den technischen Fortschritt nicht zu blockieren und die Fahrzeugsicherheit beispielsweise

durch Fahrerassistenzsysteme (FAS) zu erhöhen, ist es notwendig, die Gesetzgebung entsprechend anzupassen.

Zweitens: Die Kalifornische OBD-Gesetzgebung

Final Regulation Order §1968.2. of title 13 der kalifornischen Regierungskommission „California Air Resources Board (CARB)“ ist ein 2004 erschienenes Gesetz zur Diagnose von Motorsteuergeräten und dem Umgang im Fehlerfall. Zu diesem Zeitpunkt galten Hybridfahrzeuge wie z. B. der Toyota Prius als innovativ und umweltschonend. An reinen Elektroautos (BEVs) wurden zwar auch in dieser Zeit geforscht, allerdings gab es noch keine Großserienfahrzeuge wie z. B. das Tesla Modell S (ab 2013 auf dem deutschen Markt (AutoBild 2022b)) oder den Renault Zoe (ab 2012 auf dem deutschen Markt (AutoBild 2022a)). Daher wurde diese Gesetzgebung rein für Fahrzeuge mit Verbrennungsmotor ausgelegt. In Deutschland waren am 01. April 2022 ca. 687.200 BEVs und 623.000 Plug-in-Hybride (PHEV) zugelassen (Kords 2022b). Dies entspricht jeweils einem Marktanteil von 1,3 %. Im Vergleich waren im Jahr 2017 lediglich 0,07 % aller zugelassenen Fahrzeuge BEVs und 0,05 PHEVs (Kords 2022a). Da es für diese Art von Antrieb momentan kein gleichwertiges Gesetz gibt, auf dem viele weitere Gesetze innerhalb und außerhalb der USA basieren, ist auch hier eine Anpassung bzw. Erweiterung notwendig.

Diese Veränderung an Anzahl und Komplexität der Normen und Gesetze führt letztendlich zu angepassten Entwicklungsprozessen. Um die Typgenehmigung in einem Land zu erhalten, müssen die in diesem Land gültigen Gesetze bereits während der Entwicklung der Fahrzeuge berücksichtigt werden. Viele dieser Gesetze betreffen Software. Hintergrund sind die technologischen Möglichkeiten durch Software im Vergleich zu rein mechanischen Lösungen. In vielen dieser Gesetze sind auch Anforderungen an das Testen der Funktionen beschrieben. Bekannteste Beispiele sind in der Automobilbranche sicherlich die Normreihe ISO 26262-x:201x („Road vehicles – Functional safety“) und die Normreihe ISO/IEC/IEEE 29119-x:201x (Software and systems engineering - Software testing). Letztere beschäftigt sich ausschließlich mit dem Testen von Software.

1.1.3 Technologien

Die Normen und Gesetze sind bei der Entwicklung der in Abschnitt 1.1.1 beschriebenen mechatronischen Systeme zu berücksichtigen. Prinzipiell folgen diese Systeme einem schematisch gleichen Aufbau. Trotzdem haben die Systeme unterschiedliche Aufgaben, Funktionen und technische Eigenschaften. Im Rahmen dieser Arbeit wird vornehmlich der SW-Anteil des mechatronischen Systems adressiert.

Abbildung 1.3 zeigt, wie die Anzahl an mechatronischen Systemen im Fahrzeug zugenommen hat (blaue Linie). Ein Innovationstreiber sind Steuergeräte mit immer besseren Rechenleistungen, sodass die steigende Anzahl an Funktionen auf immer weniger Steuergeräten

verortet werden (grüne Linie). Zu sehen ist nicht nur die Zunahme, sondern ebenfalls die Vielfalt an verschiedenen Funktionen. Zu den Standardfunktionen wie z. B. ESP und dem Antiblockiersystem (ABS) kommen viele wettbewerbsdifferenzierende Systeme. Da diese Funktionen zudem unterschiedliche Anforderungen an deren Steuerung oder Regelung haben, sind damit auch die Struktur, der Aufbau und insbesondere der Inhalt der SW sehr unterschiedlich.

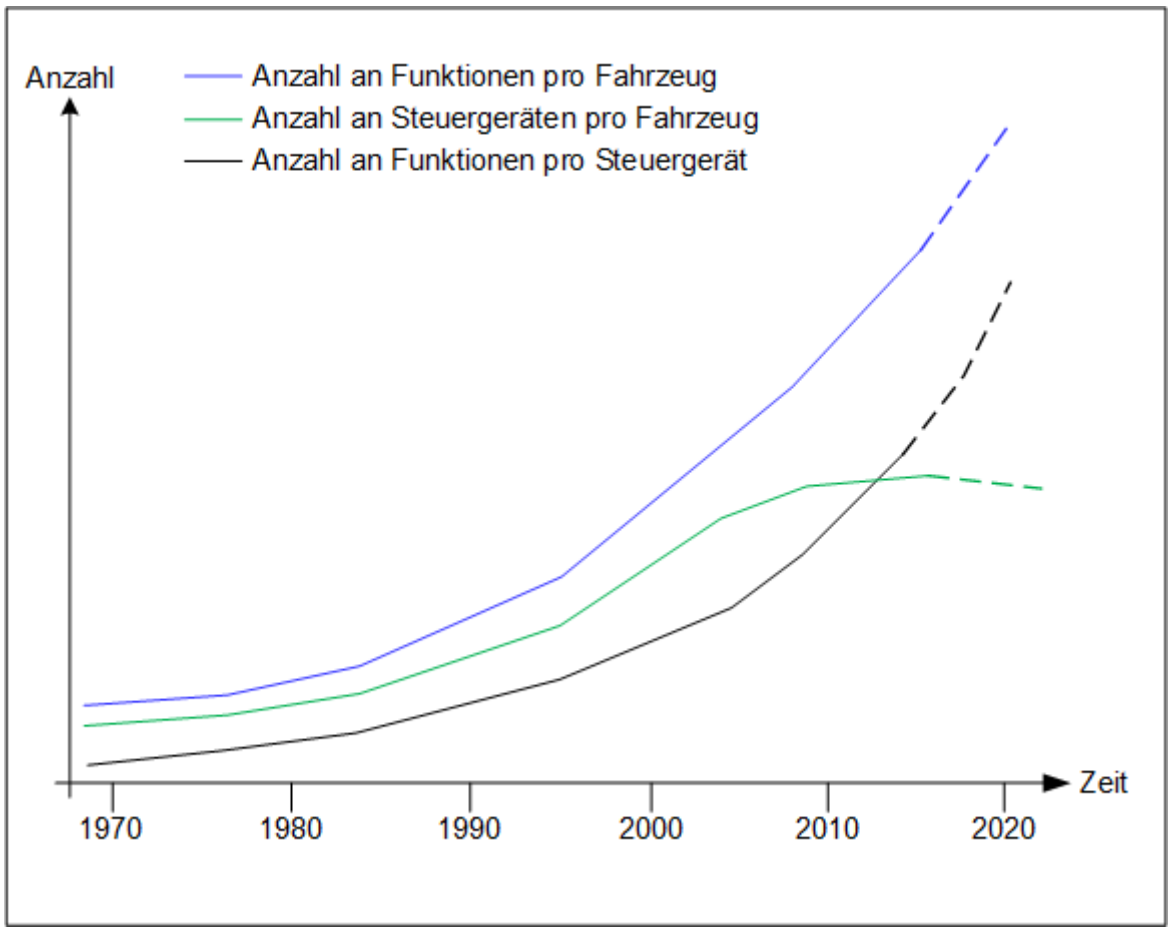

Abbildung 1.3: Mechatronische Systeme über die Zeit (modifiziert nach Schäuffele und Zurawka 2013)

Der Code in Fahrzeugen verdoppelt sich im Durchschnitt alle 42 Monate (Hatton et al. 2017). In einem aktuellen VW Golf 8 sind bis zu 100 Mio. Zeilen SW-Code (Simon 2019). Dies ist unter anderem auch auf die aktuellen Trends im Fahrzeugbau zurückzuführen. So spielt sowohl bei modernen Fahrerassistenzsystemen als auch für das Thema Digitalisierung die HW eine untergeordnete Rolle. Der zunehmende Umfang an SW in den Fahrzeugen bringt jedoch neue Herausforderungen mit sich. So mussten im Jahr 2020 knapp 7 % der Fahrzeuge aufgrund von Softwarefehlern zurückgerufen werden (Center of Automotive Management (CAM) 2021). Sowohl die Anzahl an Rückrufaktionen als auch die Anzahl an SW-Fehler haben eine steigende Tendenz.

Mit der Absicherung von HW haben die Fahrzeughersteller jahrzehntelange Erfahrung und dadurch etablierte Mechanismen. Software ist im direkten Vergleich für viele Fahrzeughersteller ein neues Entwicklungsobjekt. Vor allem durch die unterschiedlichen Arten von SW-

Produkten und die steigende Komplexität sind die Absicherungsmaßnahmen vielfältig und nicht immer vollumfänglich in den kurzen Entwicklungszyklen durchführbar.

Zusätzlich hat die Vernetzung einzelner Steuergeräte einen wesentlichen Einfluss auf die Absicherungsstrategie. So hatten die ersten mechatronischen Systeme ein Steuergerät, auf dem die gesamte SW für ein System installiert war. Dadurch konnten oftmals Lieferanten ein System entwickeln und absichern. Durch die Zunahme der SW-Funktionen und den Druck, das Gewicht der Fahrzeuge zu senken, werden zunehmend wenige, dafür komplexe Steuergeräte, im Volkswagen-Konzern High-Computing-Plattform (HCP) genannt, entwickelt. Auf diesen Steuergeräten sind viele Funktionen gleichzeitig verortet, sodass SW von mehreren Lieferanten oder Herstellern parallel auf dem Steuergerät funktionieren. Durch diese Entwicklung müssen die einzelnen Absicherungsmaßnahmen aufeinander abgestimmt sein, um die Korrektheit der Vernetzung zu gewährleisten.

Eine inkorrekte Interaktion der Funktionen kann zu gefährlichen Fahrsituationen führen, da die einzelnen Systeme gegeneinander wirken und so die Stabilisierung des Fahrzeuges verhindern. Deshalb müssen verschiedene Anforderungen an die SW zeitgleich erfüllt und abgesichert werden, beispielsweise Sicherheitsanforderungen im Sinne von Funktionssicherheit (engl. Safety) einerseits und IT-Sicherheit (engl. Security) andererseits sowie funktionale Anforderungen, z. B. Performance der SW oder gesetzliche Anforderungen, z. B. Auswirkungen auf die Abgasreinigung. Hinzukommen neue Technologien, die ein Alleinstellungsmerkmal (USP) gegenüber dem Kunden bieten. Dazu zählt beispielsweise die Fähigkeit, einzelne Funktionen Over-the-Air upzudaten, ohne einen negativen Einfluss auf Security oder Safety zu haben.

Um eine vollständige Absicherung und damit ein unkritisches Fahrverhalten gewährleisten zu können, ist eine Absicherungsstrategie notwendig. Diese ist vergleichbar mit Testkatalogen zur Prüfung von HW-Bauteilen. Da die SW-Technologien eine hohe Diversität aufweisen, müssen die Maßnahmen anwendungsfallspezifisch eingesetzt werden. Diese Diversität spiegelt sich in den Absicherungsmaßnahmen wieder. Diese müssen zielgerichtet und effizient eingesetzt werden, um bei steigender SW-Komplexität, neuen Technologien und steigende Anzahl an Anforderungen Fehler adäquat in der SW zu detektieren.

1.2 Literaturübersicht

Wegen der in Abschnitt 1.1.3 beschriebenen Zunahme an SW-basierten Funktionen im Automobilumfeld wurden in den vergangenen Jahren einige Arbeiten zum Thema Testen von mechatronischen Systemen verfasst. Dabei zielen alle Arbeiten auf die Fahrzeugsicherheit und Komplexitätsbeherrschung ab. In entwickelnden Abteilungen werden für diesen Zweck Teststrategien erstellt, welche auf Empfehlungen aus der Literatur aufbauen. So definiert Witte (2020) eine Teststrategie als „die Voraussetzung für planmäßige Handlung und die Grundlage

zur Definition der richtigen Testprozesse und Festlegung der Organisation“. Er hält weiterhin fest, dass solch eine Teststrategie für das gesamte Unternehmen oder mindestens gesamte Abteilungen Gültigkeit besitzt. Eine projektspezifische Strategie sieht Witte hingegen nur mit Begründung als hinnehmbar an.

Diesen Umstand berücksichtigen auch die meisten der Arbeiten und konzentrieren sich vermehrt auf die Absicherung einzelner technischer Eigenschaften. Ring (2019) beschreibt in seiner Arbeit eine Teststrategie zum systematischen Testen von Security Anforderungen im Kraftfahrzeug. Damit betrachtet er den Gesamtverbund Fahrzeug, was Rückschlüsse auf eine unternehmensweite Gültigkeit zulässt. Die Arbeit baut darauf auf, dass Fahrzeuge durch ein verpflichtendes Notrufsystem seit 2018 nicht mehr als abgeschlossenes System, sondern als Teil der kritischen Infrastruktur angesehen werden müssen. Da die Funktion solch einer kritischen Infrastruktur erhalten werden muss, stellt Ring in seiner Arbeit eine Testmethodik mit dem Ziel der Vergleichbarkeit des Security-Niveaus von Kraftfahrzeugen vor. Dazu wurden eine sogenannte Automotive Security Testing Methodik (ASTM) entwickelt. Diese umfasst neben Methoden zum Testen auch ein Prozess, bestehend aus fünf Phasen. Die fünf Phasen sind vergleichbar mit den Phasen des fundamentalen Testprozesses (siehe ATB et al. 2020). Zusätzlich zu diesem Prozess werden in der Arbeit auch neuartige Testtools mit Hinblick auf Security vorgestellt und prototypisch implementiert. Auch diese sollen bei der Komplexitätsbeherrschung helfen und die Testdauer verkürzen bzw. die Testabdeckung erhöhen. Um ein gesamtheitliches, bei KFZ-Herstellern implementierfähiges Security-Testvorgehen umzusetzen, beschreibt Ring wesentliche Security-Maßnahmen, welche die Testbasis verbessern und eine Vergleichbarkeit zwischen Herstellern ermöglichen sollen. Ganzheitlich wird in der Arbeit eine Teststrategie zum Testen von Fahrzeug-Security als technische Eigenschaft vorgestellt. Es wird argumentiert, dass Security-Testing unabhängig von den Testebenen durchgeführt werden kann. Daher geht er nicht auf einzelne Testmethoden auf verschiedenen Ebenen ein.

Einen ähnlichen Ansatz verfolgt auch Sattler (2015) in ihrer Arbeit. Bewusst geht sie auf die Testebene des Systemtests ein. Sie stellt eine Strategie zur Absicherung der funktionalen Sicherheit vor und möchte damit bewusst eine Detaillierungsebene weiter gehen als die Normreihe ISO 26262-x:201x. Ihr Ziel ist es, insbesondere durch die Verwendung von HiL-Tests eine Durchgängigkeit der Tests auf allen Ebenen zu erhalten und somit die Sicherheitsgefährdung durch effektive Testvorgehen zu reduzieren. Dazu wurde eine kennzahlenbasierte Methode zur Bewertung von Testmethoden und deren Einsatzmöglichkeiten erarbeitet. Auch hier wird durch ergänzende Forschungspunkte zur Testauswahl und Testfallpriorisierung, der Entwicklung einer beispielhaften Testumgebung inkl. HiL-Prüfplatz und einer Metrik zur Bewertung des Testfortschrittes ein ganzheitlicher Ansatz zum Testen verfolgt, um den gesamten Testprozess auf Systemebene zu beschreiben.

Im Vergleich zu diesen beiden Arbeiten geht Kiefner (2014) in seiner Arbeit nicht auf die Absicherung einer technischen Eigenschaft ein, sondern beschreibt eine Strategie zur Absicherung eines Verbunds von mechatronischen Systemen, insbesondere im Bereich Fahrwerk. In dieser Arbeit wird sich auf die Absicherung der obersten Testebene, den Abnahmetests mittels Fahrsimulator, Erprobungsfahrzeug und Verbund-HiLs konzentriert. Dadurch soll im Entwicklungsprozess die Fehlerfindung auf letzter Ebene verbessert werden, um die Gefährdung für Fahrzeugführer, Insassen und Dritte durch Fahrerassistenzsysteme zu reduzieren. Begründet wird die Auswahl der Testebene damit, dass Funktionen final nur im Verbund getestet und freigegeben werden können. Allerdings wird in dieser Arbeit kein vollumfänglicher Ansatz vorgestellt, der als Teststrategie bezeichnet werden kann, da sie sich ausschließlich auf risikobasierte Testmethoden konzentriert.

All diese Arbeiten versuchen durch verschiedene Ansätze die Effizienz des Testprozesses und der Testverfahren zu optimieren. Einen weiteren Ansatz dazu bietet Thiel (2012). Er stellt in seiner anwendungsfallbezogenen Arbeit ein automatisiertes Verfahren zur Überprüfung von Testfällen hinsichtlich semantischer Fehler für das Tool „EXAM“ des Volkswagenkonzerns vor. Die hier vorgestellten Beispiele zeigen die Breite der Lösungsansätze, mit denen Forschende das Problem der Komplexitätsbeherrschung im Bereich des Testens zu beherrschen versuchen. In der Industrie gibt es eine Vielzahl unternehmens- oder abteilungsspezifischer Teststrategien. Diese unterliegen meist der Geheimhaltungspflicht und werden daher nicht oder nur in groben Zügen veröffentlicht. Diese Strategien sind häufig jedoch so ausgelegt, dass neben der Sicherstellung der Qualität noch die Sicherstellung der funktionalen Sicherheit gewährleistet ist. Übergreifende Strategien sind jedoch sehr allgemein gehalten, sodass besondere technische Lösungen eine eigene Absicherungsstrategie benötigen.

1.3 Zielsetzung der Arbeit

Die Komplexitätssteigerung von Fahrzeugen durch neue Anforderungen beeinflusst deren Entwicklung. Der Software-Anteil mechatronischer Systeme ist besonders von Veränderungen geprägt, da dadurch neuartige Funktionen im Fahrzeug ermöglicht werden. Um die Qualität zu gewährleisten, werden vergleichbare Entwicklungsprozesse durchlaufen (Abbildung 1.4 oberer Teil). Ein wesentlicher Entwicklungsschritt ist das Testen. Teststrategien im SW-Testing werden durch unterschiedlichste Faktoren beeinflusst. Wie in Abschnitt 1.1.1 beschrieben, verfolgen die Tests das Ziel Fehlerzustände aufzudecken und Kenntnisse über die SW zu erlangen. Im Fokus steht der bestimmungsgemäße Gebrauch der Software. Daher wird SW im Automobilbau überwiegend gegen Anforderungen getestet, um die Sicherheit und Funktionalität zu gewährleisten.

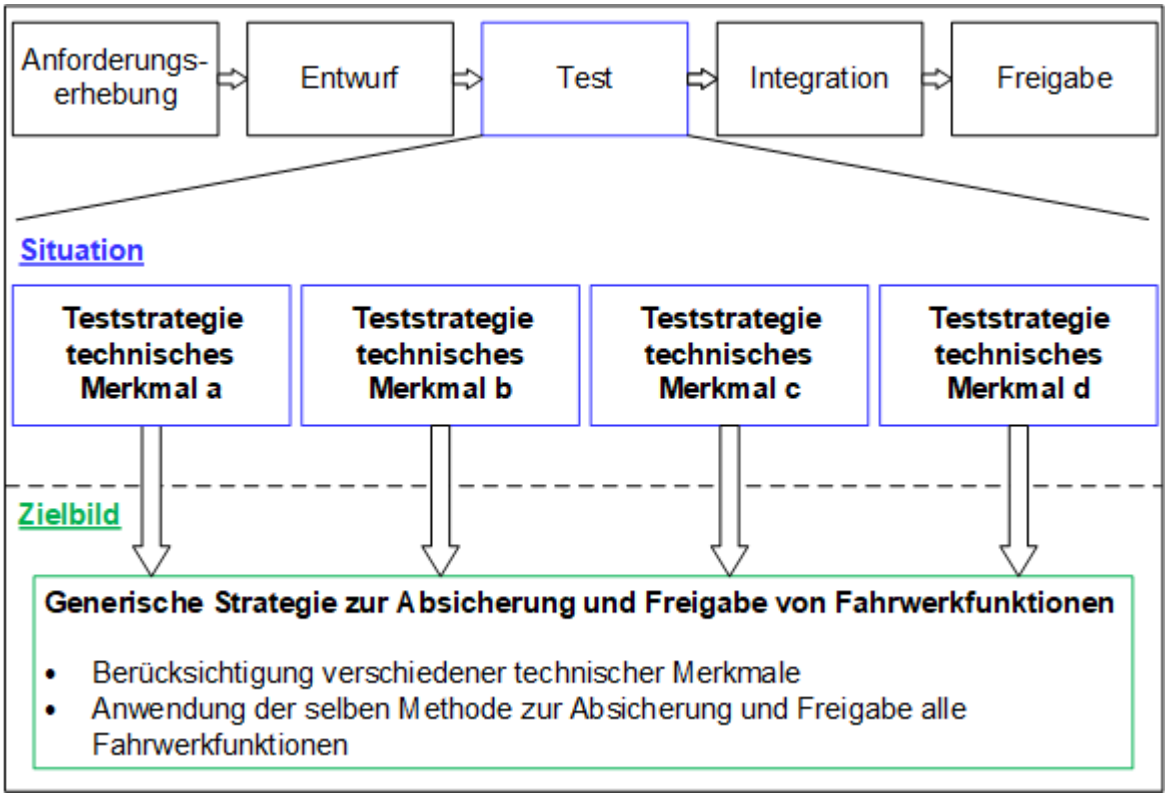

Abbildung 1.4: Zielbild der Dissertation

Häufig ist das Testen gegen Anforderungen jedoch nicht ausreichend. Aufgrund der zunehmenden Komplexität, sowohl gesetzlich als auch technologisch, ist es für eine Einzelperson fast unmöglich, den Gesamtzusammenhang zu überblicken und daher sicherzustellen, dass alle erforderlichen Tests durchgeführt wurden. Da diese Herausforderung schon lange besteht, gibt es, wie in Abschnitt 1.2 beschrieben unterschiedlichste Strategien, um SW zu testen und auf dieser Grundlage freizugeben. Zahlreiche unterschiedliche Strategien finden heutzutage im Automobilbau Anwendung. Wie Witte (2020) beschreibt, sollen Teststrategie mindestens abteilungsweite Gültigkeit besitzen. Da die meisten Strategien sich jedoch auf ein technisches Merkmal beziehen und nicht für Abteilungen ausgelegt sind, werden in einer Abteilung oder sogar einem Projekt mehrere Teststrategien zeitlich parallel angewandt. Dieses Phänomen beweisen auch die Arbeiten von Ring (2019) und Sattler (2015). Diese Situation ist in Abbildung 1.4 dargestellt und kann zu redundanten Tests mit demselben Erkenntnisgewinn führen. Im Umkehrschluss stellt dies einen Mehraufwand ohne Mehrwert dar. Der finanzielle Aufwand für solch ein Vorgehen lässt sich kaum rechtfertigen. Im Sinne der Effizienz gilt es daher, dieses Vorgehen zu vermeiden.

Zudem konzentrieren sich die meisten Forschungsarbeiten auf eine Systemebene bzw. den Abnahmetest. Dies gefährdet die Durchgängigkeit einer Teststrategie, was einerseits zu unzureichender Absicherung oder andererseits zu Mehraufwänden durch schlechte Abstimmung zwischen den Ebenen führen kann.

Ein weiterer wichtiger Punkt wurde in Abschnitt 1.1.2 und 1.1.3 erwähnt: die Zunahme an Normen und Softwarefunktionen. Da die Komplexität in den heutigen Entwicklungsstrukturen der Automobilhersteller über viele Personen abgebildet wird, müssen Wissen über Normen und Technologien zentral zur Verfügung gestellt werden. Zur Gewährleistung von Qualität ist es notwendig, dass Normen und Gesetze vereinheitlicht interpretiert werden. Dadurch ist es möglich, die Verifikation der SW standardisiert durchzuführen, um so den notwendigen Erkenntnisgewinn zum richtigen Zeitpunkt zu erhalten.

Ziel dieser Arbeit ist die Entwicklung einer generischen Test- und Freigabestrategie für ein agiles Entwicklungsumfeld, wie in Abbildung 1.4 unten dargestellt. Dadurch soll vermieden werden, dass mehrere Teststrategien zeitgleich Anwendung finden. Vornehmlich soll die generische Strategie dem Testmanager dienen, um in einem agilen Framework frühzeitig mit der Testplanung beginnen zu können. Teststrategien und Testdurchführung müssen dem Stand der Technik entsprechen. Um dies zu erreichen, bilden Normen und Gesetze, welche einheitlich interpretiert werden, das Fundament der Teststrategie. Relevant sind alle Technologien, die für Fahrwerkfunktionen eingesetzt werden und in Folge dessen alle gesetzlichen Anforderungen und Normen, die damit verbunden sind. Die generische Methode muss sicherstellen, dass das Testen effizient und zeitgleich ausreichend ist, um Erkenntnisse über den bestimmungsgemäßen Gebrauch zu erlangen und eine Freigabe im agilen Umfeld rechtfertigen zu können. Die Strategie muss folgende Inhalte abbilden:

- Fokus auf die Domäne „Fahrwerkentwicklung“.
- Berücksichtigung der relevanten gesetzlichen und normativen Anforderungen.
- Einheitliche Methode für verschiedene technische Merkmale.
- Durchgängigkeit der Strategie über mehrere Testebenen.

1.4 Aufbau der Arbeit

Die Arbeit ist in sieben Kapitel gegliedert, wobei das vierte, fünfte und sechste Kapitel den Hauptteil bilden. Die hier vorgestellten Methoden sind eng miteinander verzahnt (Abbildung 1.5) und werden für eine vollständige generische Methode zur Absicherung und Freigabe benötigt.

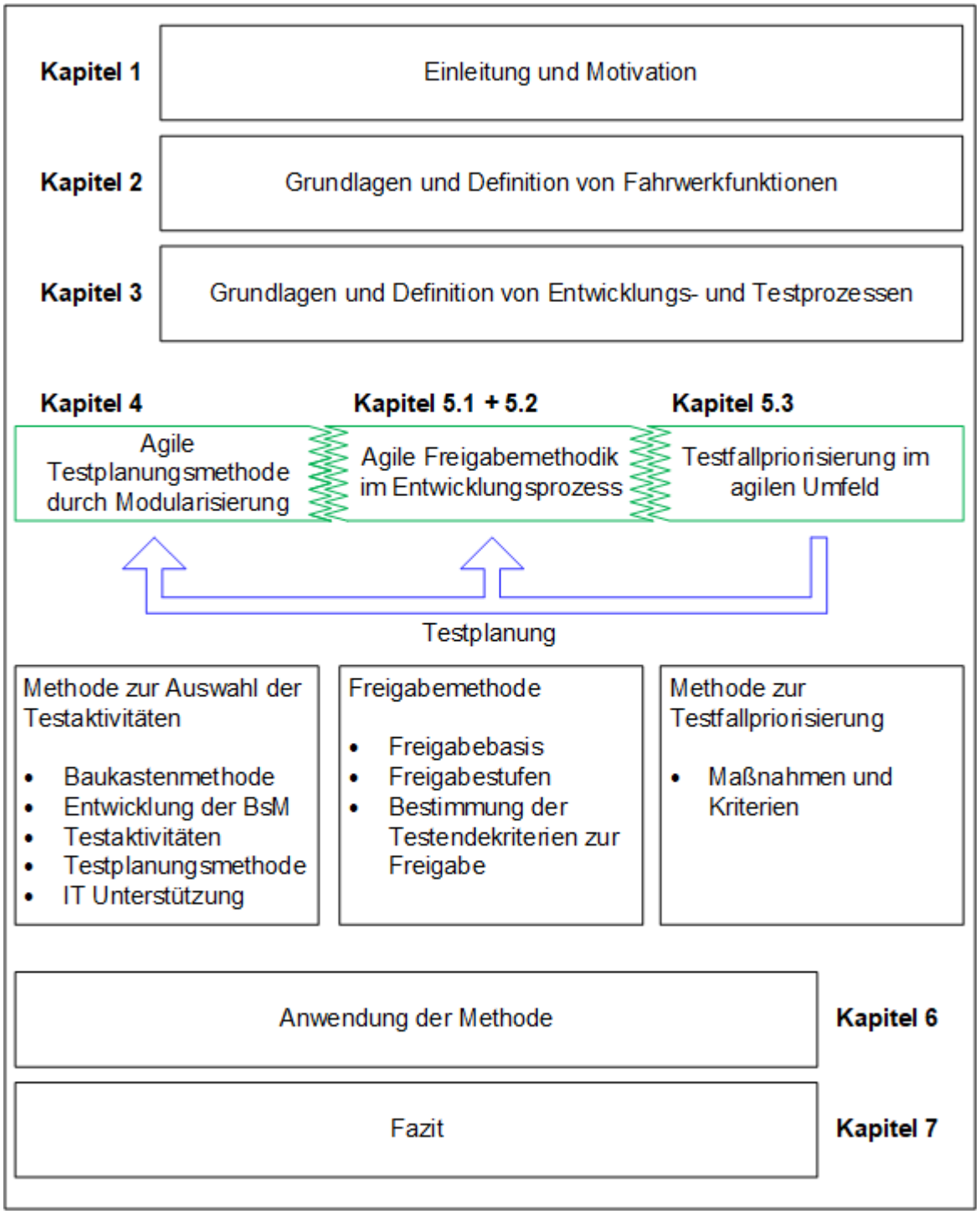

Abbildung 1.5: Struktur der Methode

Die Arbeit gliedert sich wie folgt:

Kapitel 1: In der Einleitung wird das Thema, beginnend mit einer Definition des Betrachtungsgegenstandes sowie anschließend einer Erörterung gesetzlicher und technischer Rahmenbedingungen generell vorgestellt. Des Weiteren wird die Zielsetzung der Arbeit formuliert sowie ein Einblick in den Stand der Technik gegeben.

Kapitel 2: Im zweiten Kapitel werden die Grundlagen von Fahrwerkfunktionen dargestellt. Zunächst wird das Fahrwerk als Domäne der Fahrzeugentwicklung eingeführt, anschließend wird die Bedeutung von Software hervorgehoben. Das Kapitel wird abgeschlossen mit

einer Beschreibung der System- und Softwarearchitektur, wobei auch verschiedene Bus-Systeme kurz erläutert werden.

Kapitel 3: Das dritte Kapitel konzentriert sich auf die Grundlagen der Entwicklungsprozesse, welche auf dem Produktentstehungsprozess und dem V-Modell als etablierte Modelle fußt. Darauf aufbauend wird die Bedeutung der agilen Vorgehensweise in der Softwareentwicklung diskutiert und beschrieben, wo diese im Konflikt mit den etablierten Modellen stehen. Anschließend werden in diesem Kapitel die Grundlagen von Softwaretesting und Testprozessen dargelegt. Der letzte Teil widmet sich den Grundlagen von Freigabeprozessen.

Kapitel 4: Kapitel vier stellt den umfangreichsten Teil der Arbeit dar. Hauptbestandteil des Kapitels bildet eine Methode zur Agilisierung der Testplanung. Zu diesem Zweck wurde ein Baukasten entwickelt, welcher auf Basis besonderer Merkmale verschiedene Testaktivitäten (sogenannte Testmodule) für verschiedene Testebenen vorgibt. Darüber hinaus werden neben der Entwicklung der Methode die rechtlichen Rahmenbedingungen erörtert, die zu der Methode führten. Das Kapitel mündet in der Vorstellung einer agilen Testplanungsmethodik, welche auch ein Konzept einer Toolunterstützung in der Testplanung beinhaltet.

Kapitel 5: Auch das fünfte Kapitel ist relevant für einen generischen Ansatz, es wird der Zusammenhang zwischen Testmodulen und Freigaben beschrieben. Zusätzlich werden Freigabelevel eingeführt, an die unterschiedliche Anforderungen an die Erreichung der Testendekriterien gestellt werden. Daher wird eine Methode zur Berechnung der Erfüllungsgrade für Testendekriterien herausgearbeitet.

Um die Zeitpläne, die in Kapitel 3 eingeführt werden, einhalten zu können und trotzdem norm- und gesetzeskonform zu testen, werden im fünften Kapitel zusätzlich Testfallpriorisierungsmaßnahmen vorgestellt. Anhand definierter Kriterien werden die Testfälle für die Testmethode aus Kapitel 4 priorisiert.

Kapitel 6: Im sechsten Kapitel wird der Einsatz der Methode im Test- und Freigabeprozess beschrieben. Dazu wird aufgezeigt, welche Aspekte zu welchem Zeitpunkt angewandt werden müssen, sodass ein Zusammenhang besteht und die Anwendung ausreichend beschrieben ist.

Kapitel 7: Das letzte Kapitel rundet die Arbeit durch eine Zusammenfassung ab. Es folgt die Einordnung des wissenschaftlichen Beitrags dieser Arbeit in den Kontext von Test- und Freigabestrategien in der Automobilentwicklung sowie ein Ausblick über weitere Entwicklungen und potenzielle Forschungsarbeiten.

2 Technische Grundlagen von Fahrwerkfunktionen

In diesem Kapitel werden das Fahrwerk und die Grundlagen von Fahrwerkfunktionen als Betrachtungsgegenstand der Arbeit vorgestellt. Ein wesentlicher Augenmerk liegt auf der Komplexitätssteigerung von Technik und Entwicklung, die für diese Arbeit relevant ist.

2.1 Fahrwerke in modernen Fahrzeugen

Das Fahrwerk eines Fahrzeuges hat einen maßgeblichen Einfluss auf das Fahrverhalten. Die Reifen sind der einzige Kontakt zur Fahrbahn und somit ein sicherheitskritisches Bauteil. Für die Sicherheit eines Fahrzeuges sind allerdings nicht die Reifen alleine zuständig, sondern das Fahrwerk als Verbund. Dies umfasst alle Bauteile, die in den Kontaktbereichen zur Fahrbahn für die Krafterzeugung, -beeinflussung und deren Übertragung ins Fahrzeug verantwortlich sind, sowie alle Bauteile, die zum Führen eines Fahrzeuges notwendig sind. Neben den Reifen zählen dazu Achsbauteile, Feder-Dämpfer-Systeme, Bremsen, Lenkung und eine Vielzahl an Fahrerassistenzsystemen wie z. B. ABS oder ESP. (Ersoy und Gies 2017; Pischinger und Seiffert 2021)

Neben der Verbindung zur Fahrbahn hat das Fahrwerk eine Vielzahl an weiteren Aufgaben. Diese umfassen u. a. die Spurführung, das Abfedern der Fahrzeugmasse und die Dämpfung von Schwingungen, die Kompensation von externen Störgrößen wie Seitenwind sowie die Fahrsicherheit und den Fahrkomfort zu gewährleisten. Um diese Aufgaben zu lösen und die Sicherheit und den Fahrkomfort zu erhöhen, ist das Fahrwerk besonders von den globalen Trends in der Fahrzeugentwicklung betroffen. Insbesondere die Digitalisierung und der Einsatz von Software sind ein ausschlaggebender Faktor für die Fahrwerkentwicklung. So steigt die Bedeutung von Software für die Fahrwerkentwicklung erheblich (Abschnitt 1.1.3). Zeitgleich ist ein hoher Vernetzungsgrad zwischen einzelnen Systemen notwendig, um die Fahrsicherheit und den Fahrkomfort zu erhöhen. Obwohl es möglich ist, einzelne Funktionalitäten des Fahrwerks getrennt zu entwickeln, ist es notwendig, eine Gesamtbetrachtung der einzelnen Systeme und deren Zusammenarbeit durchzuführen, um das Endergebnis für den Kunden zu optimieren. Der Grund ist, dass die einzelnen Systeme die Fahrdynamik auf unterschiedliche Weise beeinflussen und im schlimmsten Fall sogar gegeneinander arbeiten und so die Fahrsicherheit verringern. Um diese Gesamtintegration und Abstimmungsaufgabe zu lösen, können die Fahrwerksysteme in einer sogenannten Domänenstruktur gegliedert werden (Abbildung 2.1). Die einzelnen Systeme werden den Hauptfunktionen (Quer-, Längs- und Vertikaldynamik) des Fahrzeuges zugeordnet und miteinander in Zusammenhang gebracht. Hauptsächlich durch die softwaregesteuerte Fahrwerksregelung werden die einzelnen mechanischen und elektronischen Einzelsysteme zu einem Gesamtsystem verbunden. (Ersoy und Gies 2017; Pischinger und Seiffert 2021)

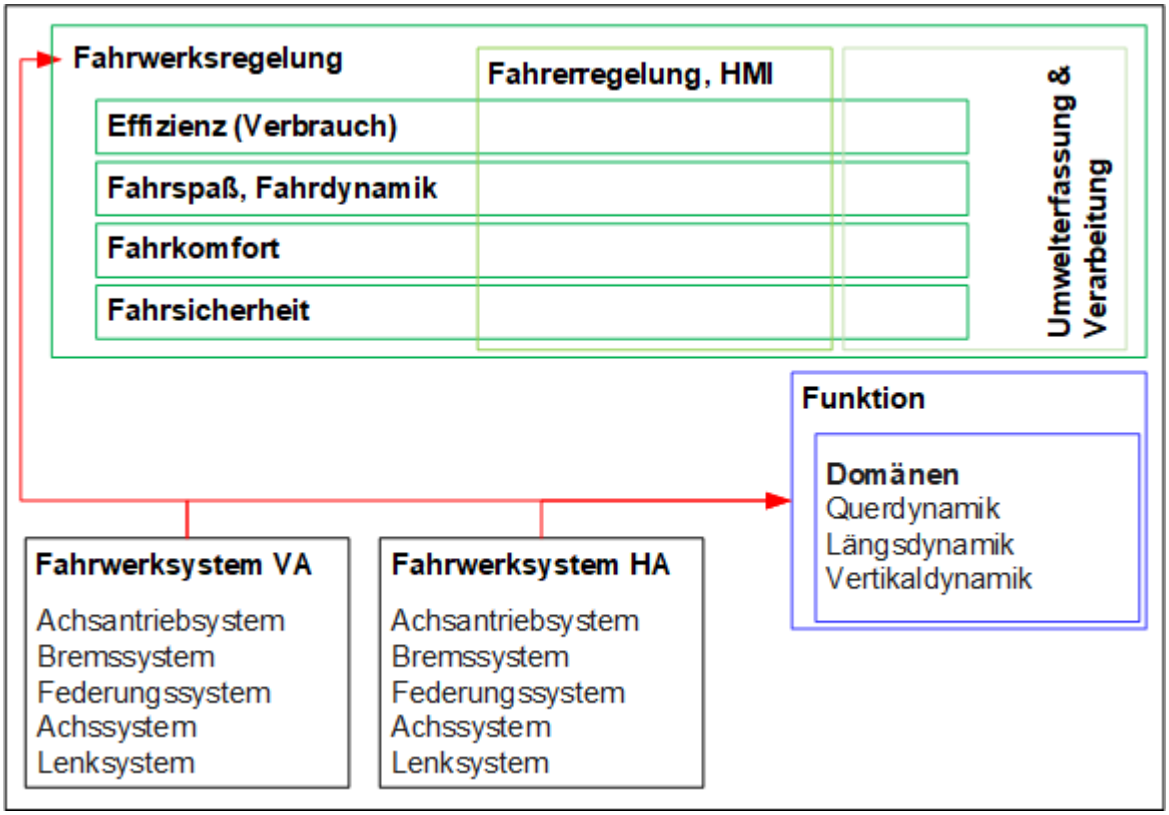

Abbildung 2.1: Strukturierung der Fahrwerkkomponenten und -system (modifiziert nach Ersoy und Vogel 2017)

Abbildung 2.1 gibt zusätzlich Aufschluss über die Auswirkungen zwischen den Systemen eines Fahrwerks und der Regelung. So haben die Systeme Einfluss auf Effizienz, Fahrkomfort oder die passive und aktive Sicherheit. Dabei müssen die Regelungen stets auf Umwelteinflüsse und Einflüsse durch den Fahrer regieren. Diese Wirkzusammenhänge und Einflüsse auf das Fahrverhalten sind in Abbildung 2.2 detailliert dargestellt. Es ist ersichtlich, welche Kraftflüsse und Bewegungsrichtungen durch die Hauptfunktionen abgebildet werden. Dadurch werden sowohl an den Fahrer als auch an die Regelsysteme Zustandsrückmeldungen gegeben. Diese Rückmeldungen beeinflussen die Aktionen des Fahrers und der Systeme, um den Fahrzustand positiv zu beeinflussen. Abhängig von der ausgeführten Aktion des Fahrers verändern sich die Reifenkräfte und damit auch die Notwendigkeit des Eingreifens durch Regelsysteme, um die übergeordneten Ziele (Komfort, Fahrverhalten, Sicherheit) zu erreichen. Aus der Abbildung lassen sich die Komplexität des Fahrwerks und die Aufgabe der gesamtheitlichen Abstimmung zur Erfüllung der übergeordneten Ziele ersehen. (Pauly et al. 2021)

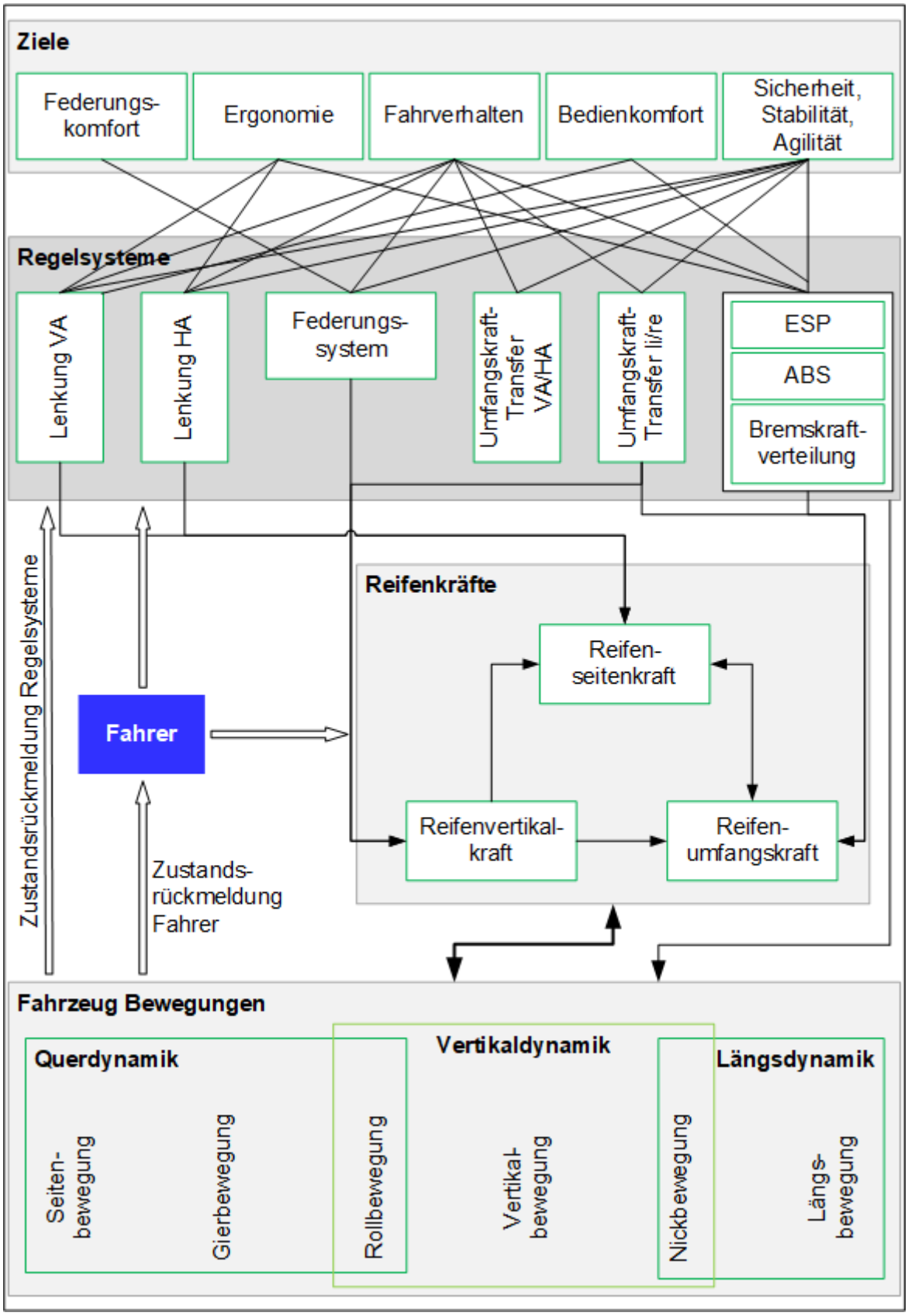

Abbildung 2.2: Wirkzusammenhänge fahrdynamischer Regelsysteme (modifiziert nach Pauly et al. 2021)

2.2 Software in der Fahrwerkentwicklung

Nicht nur Fahrwerke, sondern die Fahrzeugentwicklung generell sind heute eine sehr komplexe Aufgabe. Zur effektiven Gestaltung werden Fahrzeuge durch große Teams simultan entwickelt (Details dazu sind auch in Kapitel 3 zu finden). Die Fahrwerkentwicklung ist häufig so gegliedert, dass Teams aus Spezialisten einzelne Fahrwerksysteme verantworten und zur Reife bringen. Damit das Zusammenspiel der einzelnen Systeme hinreichend gut funktioniert und ein Fahrerlebnis generiert wird, das zu einem bestimmten Markenimage passt, müssen die Systeme durch eine Fahrdynamikabteilung ganzheitlich aufeinander abgestimmt werden.

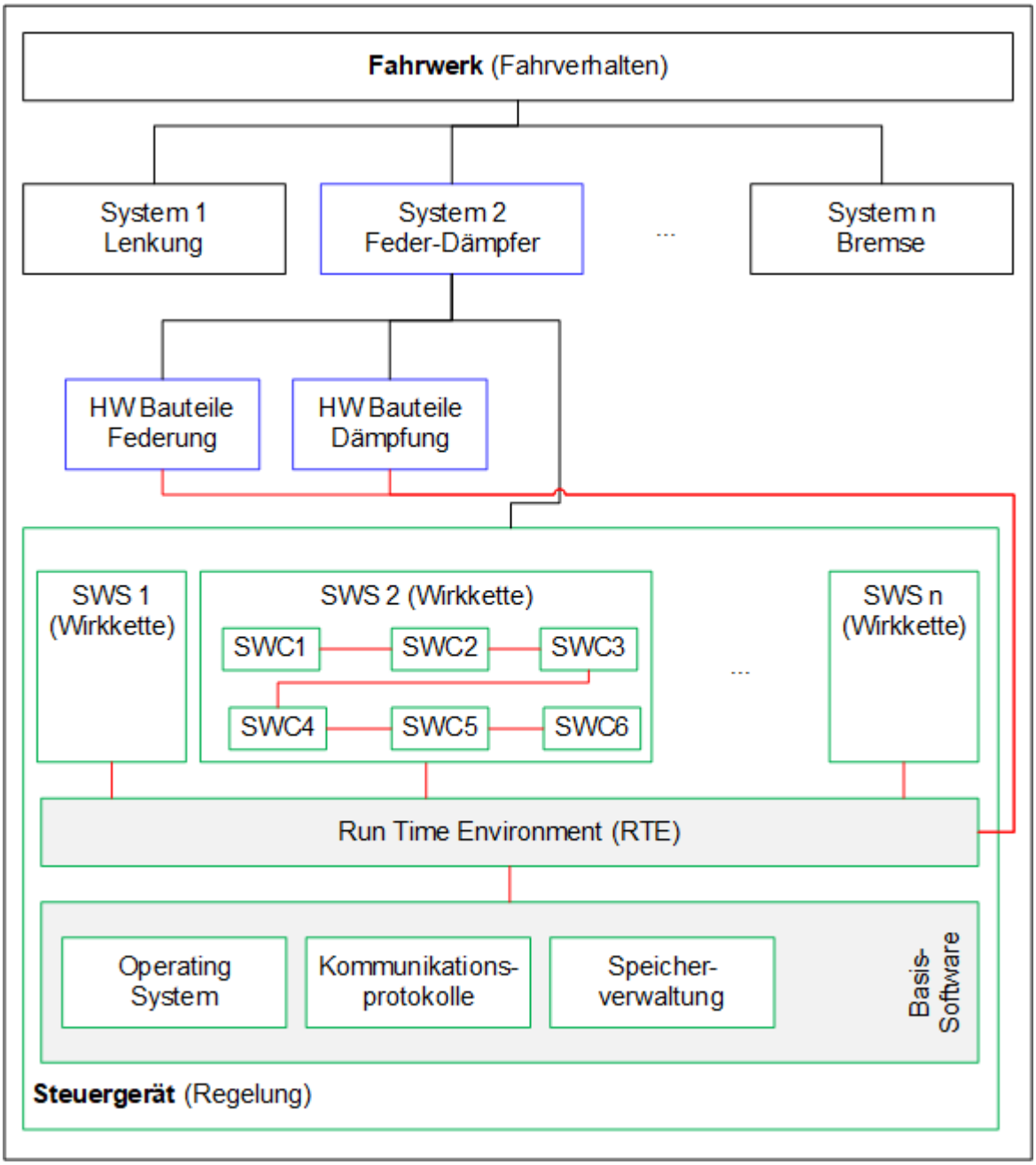

Abbildung 2.3: Zusammenwirken von HW-Bauteilne mit der Regelung im SG

Beispielsweise besteht ein Fahrwerksystem, z. B. ein geregeltes Feder-Dämpfer-System aus HW-Elementen sowie der Regelung. Die HW-Elemente umfassen u. a. Federbeine,

Lager oder Dämpfer. Die Regelung ist als Softwareanteil zentral in einem Steuergerät wie einem HCP, verortet. Das SG regelt nicht ausschließlich das Feder-Dämpfer-System, sondern ebenso eine Vielzahl an weiteren Systemen. In Abbildung 2.3 ist schematisch der Aufbau eines Fahrwerksystems dargestellt. Mit den schwarzen Linien wird das Herunterbrechen des Fahrwerks in einzelne Elemente dargestellt. Das Fahrwerk besteht aus mehreren mechatronischen Systemen, die wiederum aus HW, SG und SW bestehen. In roter Farbe ist der Regelfluss des Feder-Dämpfer-Systems dargestellt. Auf dem Steuergerät sind die Basissoftware, Wirkketten (bzw. Softwaresysteme (SWS)) bestehend aus mehreren Softwarekomponenten (SWCs)) und die Run Time Environment (RTE) installiert. In Verbindung erfolgt über diese Elemente die Regelung für die einzelnen Fahrwerksysteme. In dieser Arbeit wird sich insbesondere auf die Absicherung der SWCs und SWS‘ konzentriert.

2.2.1 System- und Softwarearchitektur

Einige Funktionen benötigen Informationen bzw. Signale von anderen Steuergeräten oder die SWCs einer Wirkkette sind über mehrere Steuergeräte verteilt. Deshalb muss für die Realisierung der Funktionen eine konkrete Architektur auf SG Ebene festgelegt werden, welche Steuergeräte, Vernetzungspläne mit Gateways und verschiedene Bussysteme umfasst. Über die Bussysteme können die SG miteinander kommunizieren und Signale versendet werden. In modernen Fahrzeugen kommt mehr als ein System zum Einsatz. So können, wie in Abbildung 2.4 schematisch dargestellt, zeitgleich mehrere CAN, Ethernet, FlexRay, MOST und LIN Verbindungen zum Einsatz kommen (Ring 2019; Schäuffele und Zurawka 2013).

Um die Komplexität zu beherrschen, gibt AUTOSAR (Automotive Open System Architecture) Standards zur Entwicklung vor. Man unterscheidet zwischen der Classic Plattform und der Adaptive Plattform. Die Classic Plattform unterscheidet drei Layer wie auch in Abbildung 2.3 dargestellt. Diese sind:

- Basissoftware: Betriebssystem, Services und Driver.
- Run-Time-Environment (RTE): Schnittstelle zwischen Basissoftware und Anwendungssoftware.
- Anwendungssoftware: Hardwareunabhängige SWCs zur Funktionsdarstellung.

Die Adaptive Plattform versucht hingegen eine flexiblere Entwicklung durch die Verwendung zentraler Steuergeräte auf Ethernet Basis zu ermöglichen. Dadurch soll es über einen Fahrzeuglebenszyklus hinweg möglich sein, die Anwendungssoftware upzudaten, auszutauschen oder zu ergänzen. (AUTOSAR 2022a, 2022b; Tischer 2018) In dieser Arbeit liegt das Augenmerk auf der Anwendungssoftware und wird lediglich für ein besseres Verständnis hier in einen allgemeinen Zusammenhang gebracht. Weiterführende Informationen zum AUTOSAR Standard sind auf https://www.autosar.org/ nachzulesen.

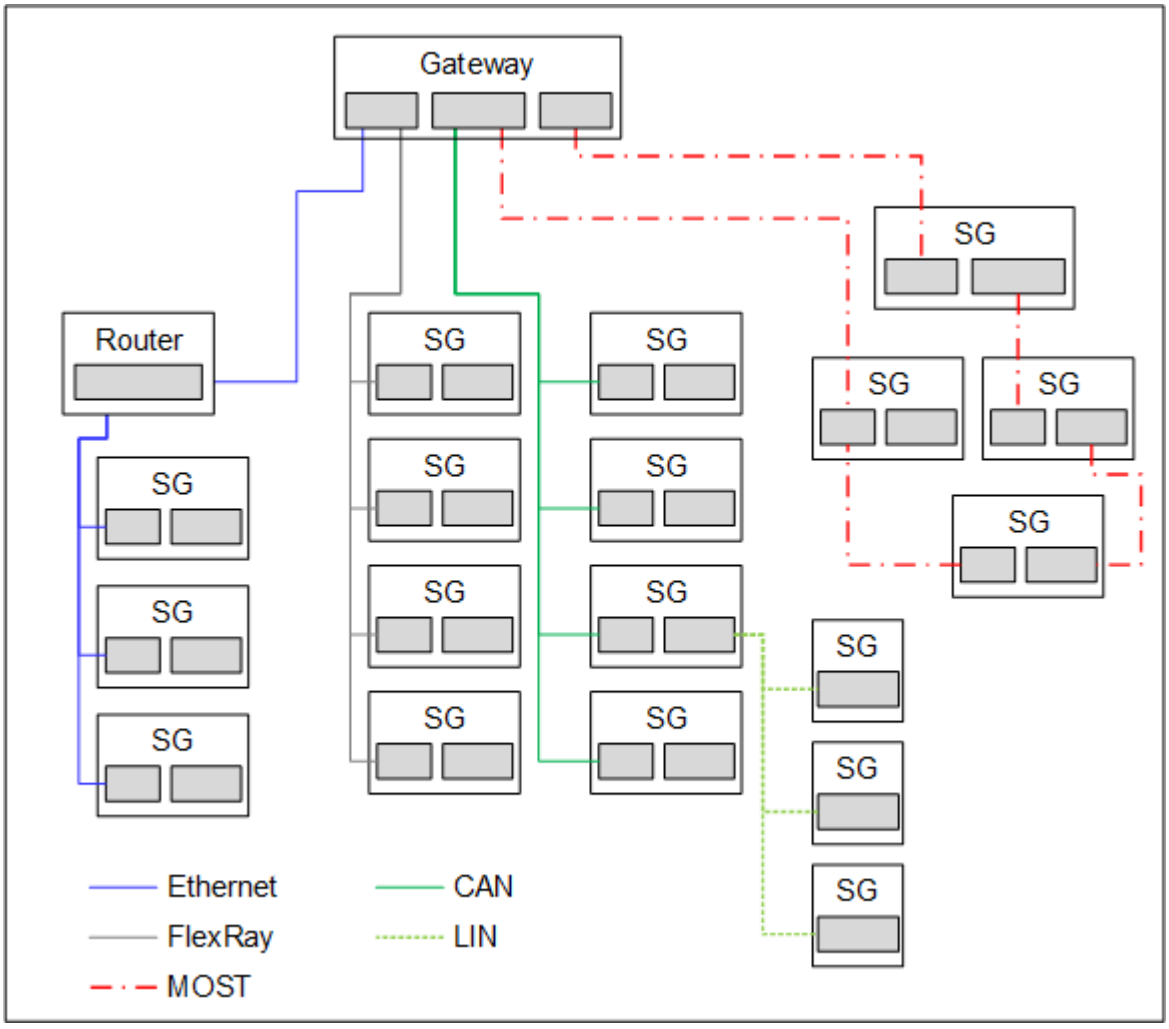

Abbildung 2.4: Bussysteme im KFZ (modifiziert nach Ring 2019)

Abbildung 2.3 zeigt beispielhaft die SWCs des Feder-Dämpfer-System. Zusammengefasst ergeben diese SWCs die Wirkkette und sind nach dem AUTOSAR-Standard von der Hardware losgelöst und wiederverwendbar (Zwintzscher 2005). Dadurch können SWCs in allen Steuergeräten, die nach dem AUTOSAR-Standard spezifiziert sind, integriert werden. Dieser wesentliche Vorteil hilft Entwicklern von Software bei der Beherrschung der Komplexität während der Entwicklung. Der Standard gibt eine klare Softwarearchitektur vor. Die kleinsten Elemente bilden Units, oftmals auch als Module bezeichnet. Nach der ISO 262626-6 (ISO 26262-6:2018) ist eine Unit die kleinste testbare Einheit. Mehrere Units werden zu SWCs integriert. Weder eine Unit noch eine SWC können ein System im Fahrzeug steuern oder regeln. Erst durch die Integration mehrere SWCs zu einer Wirkkette können einzelne Aufgaben der SW gelöst werden. Durch diese Zerlegung der Architektur können Verantwortungen klar verteilt werden. Die schrittweise Integration zu immer größeren Softwarebausteinen ermöglicht es, Fehler im Testprozess frühzeitig zu erkennen (ISO 26262-6:2018; Zwintzscher 2005).

Die Entwicklung von SWCs basiert sowohl auf funktionalen wie auch nicht funktionalen Anforderungen, was zu unterschiedlichsten Anwendungsfällen führt. Beispielsweise Steuerung oder Regelung einer Funktion, die Sicherheitsüberwachen, Schutz gegenüber (Hacker-)

Angriffen zu bieten oder die Anbindungen an eine Cloud zu ermöglichen. Durch den Technologiezuwachs und die Verpflichtung, weltweit mehr und mehr Regularien zu erfüllen, nimmt die Anzahl an SWCs zu, was wiederum die Anforderungen an die Architektur und Rechenleistung erhöht. Diese Veränderungen haben Einfluss auf Teststrategien und damit eine Relevanz für die vorliegende Arbeit. (Jooß und Schramm 2022)

3 Entwicklungsprozesse in der Fahrzeugentwicklung

Um die Komplexität moderner Fahrzeuge zu beherrschen, ist es notwendig, die Entwicklung strukturiert, koordiniert und simultan durchzuführen. Verschiedene, aufeinander abgestimmte Prozesse ermöglichen dies. Eine Vielzahl solcher Prozesse zählt heute zum Stand der Technik. Grundlagen dafür sind oftmals verschiedene Normen und Handbücher. In diesem Kapitel wird zunächst auf klassische Entwicklungsprozesse und Vorgehensmodelle in der Fahrzeugentwicklung eingegangen. Als Kontrast werden agile Vorgehensweisen in der SW-Entwicklung vorgestellt. Anschließend werden Softwaretesting und die dazugehörigen Prozesse untersucht. Abgeschlossen wird das Kapitel durch die Beschreibung des Freigabeprozesses.

3.1 Produktentstehungsprozess und V-Modell

Bis Mitte des 20. Jahrhunderts wurden Fahrzeuge ausschließlich bestehend aus Hardware-Elementen, oftmals durch eine Person bzw. kleine Entwicklerteams konstruiert und getestet. Durch die massive Steigerung der Komplexität, wie in Kapitel 2 vorgestellt, ist dieses Vorgehen in der modernen Fahrzeugentwicklung unmöglich. Einzelne Fahrzeugtypen werden durch viele Personen gleichzeitig entwickelt. Widmann et al. (2021) sprechen von über 1.000 Personen, die an der Entwicklung eines Fahrzeuges beteiligt sind. Die Beteiligten sind in unterschiedliche Teams eingeteilt, die parallel die Entwicklung des Fahrzeuges vorantreiben und unterschiedliche Verantwortlichkeiten haben. Die Arbeit ist geprägt durch viele (technische) Schnittstellen zu anderen Teams. Um die Funktionsfähigkeit eines Fahrzeugs am Ender der Entwicklung garantieren zu können, ist eine zeitliche und inhaltliche Synchronisation der einzelnen Teams wesentlich. Einerseits sind Abstimmungen auf Arbeitsebene notwendig, andererseits sind gemeinsame Terminpläne und Projektmeilensteine unabdingbar.

In der Richtlinie VDI 2221 (VDI Richtlinie VDI 2221) wird dazu eine „Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte“ beschrieben. Hierbei wird die Entwicklungsaufgabe in einen Prozess, bestehend aus sieben Prozessschritten, unterteilt. Das Ziel der Methode ist es, die Entwicklungsaufgabe so zu strukturieren, dass trotz steigender Komplexität ein wettbewerbsfähiges Produkt entwickelt werden kann. Hierfür werden im vierten Schritt die Produkte in mehrere Einzelmodule zerlegt, die im fünften Schritt getrennt voneinander entwickelt werden, um sie dann im sechsten Schritt zu einem Gesamtprodukt zu integrieren. Diese Abfolge wird in der Richtlinie VDI 2223 (VDI Richtlinie VDI 2223) nochmals genauer beleuchtet, um die Methodik auch auf abweichende Rahmenbedingungen zu adaptieren. In der Praxis bedeutet die Anwendung der Richtlinien, dass die Entwicklung klar strukturiert sein muss, sodass jeder Zeitpläne und Verantwortlichkeiten nachvollziehen kann. Unter

anderem erfolgt dies mittels eines zentralen Projektmanagements, das Abläufe und Terminpläne vorgibt und kontrolliert. Realisiert wird dies in der Fahrzeugentwicklung meist mit einem generischen Produktentstehungsprozess (PEP).

Der PEP gilt als übergreifender Ablaufplan der Entwicklung und kommt überwiegend in der Entwicklung komplexer Produkte wie Automobilen zur Anwendung. Es werden alle Prozesse, die zu den Entwicklungsabläufen von der Produktidee bis zur Markteinführung zählen, beschrieben. Der PEP bündelt alle wichtigen Prozesse, um ein Fahrzeugmodell zu einem definierten Termin mit einer bestimmten Qualität und zu festgelegten Kosten gesamtheitlich zu entwickeln. Einzelnen Fahrzeughersteller (Original Equipment Manufacturer (OEM)) nutzen hierzu angepasste Varianten des PEPs, die sich in den Grundzügen jedoch ähneln. (Ersoy et al. 2017; Widmann et al. 2021)

Mit steigender Komplexität der Fahrzeuge nimmt die Bedeutung des PEPs zu. So werden Fahrzeuge und Fahrzeugsysteme nicht ausschließlich von einem OEM, sondern ebenso durch Lieferanten und Entwicklungsdienstleistern entwickelt. Zusätzlich müssen immer mehr Anforderungen aus unterschiedlichen Bereichen, sowohl gesetzliche als auch Kundenwünsche berücksichtigt werden. Um all diesem gerecht zu werden, muss der PEP alle Beteiligten miteinander vernetzen. Nur so kann am Ende der Entwicklung ein konkurrenzfähiges Produkt entwickelt und verkauft werden. (Widmann et al. 2021)

Der PEP teilt sich in mehrere Phasen und Meilensteine auf. Diese sind bereichsübergreifend verbindlich und unabhängig von Funktionalität einzelner Bauteile. In der Literatur werden unterschiedliche Phasen genannt. So erläutern Ersoy et al. (2017), dass der PEP in vier Phasen unterteilt werden kann, in der Praxis jedoch häufig nur drei Phasen definiert werden. Die erste Phase ist jedoch bei allen die sogenannte Konzept- oder Produktdefinitionsphase. Anschließend beginnt die Konzeptabsicherungsphase, gefolgt von der Serienentwicklungsphase. In einigen Ansätzen ist die letzte, eher kurze Phase des Serien- oder Produktionsanlaufs. (Ersoy et al. 2017; Widmann et al. 2021)

Bei der Porsche AG wird der PEP in die drei Phasen „Definitionsphase“, „Konzeptabsicherungsphase“ und „Serienentwicklungsphase“ eingeteilt. Den Abschluss der Serienentwicklungsphase bildet der SOP (Start of Production). Bei der Konzernschwester Audi AG kommen ebenfalls drei Phasen zur Anwendung. Allerdings ist die dritte Phase, der sogenannte „Launch“, der mit SOP beginnt und drei Monate andauert (Ersoy et al. 2017). Zur Synergienutzung im VW-Konzern werden Fahrzeugprojekte der Marken Audi und Porsche simultan in Entwicklungsabteilungen mehrerer Konzern-Unternehmen entwickelt. Insbesondere auf Steuergeräte-Ebene ist dies der Fall. Daher müssen bei der Softwareentwicklung die Produktentstehungsprozesse beider Marken berücksichtigt werden, um die entwickelnden Teams zu synchronisieren. Aufgrund dessen müssen die Meilensteine der einzelnen PEPs über den gesamten Konzern synchronisiert sein und für die agile SW-Entwicklung (Abschnitt 3.2) Gültigkeit besitzen.

Zur Synchronisierung werden die Phasen wiederum in mehrere Meilensteine unterteilt. Diese stellen Kontrollpunkte dar. Deshalb müssen zuvor definierte Anforderungen erreicht und Entwicklungsaktivitäten abgeschlossen sein, um mit den nächsten Entwicklungsschritten fortzufahren (Ersoy et al. 2017). Damit es bei der simultanen Entwicklung von Porsche- und Audi-Modellen nicht zu Verzögerungen kommt, sind diese Meilensteine teilweise synchron. In Abbildung 3.1 sind die unterschiedlichen Phasen in einen zeitlichen Zusammenhang gebracht. Beispielhaft sind an den Meilensteinen DF (Design Freeze), BST (Baustufe, Erprobungsstart) und PVS (Produktionsvorserie) die Synchronisierungspunkte zur Überwachung des Projektfortschrittes dargestellt.

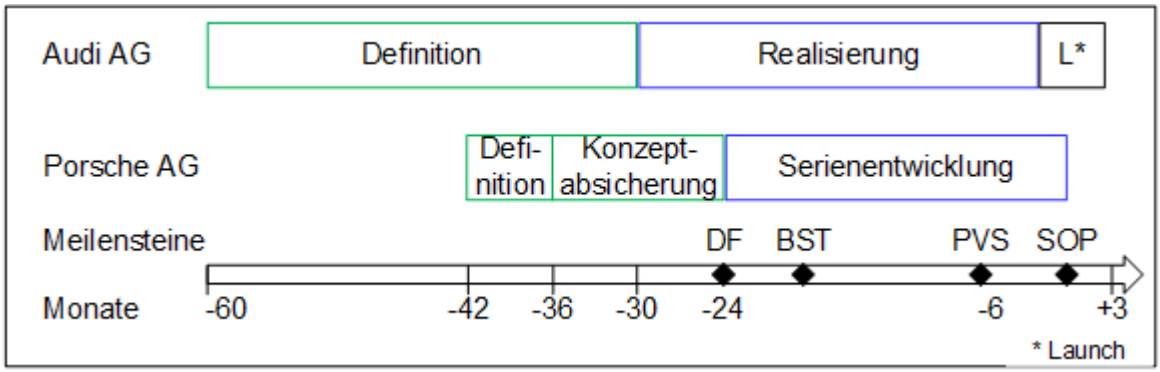

Abbildung 3.1: Vergleich der Phasen und Meilensteine zwischen der Audi AG und der Porsche AG (modifiziert nach Ersoy et al. 2017)

Weiterführende Informationen zu Produktentstehungsprozessen sind beispielsweise zu finden in Ersoy et al. (2017), Widmann et al. (2021) oder der VDI Richtlinie VDI 2221 und der VDI Richtlinie VDI 2223.

Da der PEP ein Rahmenwerk für einen langen Zeitraum vorgibt, in dem die meisten Entwicklungsschritte in den Entwicklungsphasen mehrmals wiederholt werden, um das Produkt zu einer Reife zu bringen, werden Vorgehensmodelle angewandt. Das wohl bekannteste Vorgehensmodell in der Softwareentwicklung ist neben dem „Wasserfallmodell“ das „V-Modell“. Das V-Modell findet auf unterschiedlichen Ebenen Anwendung. So beschreibt der VDI in der VDI Richtlinie VDI 2206 ein V-Modell für die Entwicklung mechatronischer Systeme. Da die Software nur ein Teil des mechatronischen Systems darstellt, erfolgt die Software ebenfalls einem Prozess, der sich am V-Modell orientiert. (Hoffmann 2013; Krause und Gebhardt 2018)

Das V-Modell dokumentiert einen durchgängigen Entwicklungsprozess, in dem Systeme zur Komplexitätsbeherrschung zunächst in kleine Komponenten zerlegt und anschließend wieder zu einem Gesamtsystem integriert werden. Dabei werden die einzelnen Schritte in einem V-förmigen Diagramm dargestellt (Abbildung 3.2). Der linke Ast des V-Modells repräsentiert die Anforderungs- und Entwicklungsseite dar. Hier werden die Systeme und deren Anforderungen auf die unterste Ebene (kleinste zu entwickelnde Einheit) heruntergebrochen. Die rechte Seite des V-Modells beschreibt hingegen die Integrations- und Testaktivitäten. Jede Anforderungsebene auf der linken Seite entspricht einer Integrationsebene auf der rechten

Seite. Die enge Vernetzung der beiden Seiten soll dazu führen, dass Auffälligkeiten und Fehler möglichst frühzeitig entdeckt werden. (Hoffmann 2013; Schäuffele und Zurawka 2013)

Diese Arbeit bezieht sich überwiegend auf den rechten unteren Bereich des V-Modells, also auf die Arbeitsschritte nach der Entwicklung einzelner Softwareelemente, die deren Verifikation dienen. Überwiegend umfasst dies die Tests der einzelnen Komponenten, die Integration dieser zu einer Gesamtsoftware und daraus folgend die Integrationstests. Die Systemebene, die der Validierung dient, wird hingegen nicht berücksichtigt.

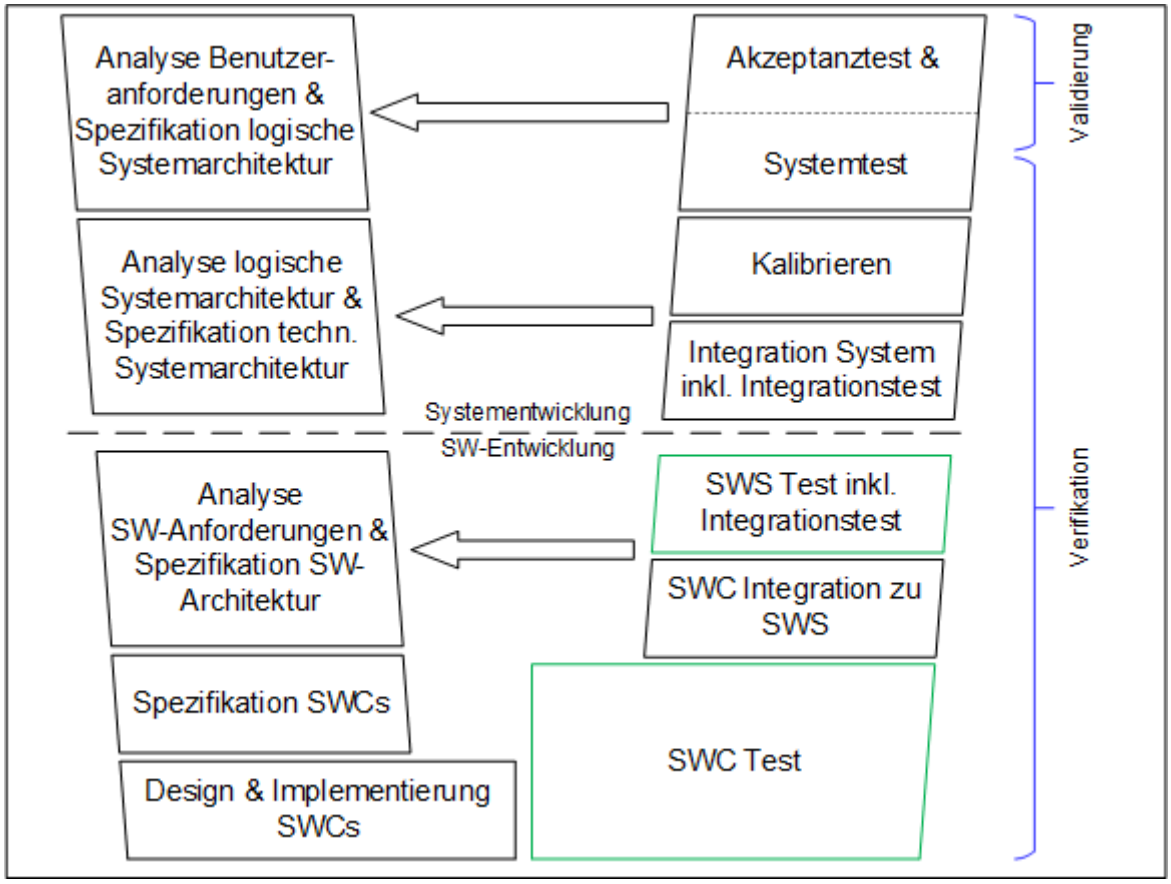

Abbildung 3.2: Das V-Modell (modifiziert nach Schäuffele und Zurawka 2013)

In der Literatur sind Vorgehensmodelle hinreichend beschrieben, weitere Informationen können beispielsweise Hoffmann (2013) oder Krause und Gebhardt (2018) entnommen werden.

3.2 Agilisierung der Softwareentwicklung

Der Produktentstehungsprozess und die damit verbundenen Vorgehensmodelle sind in ihrem zeitlichen Ablauf starr und bieten wenig Flexibilität. Als eine Ursache kann angebracht werden, dass diese auf die Entwicklung von Hardware im Fahrzeugbau ausgelegt sind. Die starren Abläufe und Terminschienen führen jedoch nicht zwingend zu einer fristgerechten Fertigstellung des Entwicklungsprojektes. Beispielsweise war in der Presse von Verzögerung der Auslieferungen bzw. Auslieferungsstopps der Volkswagenmodelle ID.3 und Golf VIII zu lesen

(Dralle 2020), aber auch andere Hersteller haben ihre Probleme, was sich in den hohen Rückrufquoten widerspiegelt (Doll 2018). Ursache sind meistens Softwareprobleme, welche durch steigende Komplexität sowie höhere Entwicklungsgeschwindigkeit zustande kommen (Bohnet 2015; Center of Automotive Management (CAM) 2021). Anders als Hardwarebauteile bedarf die Entwicklung hochgradig individueller Software eine hohe Flexibilität. Um dies zu erreichen, wurde in den vergangenen Jahren eine Agilisierung der Softwareentwicklung populär.

3.2.1 Agile Methoden in der Software Entwicklung

Agile Methoden werden als Ergebnis der streng reglementierten Entwicklungsprozesse angesehen. Dadurch mussten viele Entwickler Aufgaben nachgehen, die nicht direkt mit der Entwicklung verbunden, sondern eher organisatorischer Natur waren. Die Tätigkeiten werden als nicht-wertschöpfend betrachtet. Darunter leidet das Produkt „Software“. Folglich ist das Ziel agiler Methoden in erster Linie die Entwicklung zu flexibilisieren und den Schwerpunkt auf das Entwicklungsergebnis zu legen. Anders als bei starren Entwicklungsprozessen wird bei der agilen Entwicklung stets davon ausgegangen, dass etwas Unvorhersehbares passieren wird und eine Reaktion darauf notwendig ist. Daher ist hier kein standardisierter Ablauf vorgesehen, sondern ein Ablauf, der vor allem darauf fußt, dass auf Ereignisse reagiert werden muss. (Douglass 2016; Dyba und Dingsoyr 2009)

Folglich benötigen agile Methoden schnelle Reaktionen des Projektteams. Anders als bei den klassischen Vorgehensmodellen werden hier innerhalb kürzester Zeit Softwareteile entwickelt und entwicklungsbegleitend verifiziert. Hingegen werden bei klassischen Vorgehensmodellen langwierige Prozesse durchlaufen. (Douglass 2016)

2001 haben mehrere Softwareentwickler und Autoren aus diesem Bereich ein Manifest der agilen Softwareentwicklung veröffentlicht (Beck et al. 2001). Dieses Manifest besteht aus vier wesentlichen Grundsätzen:

1. Individuen und Interaktionen sind wichtiger als Prozesse und Werkzeuge
2. Funktionierende Software ist wichtiger als eine umfassende Dokumentation
3. Zusammenarbeit mit dem Kunden ist wichtiger als Vertragsverhandlungen
4. Reagieren auf Veränderungen ist wichtiger als das Befolgen eines Plans

Aus diesen Grundsätzen kann man zwölf Prinzipien der agilen Softwareentwicklung ableiten. Diese lauten:

1. Die höchste Priorität hat die Kundenzufriedenheit. Dies wird erreicht durch frühzeitige und kontinuierliche Auslieferung nützlicher Software.
2. Änderungen der Anforderungen sollen auch spät in der Entwicklung willkommen geheißen werden. Agile Prozesse nutzen Veränderungen zum Wettbewerbsvorteil des Kunden.

3. Funktionierende Software soll regelmäßig geliefert werden. Der Abstand zwischen den Lieferungen kann wenige Wochen oder Monate sein, wobei kürzere Abstände zu präferieren sind.
4. Fachexperten und Entwickler müssen während des ganzen Projektes täglich zusammenarbeiten.
5. Projekte sollen um motivierte Personen herum aufgebaut werden. Mit der entsprechenden Umgebung, notwendiger Unterstützung und Vertrauen werden sie das Projekt erfolgreich bearbeiten.
6. Am effizientesten und effektivsten können Informationen innerhalb eines Entwicklungsteams durch persönliche Gespräche verteilt werden.
7. Funktionierende Software ist das wichtigste Maß für Fortschritt.
8. Agile Prozesse fördern nachhaltige Entwicklung. Auftraggeber, Entwickler und Nutzer sollten eine gleichmäßige Geschwindigkeit auf unbestimmte Zeit einhalten können.
9. Anhaltende Aufmerksamkeit auf technisch ausgeklügelte Lösungen und gute Entwürfe fördern Agilität.
10. Einfachheit ist essenziell.
11. Die besten Architekturen, Anforderungen und Entwürfe entstehen durch selbst organisierte Teams.
12. In regelmäßigen Abständen sollen Teams prüfen, wie sie effektiver arbeiten können, um anschließend ihr Vorgehen entsprechend anzupassen.

Neben der Originalquelle (Beck et al. 2001) werden diese immer wieder in der Literatur aufgegriffen. Beispielsweise in Douglass (2016) oder Meyer (2014).

Auf dieser Basis gilt für agile Arbeit folglich, dass Kundenwünsche stets im Zentrum der Entwicklung stehen und die Arbeit in mehreren kleinen Iterationen erledigt werden soll. Elementar ist hierbei die Berücksichtigung von möglichen Änderungen der äußeren Bedingungen. Dies kann durch enge Zusammenarbeit zwischen Entwicklern und Auftraggebern geschehen. Eine strenge Hierarchie ist hingegen kontraproduktiv in agilen Entwicklungsprojekten und sollte daher vermieden werden (Meyer 2014). Ebenso soll auf feste Projektpläne verzichtet werden. Vertrauen wird vielmehr durch regelmäßige Auslieferungen funktionierender Software an den Kunden erzeugt. Dadurch werden nach Douglass (2016) folgende Vorteile erzielt:

- Verbesserung der Entwicklungsergebnisse.
- Verbesserung der Effizienz.
- Frühzeitige Renditemöglichkeiten.
- Zufriedene Stakeholder.
- Verbesserung der Kontrolle über einzelne Projekte.

- Erhöhte Reaktionsfähigkeit auf Veränderungen.
- Reduzierung der Projektrisiken.

Zur Erreichung dieser Ziele werden agile Methoden und Rahmenwerke angewandt. Diese basieren auf der Idee, die Software inkrementell, iterativ und evolutionär zu entwickeln und weichen daher stark von der klassischen Fahrzeugentwicklung mit vielen frühzeitig festgelegten Meilensteinen ab. Die wohl bedeutendsten Methoden bzw. Rahmenwerke sind Extreme Programming und Scrum (Schmidt 2016).

Unter **Extreme Programming** wird eine Gruppe von Methoden verstanden, durch deren Anwendung Entwickler die Qualität der Software erhöhen und gleichzeitig die Reaktionsfähigkeit auf Änderung verbessern. Methoden und Konzepte der Softwareentwicklung werden auf ein „extremes“ Level gehoben, das die Programmiertätigkeit und den Code ins Zentrum der Entwickler rückt. Prozesse und Dokumentation rücken in den Hintergrund (Schmidt 2016). Zu den Methoden zählen unter anderem:

- Pair Programming: Der Code wird zeitgleich durch zwei Entwickler geschrieben. Ein Entwickler schreibt den Code, der andere unterstützt und versucht eine besser Möglichkeit zur Problemlösung zu finden. Dadurch soll die Qualität der Software steigen.
- Test-driven Development: Bei diesem Ansatz wird erst ein Testfall geschrieben. Anschließend wird der Code geschrieben und mit dem Testfall überprüft, ob die gewünschte Funktionalität erzielt wurde.
- Coding Standards: Die Entwicklung eines Teams erfolgt auf Basis von Richtlinien, die vor dem Projektstart festgelegt wurden. Dadurch soll die Qualität und Lesbarkeit des Codes verbessert werden
- Automated Testing: Bei dieser Methode soll der Code iterativ und automatisiert mittels festgelegter Programme getestet werden. Die Testfälle sind so geschrieben, dass die Software gegen ein erwartetes Ergebnis getestet wird. Die Software kontinuierlich und auf unterschiedlichen Ebenen getestet. Als Output wird so geprüft, ob das erwartete Ergebnis erreicht wird oder nicht.

Anders als Extreme Programming ist Scrum keine Entwicklungsmethode, sondern ein Rahmenwerk für die agile Lösung komplexer Aufgaben innerhalb eines Entwicklungsteams (Schwaber und Sutherland 2020). Dabei geht Scrum auf Rollen innerhalb eines Teams, Arbeitsabläufe und Koordinationsmöglichkeiten ein wie in Abbildung 3.3 dargestellt. Ein weiteres Rahmenwerk ist das Scaled Agile Framework (SAFe), welche ähnliche Teamstrukturen und Rituale wie Scrum verwendet. Im Folgenden wird vermehrt auf SAFe eingegangen.

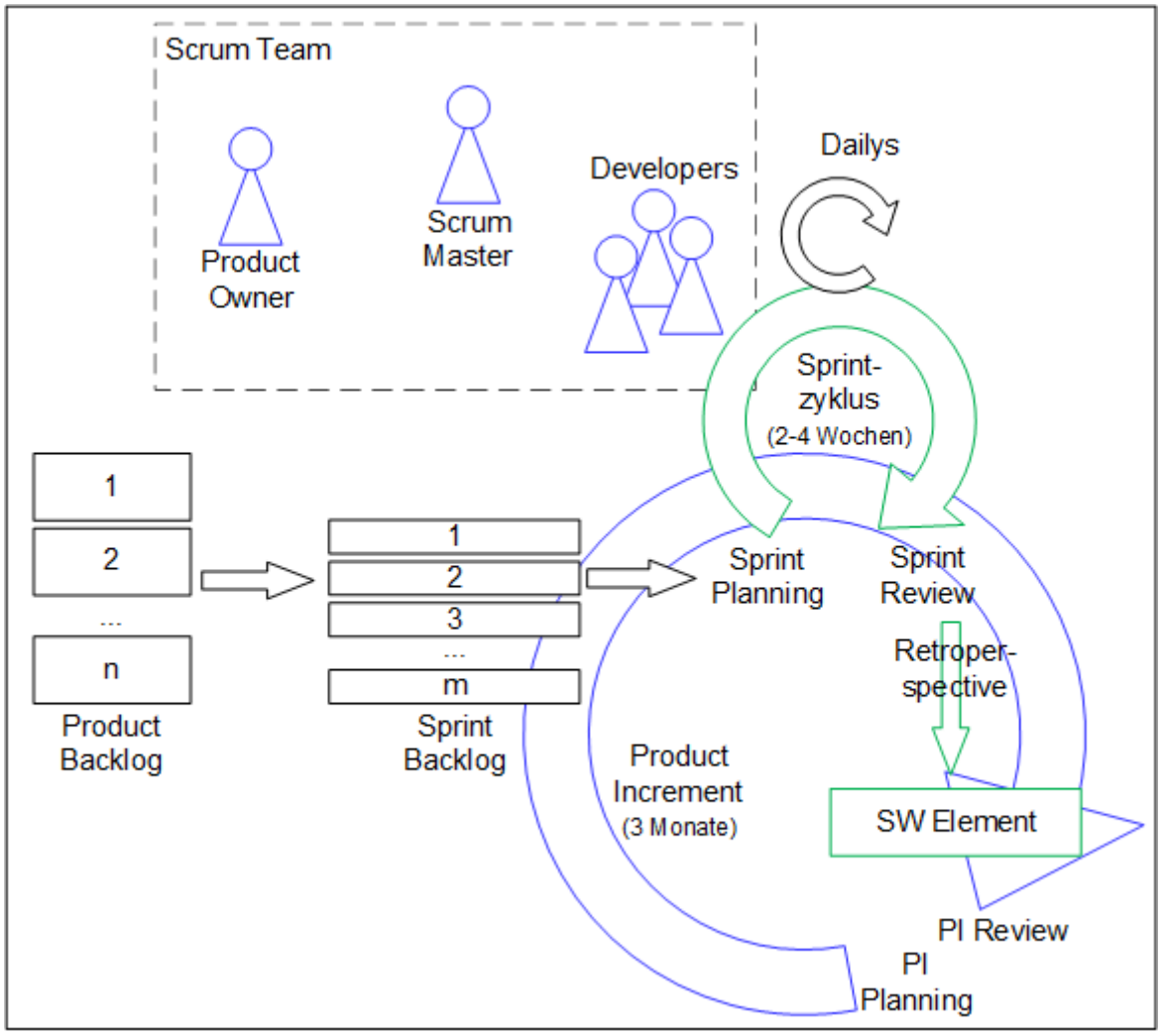

Abbildung 3.3: Rahmenwerk für das Projektmanagement nach Scrum/SAFe (modifiziert nach Schmidt 2016)

Das selbstorganisierte Scrum Team, welches auf die Idee von Takeuchi und Nonaka (1986) zurückgeht, besteht typischerweise aus circa zehn Personen, die drei verschiedene Rollen innehaben. Der Scrum Master (SM) ist verantwortlich für die Einhaltung der Prozesse und Abläufe. Der Prodcut Owner (PO) verantwortet innerhalb des Scrum-Teams die Entwicklungsziele. Die Aufgaben dieser Rolle umfassen u. a. die Priorisierung von Arbeitsaufträgen, die Definition von Anforderung oder die Überprüfung der Arbeitsergebnisse. Alle anderen Scrum-Teammitglieder (Developers) sind für die Entwicklung des Produktes verantwortlich. In ihrer Summe sollten alle Teammitglieder alle Fähigkeiten besitzen, um die Entwicklung eines Softwareinkrements erfolgreich abzuschließen, sodass keine weiteren Spezialisten, z. B. Tester benötigt werden. (Schmidt 2016; Schwaber und Sutherland 2020)

Die Ziele und Aufgaben eines Scrum Teams werden in einem Backlog festgehalten. Der PO verwaltet das Backlog und fügt neue Aufgaben hinzu. Zur Abarbeitung werden sogenannte Sprints durchgeführt. Ein Sprint ist ein definierter Zeitraum von zwei bis vier Wochen. Zu Beginn findet das Sprint Planning statt, bei dem das Team definiert, welche Entwicklungs-

aufgaben im Sprint durchgeführt werden sollen und welches Teammitglied welche (Teil-)Aufgaben übernimmt. Während des Sprints synchronisiert sich das Team in Dailys. In diesen kurzen Meetings wird der Stand präsentiert und sichergestellt, dass die Arbeitsergebnisse nicht voneinander abweichen. Am Ende eines Sprints findet das Sprintreview statt, während derer dem PO die Ergebnisse präsentiert und der aktuelle funktionierende Stand der Software übergeben werden. Zur Fortschrittsmessung wird das Produkt und damit verbundene Nutzererlebnis verwendet. Weitere Zahlen können und sollen zwar auch erhoben werden, beispielsweise entstandene und noch geplante Aufwände stellen aber nicht den Mittelpunkt des Reportings dar. (Böhm 2019; Schmidt 2016; Schwaber und Sutherland 2020)

3.2.2 Agile Methoden in der Fahrzeugentwicklung

Klassische Maschinenbauunternehmen wie auch Fahrzeughersteller setzen heute noch stark auf starre, zu Beginn eines Fahrzeugprojektes erhobene Meilensteinpläne. Dieser Ansatz funktioniert für die Hardwareentwicklung erfahrungsgemäß gut, da man über viele Jahrzehnte ausreichend Erfahrung sammeln konnte. Klassische Projektpläne sind so gestaltet, dass bereits zu Beginn der Planung ein umfassendes Wissen über das zu entwickelnde Produkt vorhanden ist. Bei der Softwareentwicklung kommt es hingegen häufig zu unvorhersehbaren Ereignissen und bedarf somit anderer Vorgehensmodelle. Dies bedeutet, dass die Abweichung zwischen Plan und Realität bedeutend größer sein kann als bei der Hardwareentwicklung. Für die Softwareentwicklung sehen viele Manager die Notwendigkeit der Agilisierung der Entwicklung. Ettelbrück (2021) prangert an, dass führende Personen in Unternehmen vordergründig angeben, die Unternehmen „in ein agiles Zeitalter mit einer digitalen Prozesslandschaft und flexiblen Organisation führen“ wollen, in der Realität jedoch auf bestehenden Strukturen und Vorgehensweisen beharren. (Ettelbrück 2021; Schrof und Paetzold 2019)

Als Ursache wird oftmals auf bestehende Abläufe, die bisher gut funktioniert haben oder bestehende Lieferantenbeziehungen verwiesen. Es sind aber auch Ansätze von agilem Handeln erkennbar. So werden in einigen Entwicklungsprojekten agile Methoden eingesetzt, oder das Projektmanagement ist agil organisiert. Die Einführung von vollumfänglichen agilen Arbeitsweisen und Methoden bei einem Automobilhersteller stellt noch immer eine massive Herausforderung dar. So haben Hardwarebauteile immer auch physikalische Eigenschaften, die als Baugruppe nur in Verbindung funktionieren können. Dies erschwert die inkrementelle Auslieferung einzelner Bauteile erheblich, da die Verbindungen zweier Bauteile stets zusammenpassen müssen. Bei der Einführung von agilen Vorgehensweisen ergeben sich nach der Literatur deshalb zusätzliche Herausforderungen:

- Die Planung und Koordination der Entwicklung.
- Teilung der Entwicklungsaktivitäten in einzelne Aufgaben für einen Sprint.
- Priorisierung der Entwicklungsaufgaben.

- Abschätzung des zeitlichen Aufwandes für einzelne Entwicklungsschritte; v. a. Abschätzung der Arbeitsstunden anstelle von Arbeitswochen.
- Entwicklung separater Ablieferprodukte pro Sprint und anschließende Integration zu einem Gesamtprodukt.
- Sicherstellung, dass die Qualität über die Vielzahl der beteiligten Teams gleichbleibend ist.
- Verteilung und Synchronisation von Wissen.
- Einrichtung von Flexibilität innerhalb der Produkte und Prozesse.

(Ovesen 2012; Schrof und Paetzold 2019; Sekitoleko et al. 2014)

Zusätzlich müssen bei der Fahrzeugentwicklung viele Richtlinien, Normen und Gesetze berücksichtigt werden. Diese erfordern oftmals eine umfangreiche Dokumentation der Entwicklungsaufgaben oder geben teilweise Prozessschritte und Ablieferprodukte vor. Wie in 1.1.2 beschrieben, gibt es immer mehr solcher Vorgaben, die erfüllt werden müssen. Die Anforderungen aus diesen stehen häufig im Widerspruch zum agilen Manifest. Insbesondere eine umfangreiche Dokumentation und starre Zeitpläne stehen im Zielkonflikt zu einer Konzentration auf das Produkt. Die Erfüllung der Anforderungen aus Richtlinien, Normen und Gesetzen steigert die Schwierigkeit der Einführung von agilen Methoden erheblich und fördert die Beibehaltung starrer Produktentstehungsprozesse.

3.3 Softwaretests und Testprozesse

Sowohl im V-Modell als auch bei der agilen Entwicklung zählen Tests zu einer Haupttätigkeit der Entwicklung. Die Absicherung von Software folgt in beiden Vorgehensmodellen meist formalen Vorgaben, die auf detaillierten und umfangreichen Testprozessen aufbauen. Diese fußen in großen Teilen auf dem fundamentalen Testprozess, wie er vom International Software Testing Qualification Board (ISTQB) beschrieben wird (ATB et al. 2020). Softwaretests verfolgen stets dieselben Ziele: Einerseits das frühzeitige Erkennen von Fehlern in der Software und andererseits die Informationsbeschaffung. Die Informationen werden genutzt zur Qualitätssicherung bzw. Qualitätssteigerung in der Softwareentwicklung, zur Bewertung des Restrisikos im Verhältnis zum Testaufwand und zur Minimierung des Risikos bei auftretenden Fehlerwirkungen für Anwender bei schlechter Produktqualität. Die erhobenen Informationen können im Projekt auch zur Entscheidungsfindung herangezogen werden oder zur Verbesserung der Robustheit bei Fehlanwendung, umso das Vertrauen in die Software zu erhöhen. (ISO/IEC/IEEE 29119-1:2013; ATB et al. 2020)

Bei der Festlegung der Ziele und der Überprüfung, ob sie erreicht wurden, helfen sieben Grundsätze bzw. Leitlinien des Softwaretestens.

1. Durch Testen werden Fehlerzustände nachgewiesen, der Umkehrschluss ist jedoch nicht möglich, da exzessives Testen lediglich die Wahrscheinlichkeit senkt, dass unentdeckte Fehlerzustände noch im Testobjekt vorhanden sind, es wird jedoch nicht nachgewiesen, dass keine Fehlerzustände vorhanden sind.
2. Es ist unmöglich, komplexe Softwareelemente vollständig zu testen. Testen ist eine stichprobenartige Überprüfung des Testobjektes. Der Aufwand, jede erdenkliche Kombination aus Eingabewerten zu testen, übersteigt das technisch Mögliche. Daher sollte der geplante Testaufwand in Abhängigkeit vom Risiko bestimmt werden.
3. Mit dem Testen sollte so früh wie möglich begonnen werden und jeder Testfall sollte ein zuvor festgelegtes Ziel verfolgen.
4. Die Teile der SW, in denen am meisten Fehlerzustände erwartet werden, sollen verstärkt getestet werden.
5. Wiederholungen von Tests bringen keinen neuen Informationsgewinn. Deshalb müssen Testfälle regelmäßig überprüft und angepasst werden. Oftmals ist es sogar sinnvoll, neue Testfälle zu erstellen, um alle Teile der Software zu testen.
6. Tests müssen passend zum Einsatzzweck der Software sein.
7. Durch Testen kann man nicht nachweisen, dass Software auch nutzbar ist. Fehlerzustände können identifiziert und beseitigt werden, das bedeutet jedoch nicht, dass die Software dadurch für Anwender nutzbar wird.

Die Ziele und Leitlinien helfen bei der Planung der Testdurchführung, sind jedoch nicht ausreichend. Vielmehr ist es notwendig, eine effiziente und effektive Gestaltung aller Aktivitäten, die mit dem Testen zusammenhängen, zu ermöglichen. Eine Grundlage bilden Testprozesse, die ein Entwicklungsprojekt von Projektstart bis Projektende begleiten. Der fundamentale Testprozess stellt einen generischen Rahmen dar und wird im Folgenden vorgestellt.

3.3.1 Fundamentaler Testprozess nach ISTQB

Der fundamentale Testprozess ist generisch gestaltet und beschreibt den Test unabhängig von Produktentstehungsprozessen, dem V-Modell oder agilen Methoden. Während der Softwareentwicklung müssen die einzelnen Prozessschritte unterschiedlich oft wiederholt werden oder gar nur einmal durchgeführt werden. Deshalb sollten die einzelnen Phasen parallel zur Entwicklung nach dem V-Modell stattfinden. Die VDI Richtlinie VDI 2221 ist eine Richtlinie zur Entwicklung mechatronischer Bauteile und muss für verschiedene Entwicklungsobjekte interpretiert werden. Der fundamentale Testprozess kann damit verglichen werden. Auch dieser ist Vorgabe, die zur Anwendung an konkrete Projekte oder Organisationen angepasst werden muss.

In der Version 2011 des ISTQB Lehrplans Foundation Level wird der Prozess mit fünf Schritten, wie in Abbildung 3.4 gezeigt, dargestellt. In der Version 2018 werden diese in sieben

Schritte geteilt. Dabei werden die ersten drei Schritte jeweils geteilt und die letzten beiden zusammengefasst, sodass es in der Realität zu keinen Unterschieden kommt. Im Folgenden werden die einzelnen Prozessschritte kurz erläutert. (ATB et al. 2011, 2020).

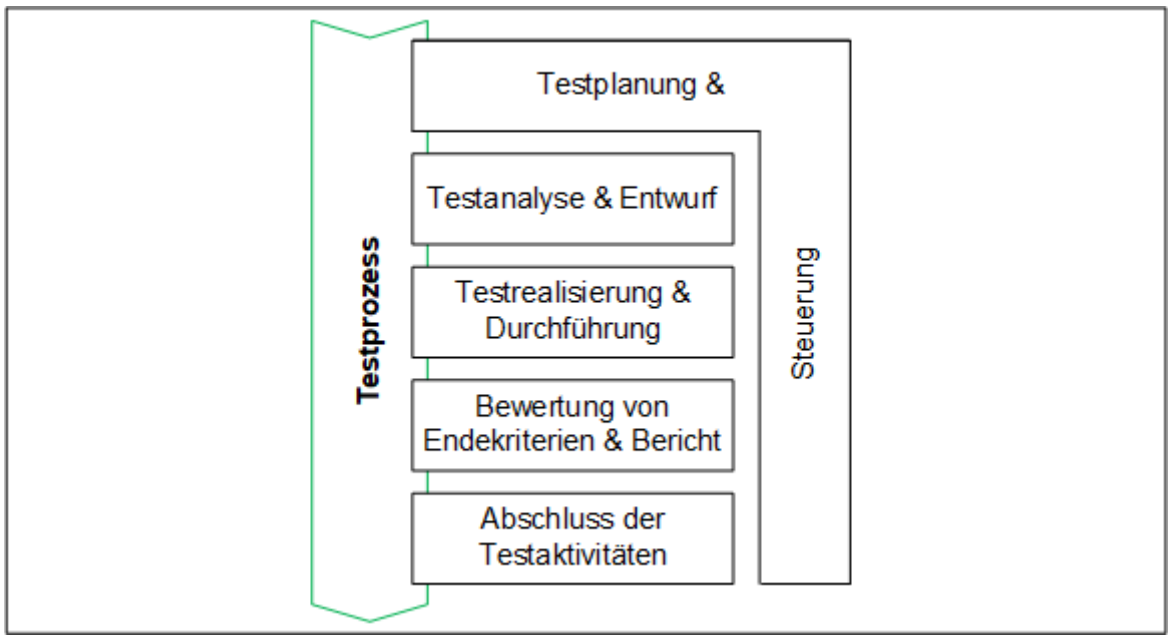

Abbildung 3.4: Der fundamentale Testprozess nach ISTQB (modifiziert nach ATB et al. 2011)

Der erste Prozessschritt besteht in der Testplanung. Als Input dient die Teststrategie (Abbildung 3.5), welche für ein spezifisches Projekt angepasst werden muss. Für einzelne Projekte wird während der Testplanung die exakte Vorgehensweise des Testens bestimmt. Dies umfasst u. a. die Festlegung von Testzielen, Testeingangs- und Testendekriterien, die Planung der weiteren Aktivitäten des fundamentalen Testprozesses und die Planung der notwendigen Ressourcen. Die Testplanung wird zu Beginn eines jeden Projektes durchgeführt. Unter Teststeuerung wird der gesamte Testzeitraum verstanden. Während dieser Phase wird der Testfortschritt überwacht und gegen die Testplanung abgeglichen. D. h. dass der Ist-Status mit dem Soll-Status verglichen wird. Dabei unterstützen Testfortschrittsdaten. Bei Abweichungen müssen entsprechende Maßnahmen definiert, festgehalten und gesteuert werden.

Im weiteren Prozessverlauf werden allgemeine Testziele in konkrete Testbedingungen und Testfälle umgewandelt. Für jede Teststufe wird die Testinstanz festgelegt, die Testbasis, also das Testobjekt, die Anforderungen, die Architektur etc. analysiert, um Testfälle abzuleiten.

Anschließend werden die gesammelten Informationen dazu verwendet, die spezifizierten Testfälle auf der Testumgebung zu implementieren und durchzuführen. Einzelne Testfälle werden priorisiert, um eine Rheinfolge festzulegen und die Konfiguration der Testumgebung wird abgeschlossen. Anschließend erfolgt die Testdurchführung unter Berücksichtigung der Testpläne und der priorisierten Reihenfolge. Zudem werden die Ergebnisse mit den erwarteten Ergebnissen abgeglichen, umso Fehler aufdecken und analysieren zu können. Im Fehlerfall müssen Fehlernachtests nach Beseitigung der identifizierten Auffälligkeit durchgeführt werden.

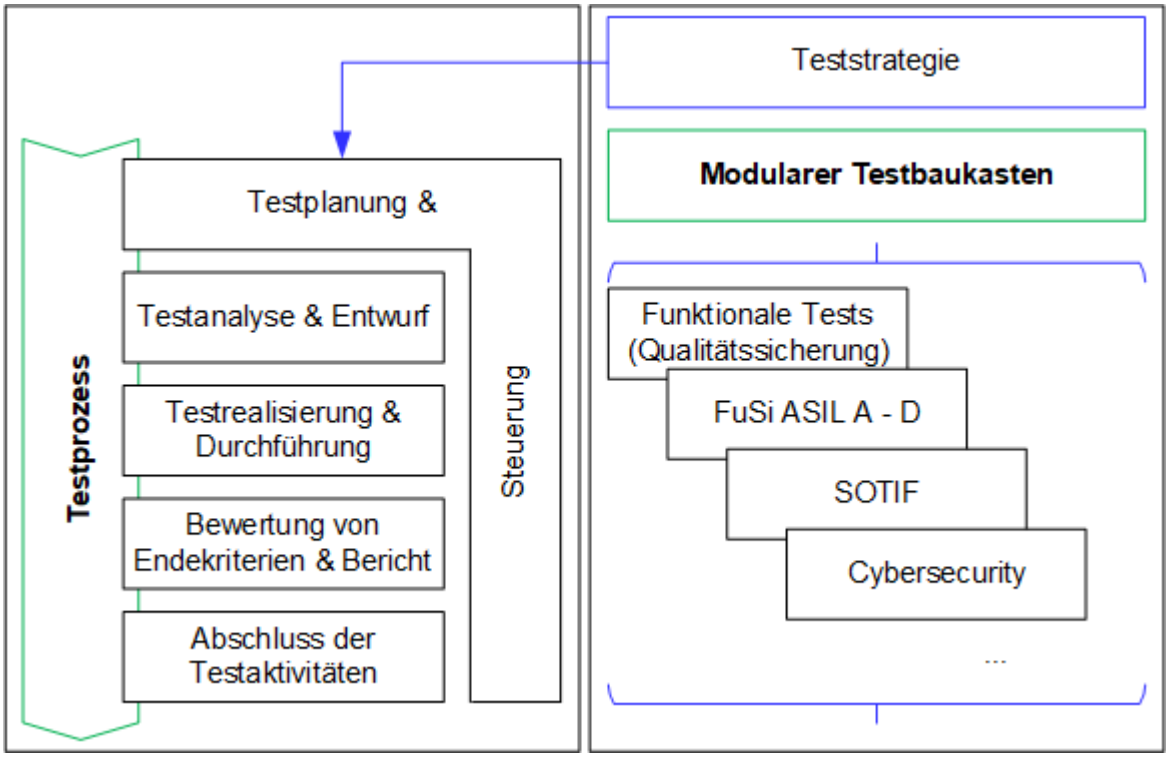

Abbildung 3.5: Die Teststrategie als Input für den fundamentalen Testprozess (modifiziert nach Jooß und Schramm 2022)

Im vierten Prozessschritt werden die Ziele der Testaktivitäten auf ihre Zielerfüllung hin analysiert. Dies bedeutet, dass die Testprotokolle der Testdurchführung mit den Testzielen aus der Testplanungsphase verglichen werden, um zu entscheiden, ob noch weitere Tests durchgeführt werden müssen oder eine Anpassung der Testendekriterien notwendig ist. Sollten keine weiteren Tests notwendig sein, werden alle Informationen zu einem Testbericht zusammengefasst.

Der letzte Prozessschritt ist der Testabschluss. Dieser Schritt erfolgt i. d. R. zu Meilensteinen des Gesamtprojektes. Alle Informationen aus den vorangegangenen Prozessschritten werden gesammelt und die übergeordneten Ziele mit dem Status abgeglichen. Zudem wird hier der Übergang ins Änderungsmanagement geschaffen, was die Dokumentation der Fehler dokumentiert und das Stellen notwendiger Änderungsanträge umfasst. Des Weiteren wird der Stand der Testbasis und Testumgebung archiviert, um diese zu einem späteren Zeitpunkt wiederholen zu können. Alle gesammelten Informationen werden im Rahmen eines „lessons learned“ aufbereitet, um die einzelnen Prozessschritte zu optimieren.

Als Input für den fundamentalen Testprozess dient in erster Linie eine Teststrategie (Abbildung 3.5). Eine Teststrategie wird definiert als „eine Dokumentation, die die generischen Anforderungen an das Testen in einem oder mehreren Projekten innerhalb einer Organisation beschreibt, einschließlich Details darüber, wie das Testen durchgeführt werden soll […]“ (ISTQB 2021). Wird die Teststrategie für ein spezifisches (Teil-)Projekt während der Testplanung konkretisiert, spricht man von einem Testkonzept. Solche Strategien werden an Situationen und Strukturen angepasst. Deshalb können Teststrategien verschiedene Schwerpunkte

für eine Organisationseinheit setzen. Es wird beispielsweise zwischen prozesskonformen oder standardkonformen Teststrategien unterschieden. Bei prozesskonformen Teststrategien folgt das Testteam bereits definierten Prozessen und versucht deren Einhaltung zu gewährleisten. Standardkonforme Teststrategien basieren hingegen auf Standards, beispielsweise Normen, Gesetzen oder auch betriebsinternen Vorgaben. In Abschnitt 1.1 wurde bereits darauf eingegangen, dass die Fahrzeugherstellung heute durch eine Vielzahl von Gesetzen reguliert wird und diese Regelungen bei der Entwicklung berücksichtigt werden müssen. Dies führt dazu, dass verschiedene Teststrategien parallel entwickelt und eingehalten werden müssen (Abbildung 1.4). Da Testaktivitäten stets dem Zielkonflikt zwischen Effizienz (kurze Durchführungsdauer) und Effektivität (Fehleridentifizierungsrate) gerecht werden müssen, ist dieses Vorgehen für jede einzelne Regelung, die für die Fahrwerkentwicklung relevant ist, nicht realisierbar. Deshalb ist es notwendig, modulare Teststrategien zu entwickeln. Wie in Abbildung 3.5 dargestellt, ist es das Ziel dieser Arbeit, einen solchen modularen Ansatz als normkonforme Teststrategie für die Fahrwerkentwicklung zu entwickeln. (Daigl und Glunz 2016; ISTQB 2021; Jooß und Schramm 2022)

3.3.2 Relevante Richtlinien und Normen

Bei der Gestaltung und Auslegung von Testprozessen und Teststrategien orientieren sich Testmanager an verschiedenen Richtlinien und Normen. Zentral ist der Lehrplan des ISTQB. Dieser dient der Ausbildung von SW-Testern mit dem Ziel, ein einheitliches Verständnis von Softwaretest zu schaffen (ISTQB 2022). Der Lehrplan, der auch als De-facto-Standard angesehen wird, orientiert sich an der ISO/IEC/IEEE 29119 Software and systems engineering - Software testing. Die Normenreihe besteht aus insgesamt fünf Teilen, die sich wie folgt gliedern:

- Teil 1: Konzepte und Definitionen (ISO/IEC/IEEE 29119-1:2013)
- Teil 2: Testprozesse (ISO/IEC/IEEE 29119-2:2013)
- Teil 3: Testdokumentation (ISO/IEC/IEEE 29119-3:2013)
- Teil 4: Testverfahren (ISO/IEC/IEEE 29119-4:2015)
- Teil 5: Keyword-Driven Testing (ISO/IEC/IEEE 29119-5:2016)

Die Normenreihe löste einige bis dahin gültige Normen zum Testen von Software ab, darunter z. B. die IEEE 829 (IEEE Standard for Software and System Test Documentation), die BS 7925-1 (Vocabulary of Terms in Software Testing) und BS 7925-2 (Software Component Testing Standard). Die Gründe für die Neueinführung sind u. a. die Aktualisierung der bestehenden Normen sowie die einheitliche Strukturierung und inhaltliche Zusammenfassung des Bestehenden. Der für diese Arbeit wichtigste Punkt ist jedoch die Prozessorientierung der ISO/IEC/IEEE 29119. (Daigl und Glunz 2016)

Inhaltlich beschäftigt sich die Normenreihe ausschließlich mit dem Thema Testen und gibt Handlungsempfehlungen zum Entwurf von Testprozessen und Testaktivitäten. Darüber hinaus erörtern Daigl und Glunz (2016) den inhaltlichen Bezug zu weiteren Normen.

So ist die ISO 26262 (Road vehicles - Functional safety) eine Normenreihe speziell für sicherheitskritische mechatronische und elektronische Funktionen in der Fahrzeugentwicklung. In dieser Normenreihe (insbesondere Teil 6 Product development at the softwarelevel) werden neben verschiedenen Teststufen auch Testmethoden in Abhängigkeit von Automotive Safety Integrity Leveln (ASIL) gefordert. Diese Methoden und deren Anwendungen werden wiederum in der ISO/IEC/IEEE 29119-4:2015 beschrieben und so eine klare Abhängigkeit zwischen den Normen geschaffen. (Daigl und Glunz 2016; ISO 26262-6:2018; ISO/IEC/IEEE 29119-4:2015)

Eine weitere Norm, deren Zusammenhang mit der ISO/IEC/IEEE 29119 von Daigl und Glunz (2016) erörtert wird, ist die ISO/IEC/IEEE 12207:2017 (System and software engineering – Software life cycle processes). Diese Norm beschreibt Prozesse des Softwarelebenszyklus. Worunter auch der Prozess zur Verifizierung bzw. Validierung von Software, welche starke Anforderungen an das Softwaretesten stellt, fällt. Diese Anforderungen greift die ISO/IEC/IEEE 29119 auf und leitet daraus Handlungsempfehlungen ab.

Eine weitere, im Zusammenhang zu Prozessen und Automobilentwicklung relevante Regelung, auf die die Autoren allerdings nicht eingehen, ist ASPICE (Automotive Software Process Improvement and Capability Determination) (VDA QMC Working Group 13 / Automotive SIG 2017). ASPICE basiert auf der branchenunabhängigen Normreihe ISO/IEC 330xx:2015 (Information technology - Process assessment), die als oberstes Ziel die Verbesserung von Softwareentwicklungsprozessen hat. In der Norm wird einerseits auf die Prozessdimension und andererseits auf die Reifegraddimension eingegangen. Die Prozessdimension gibt vor, welche Prozesse eingehalten werden müssen. Große Teile davon werden der ISO/IEC 20246:2017 beschrieben. Über die Reifegraddimension wird angegeben, zu welchem Grad die einzelnen Stufen erfüllt werden. (VDA 2022). Im Rahmen dieser Arbeit sind in besonderem Maße die Prozesse für Validierung und Verifikation auf Software- und System-Ebene relevant. Dies sind:

- SWE.4: Software Unit Verification
- SWE.5: Software Integration und Integration Test
- SWE.6: Software Qualification Test
- SYS.4: System Integration and Integration Test
- SYS.5: System Qualification Test

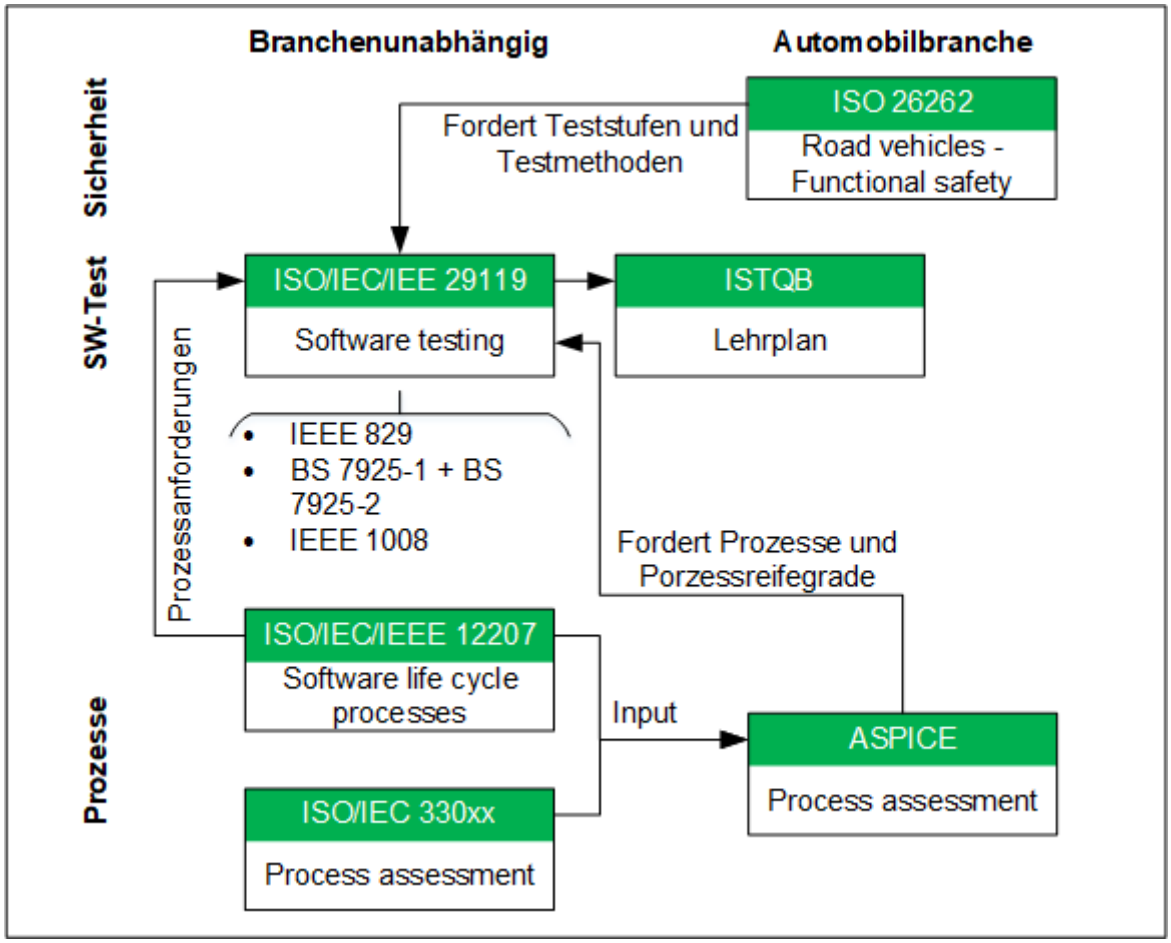

Abbildung 3.6: Normen und Standards in Zusammenhang mit der ISO/IEC/IEEE 29119

Abbildung 3.6 bringt die für dieser Arbeit relevanten Normen in einen grafischen Zusammenhang. Dies stellt die momentan etablierten Normen in der Automobilentwicklung dar. Da in Zukunft immer mehr Normen und Gesetze die Entwicklung regulieren werden (Abschnitt 1.1.2) können an dieser Stelle nicht alle aufgelistet werden. Weitere für diese Arbeit relevante Normen werden im weiteren Verlauf vorgestellt.

3.3.3 Testen im Produktentstehungsprozess

Im PEP werden verschiedene Meilensteine definiert, die auch für die Softwareentwicklung Gültigkeit haben. Daher muss der fundamentale Testprozess stets so ausgelegt sein, dass der Testabschluss zeitlich zum Meilenstein, zu dem er gefordert wird, abgeschlossen ist. Wie bereits diskutiert, beinhaltet die Softwareentwicklung im Vergleich zur Hardwareentwicklung jedoch viele Unbekannte, wodurch die starren Zeitpläne nicht zu den Entwicklungsabläufen passen. Trotzdem gibt es unzählige Ansätze, bei denen mit Hilfe des V-Modells die Softwareentwicklung in den gleichen Zyklen wie die Hardwareentwicklung stattfindet.

Dieses Vorgehen wird beispielsweise durch ASPICE dahingehend unterstützt, dass die geforderten Prozesse entlang des V-Modells angeordnet sind (VDA QMC Working Group 13 / Automotive SIG 2017). Die fünf in 3.3.2 genannten Prozesse aus ASPICE sind am rechten Ast

des V-Modells angeordnet. Die einzelnen Softwareelemente werden zu immer größeren Teilen integriert und getestet (Schäuffele und Zurawka 2013). Einen ähnlichen Aufbau kann man auch in anderen Normen wie der ISO 26262 beobachten. Ein wesentlicher Unterschied zwischen den Normen ist die Bezeichnung für diese Stufen. Für diese Arbeit gelten die in Abschnitt 3.1 gemachten Einschränkungen auf die Teststufen der reinen Softwareentwicklung. Die Prozesse aus anderen Normen werden an diesen angelehnt.

Diese Integrations- und Testschritte müssen alle vor den Projektmeilensteinen aus dem PEP abgeschlossen sein. Folglich werden in der Planungsphase des Testprozesses diese Testschritte berücksichtigt. Auch eine vorgelagerte Teststrategie muss auf mehrere verschiedene Teststufen ausgelegt sein.

Durch diese starre Auslegung der Prozesse im PEP werden die einzelnen Teststufen und damit die Identifizierung und Optimierung von Auffälligkeiten langwierig. Dadurch können während der Entwicklung Kosten entstehen, die durch schnelle und dynamische Abläufe vermeidbar sind (Schäuffele und Zurawka 2013). Dementsprechend ist eine Agilisierung der Prozesse notwendig, die es ermöglicht, kurze Testzyklen für entwicklungsbegleitende Tests zu etablieren und gleichzeitig die Meilensteine des PEPs unter Berücksichtigung der normativen Anforderungen zu erreichen.

3.3.4 Testarten und Testmethoden

Um einen möglichst vollständigen Testprozess abbilden zu können, ist neben der Identifizierung der richtigen Teststufen auch der gezielte Einsatz von Testarten und Testmethoden ein wesentliches Element.

Testarten werden in der Literatur beschrieben als mehrere Testaktivitäten, die zusammen ein Testziel erfüllen oder aufgrund einer bestimmten Ursache durchgeführt werden. Sobald wegen einer Ursache, i.d.R. einer Änderung der SW, getestet wird, spricht man von änderungsbasierten Tests. Wird hingegen ein bestimmtes Testziel verfolgt, werden die Tests unterschieden nach funktionalen, nicht-funktionalen und strukturbasierten Tests. Funktionale Tests zielen darauf ab, die gewünschte Funktion der Software zu bewerten und basieren häufig auf funktionalen Anforderungen oder Spezifikationen. Nicht-funktionale Tests sollen hingegen einen Nachweis zur Anwendungseignung erbringen. Ihnen liegen daher übergeordneten Anforderungen aus Regelwerken, beispielsweise hinsichtlich Qualität, Performance oder auch Security zugrunde. (ATB et al. 2020; Ring 2019)

Strukturbasierte Tests fußen im Gegensatz dazu nicht auf Anforderungen, sondern auf Strukturelementen der Software. Darunter werden u. a. die Architektur, der Code bzw. das Modell (bei modellbasierter SW-Entwicklung) oder Datenflüsse innerhalb der Software verstanden. Das Ziel ist die Überprüfung der Struktur des Testobjektes. Dieser Nachweis erfolgt häufig über Überdeckungselemente, bei denen nachgewiesen wird, zu welchem Teil ein SW-Element durch Tests abgedeckt wurde. (ATB et al. 2020; Ring 2019)

Eine weitere Dimension des Testens sind Testmethoden. Es wird zwischen Methoden, bei denen die Ausführung des Testobjekts notwendig ist und Methoden, bei der keine Ausführung stattfindet, unterschieden. Sobald das Testobjekt ausgeführt wird, spricht man von dynamischen, im anderen Fall von statischen Testmethoden.

Dynamische Testmethoden können weiter unterteilt werden in objektorientiert und erfahrungsorientiert. In beiden Fällen wird das Testobjekt entweder in einer simulierten oder in einer realen Umgebung ausgeführt. Lediglich die Methode zur Erstellung der Testfälle unterscheidet sich. Bei erfahrungsorientierten Methoden werden Testfälle auf der Grundlage von Wissen und Erfahrungen der Tester und Entwickler erstellt, bei objektorientierten Methoden werden diese in Abhängigkeit vom Testobjekt erstellt. Weitere Unterscheidungen werden zwischen Black-Box, Grey-Box und White-Box Testing gemacht. Diese Methoden beschreiben den Bekanntheitsgrad der Struktur. Bei Black-Box Testing werden Testfälle, ähnlich wie bei funktionalen Tests, Aufgrund der Anforderungen entwickelt. Bei der Entwicklung von White-Box Tests ist hingegen die Struktur des Testobjektes bekannt und zählen somit zu den strukturbasierten Tests. (Spillner und Linz 2019; Ring 2019)

Statische Testmethoden zielen hingegen eher darauf ab, die Qualität der Software mittels statischer toolgestützter Analysen und manuell durchgeführter Reviews zu erhöhen. Bei statischen Analysen wird das Modell oder der Quellcode analysiert und hinsichtlich Fehler und der Einhaltung von Richtlinien überprüft. Dadurch soll eine Aussage über die Qualität der Software getroffen werden. Bei Reviews kann das Testobjekt hingegen unterschiedlich sein. Beispielsweise werden Anforderungen, Testfälle oder Quellcode manuell überprüft und anhand verschiedener Vorgaben eine Aussage über Richtigkeit und Vollständigkeit getroffen. Die Durchführung der Reviews können wenig bis gar nicht oder auch sehr stark strukturiert ablaufen. Die Auswahl der Reviewmethode erfolgt meist aufgrund von Anforderungen aus Gesetzen, Normen oder von Auftraggebern. Beispielsweise erfordern höhere ASIL Einstufungen stärker strukturierte Reviewmethoden. (Spillner und Linz 2019; Ring 2019)

3.4 Freigabeprozesse

Nach dem Abschluss der Entwicklungs- und Testaktivitäten werden einzelne Bauteile, Baugruppen oder Softwareelemente freigegeben. Damit bestätigen alle beteiligten Entwickler und Tester, dass der Entwicklungsgegenstand die zum Meilenstein geforderte Reife erfüllt. Anders als der Testprozess ist dieser Teil der Entwicklung im Automobilbau nicht durch Standards oder andere Rahmenwerke generisch beschrieben, sondern im Laufe der Zeit entstanden und hat sich so zum Stand der Technik etabliert. Trotzdem wird der Begriff der Freigabe in einigen Normen definiert.

3.4.1 Freigaben in der Produktentstehung

So wird der Begriff in der Normreihe DIN 69901 (Projektmanagement - Projektmanagementsysteme) als „Erlaubnis zur Durchführung nachfolgender Arbeiten mit festgelegtem Inhalt“ definiert und geht darüber hinaus auf die Phasenfreigabe ein, welche eine „formalisierte Freigabe zum Abschluss einer Projektphase, um mit der Folgephase beginnen zu können“ darstellt. Das Ziel dieser Art von Freigaben ist es sicherzustellen, dass eine Phase vollständig abgeschlossen ist, bevor die nächste beginnen kann. (DIN 69901-2:2009-01; DIN 69901-5:2009-01)

Andere Normen, z. B. die DIN EN 82045-1:2001(Dokumentenmanagement) und die DIN 6789:2013-10 (Dokumentationssystematik - Verfälschungssicherheit und Qualitätskriterien für die Freigabe digitaler Produktdaten) gehen auf die Freigabe von Dokumenten ein und beschreiben diese als eine formelle Aktion zur Gültigkeitserklärung von Dokumenten für einen festgelegten Zweck. Darüber hinaus findet sich auch eine Definition, die so auch in der Produktentwicklung Anwendung finden kann. Dort wird eine Freigabe definiert als „eine bestimmten Anweisungen entsprechende Genehmigung nach abgeschlossener Prüfung“ (DIN 6789:2013-10) und „formelle Aktion einer autorisierten Person/Organisation, mit der ein Dokument für einen deklarierten Zweck im Prozessablauf für gültig erklärt wird“ (DIN EN 82045-1:2001; DIN 6789:2013-10). Übertragen auf eine Freigabe im PEP bedeutet dies, dass sie stets nach der Prüfung bzw. nach Tests durchzuführen ist.

Allgemein sind Freigaben folglich stets die Bestätigung, dass ein bestimmtes Artefakt für einen vorgegebenen Zweck eingesetzt werden kann. Während der Entwicklung werden überwiegend Entwicklungsartefakte bzw. Produkte freigegeben. Damit wird bestätigt, dass die im Vorfeld für einen Meilenstein definierten Anforderungen und Reifegrade umgesetzt und erfüllt wurden. Um eine Freigabe aussprechen zu können, müssen entsprechende Nachweisdokumente vorhanden sein. Darunter sind unter anderem Testberichte, Prüfprotokolle oder Messergebnisse zu verstehen. (Prefi 2014)

Da Fahrzeuge komplexe Produkte sind, müssen während der Entwicklung Freigaben auf mehreren Ebenen ausgesprochen werden. So werden auf unterster Ebene einzelne SW-Elemente oder Bauteile freigegeben, um deren Reifegrad zu bestätigen. Diese einzelnen Elemente werden zu immer größeren Baugruppen, beispielsweise Fahrwerkfunktionen oder einem Steuergerät integriert, getestet und als Baugruppe freigegeben. Abschließend werden die einzelnen Baugruppen in ein Gesamtfahrzeug integriert. Auf dieser Ebene bestätigt eine Freigabe, dass die Anforderungen, die auf oberster Ebene an ein Fahrzeug gestellt werden, erfüllt sind.

3.4.2 Freigabevoraussetzungen

Die zurückgezogene DIN 6789-5 : 1995-10 hat den Freigabeprozess, wie in Abbildung 3.7 dargestellt, definiert. Obwohl diese Norm zurückgezogen wurde, findet der Prozess in der Praxis so noch Anwendung. Grundlage für eine Freigabe ist demnach stets eine vorangegangene Prüfung. Bereits im Vorfeld muss klar definiert sein, wie solch eine Prüfung aussieht und was damit bestätigt werden soll.

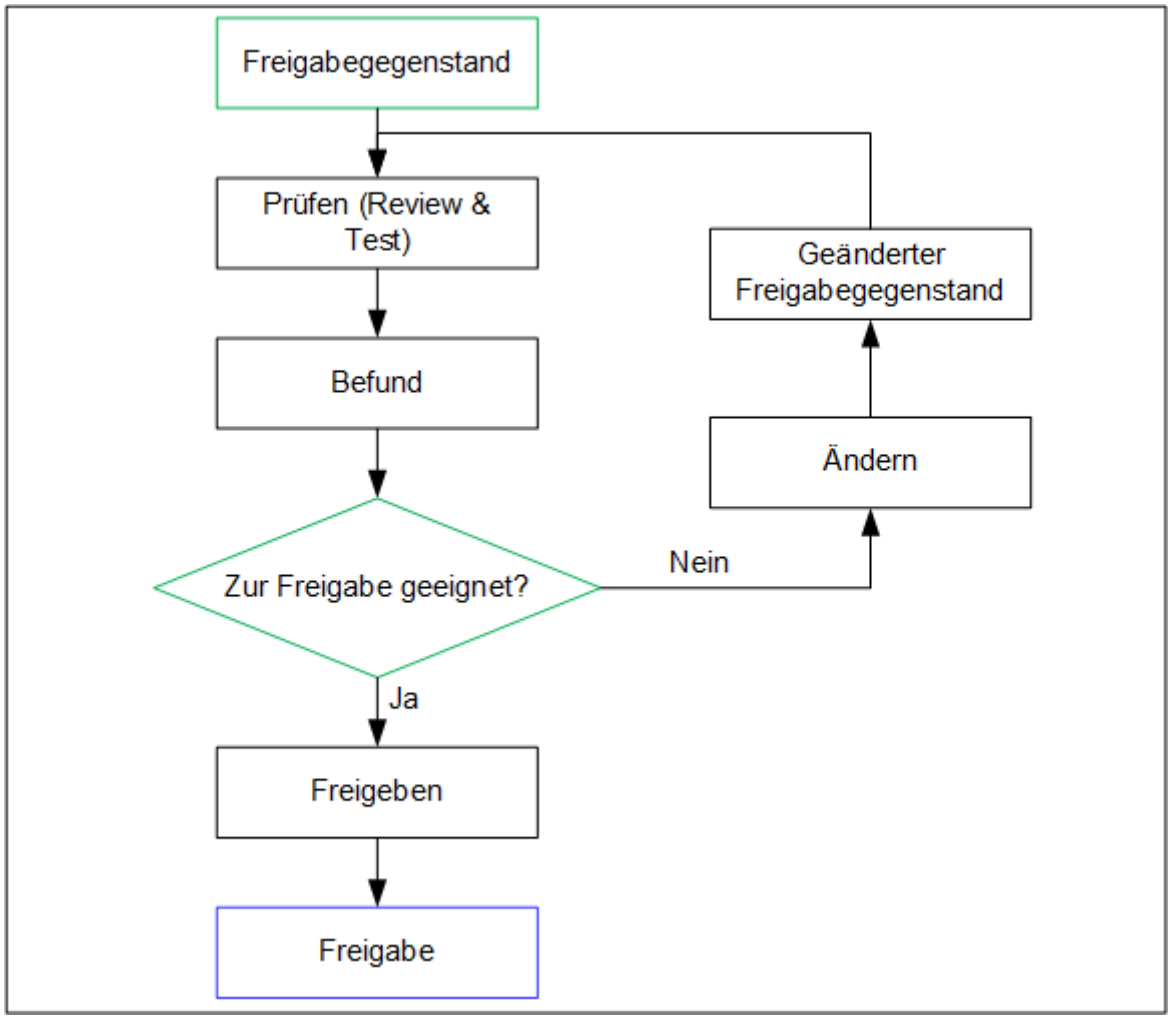

Abbildung 3.7: Freigabeprozess (modifiziert nach DIN 6789-5 : 1995-10)

Für die Freigabe der Software, die für diese Arbeit von Bedeutung ist, heißt das, dass vorgegeben sein muss, zu welcher Freigabestufe welche Tests mit welchem Erfüllungsgrad vorliegen müssen. Die in Abschnitt 3.3.4 beschriebenen Testarten und -methoden werden mit klaren Zielen und Zielwerten verknüpft, um so einen Reifegrad zu bestimmen. Für diese Arbeit bedeutet dies, dass Zielwerte innerhalb eines Testprozesses festgelegt werden müssen, um einerseits Agilität zu ermöglichen und andererseits zur Meilensteinerfüllung beizutragen.

4 Agilisierung der Testplanung

Die effiziente Durchführung von Softwaretests basiert auf einer frühzeitig festgelegten Teststrategie. Diese dient als Grundlage für die Testplanung und sollte so gestaltet sein, dass die Aufwände für Testing in einem gerechtfertigten Maß sind, um die Testziele zu erreichen. Dazu werden in solchen Teststrategien beispielsweise Teststufen und Testendekriterien festgelegt. Allerdings beziehen sich Teststrategien meist auf ein technisches Merkmal und bilden die Komplexität moderner Fahrwerke nicht ab. Deshalb werden in der Praxis oftmals mehrere Teststrategien zeitgleich angewandt. Im Folgenden wird daher eine neuartige Teststrategie vorgestellt, die den Baukastengedanken aufgreift, indem besondere Merkmale und Testaktivitäten zu einer Strategie verknüpft werden. Damit wird sowohl der Technologiezuwachs in der SW-Entwicklung beherrscht als auch die Erfüllung der zunehmenden Anzahl weltweiter Regularien in der Fahrzeugentwicklung.

4.1 Baukasten als Teststrategie

Modularisierung ist in der Fahrzeugproduktion bei steigender Varianz ein wichtiger Beitrag zur Kostensenkung. Durch die Verwendung von Modulen, die in mehreren Fahrzeugtypen eingebaut werden, lassen sich Kostenvorteile durch Skaleneffekte erzielen. Zunächst wurden einzelne Bauteile oder -gruppen in verschiedene Fahrzeug eingebaut. (Renner 2007). In einem nächsten Entwicklungsschritt wurde die Verwendung von gleichen Plattformen für Fahrzeuge der gleichen Fahrzeugklasse etabliert. Dies dient v. a. der Kostensenkung in Mehrmarkenkonzernen wie z. B. dem VW-Konzern. In den 1990er und den frühen 2000er-Jahren wurde u. a. für die Kompaktfahrzeuge VW Golf, Audi A3, Seat Leon und Skoda Octavia die PQ34 Plattform verwendet. (Scholz 2006). Im Jahr 2012 hat der Konzern mit der Präsentation des VW Golf VII den „Modularen Querbaukasten“ (MQB) vorgestellt. Dieser greift den Modularisierungsgedanken verstärkt auf. So stellt der MQB die technische Basis für Fahrzeuge unterschiedlicher Klassen und Größen dar. Durch die Verwendung der gleichen Bauteile für eine möglichst große Anzahl an Fahrzeugen können enorme Kostenpotenziale in der Entwicklung und Produktion erzielt werden. (Buiga 2012; Renner 2007; Wolff et al. 2021)

Dieser Modularisierungsgedanke ist nicht nur in der HW-Entwicklung, sondern ebenso in der SW-Entwicklung relevant, um Synergien zu schaffen und Kostenpotenziale zu heben. Durch die enorme Komplexitätssteigerung, die in den Kapiteln 2 und 3 beschrieben wurde, ist dies ein möglicher Lösungsansatz, um die Effizienz und Effektivität auch im SW-Test zu erzielen. Im Rahmen dieser Arbeit wird ein modularer Testbaukasten für agile Vorgehensweisen entwickelt und vorgestellt. Der Baukasten hilft Testmanagern, indem Testaktivitäten vorgeschlagen werden, die dem Stand der Technik entsprechen.

Ziel ist es, eine einheitliche Teststrategie für mehrere technische Merkmale in der Fahrwerkentwicklung zu entwickeln und etablieren. Folglich sollen die verschiedenen Teststrategien, wie in Abschnitt 1.2 vorgestellt, in eine einheitliche Strategie integriert werden, um die Effizienz bei der Testplanung und Testdurchführung zu steigern. Dies bedeutet, dass die Teststrategie, vergleichbar mit Fahrzeugplattformen, eine Auswahl an Testaktivitäten anbietet, die je nach technischer und gesetzlicher Anforderung an die zu testende Software herangezogen werden können. Im Folgenden wird dieses Konzept als modularer Testbaukasten (MTB) bezeichnet. Beim MQB ist der Abstand zwischen der Radmitte des Vorderrades und des Gaspedales fest vorgegeben (Gerhardt et al. 2022). Übertragen auf den MTB bedeutet dies, dass ein festgelegtes Mindestmaß an Absicherungsaktivitäten definiert ist, um die Qualität jedes Softwareelements zu gewährleisten. Dieses Mindestmaß kann vergleichbar bspw. mit dem variablen Radstand, flexibel kombiniert bzw. verschärft werden, je nach Anforderungen, die sich aus gesetzlicher oder regulativer Sicht ergeben.

In Abbildung 3.5 ist dargestellt, dass der MTB als Input für die Testplanungsphase des fundamentalen Testprozesses dient. Er wird zur Identifikation der richtigen und vollständigen Absicherungsmaßnahmen in einer frühen Planungsphase durch den Testmanager genutzt. Bereits in Jooß und Schramm (2022) ist beschrieben, welche Vorteile ein solcher Baukasten bietet. Insbesondere der Zielkonflikt zwischen Effektivität und Effizienz ist von bedeutender Rolle im Softwaretest. Die SW-Tests sollen möglichst viele Fehler bzw. SW-Bugs in möglichst geringer Zeit identifizieren. Um dies zu erreichen, ist eine aufwendige Testplanung im Vorfeld zu den Tests zielführend. Damit jedoch auch diese Testplanung effizient durchgeführt werden kann, hilft der Testbaukasten bei der Auswahl der entsprechenden Testaktivitäten. Zusätzlich ist gewährleistet, dass gesetzliche Anforderungen abgedeckt werden, ohne dass für jede Testplanung Normen und Gesetze erneut analysiert und interpretiert werden müssen.

In Abschnitt 3.2 wurde erläutert, dass agile Methoden in der SW- und Fahrzeugentwicklung zunehmend zum Einsatz kommen. Im Gegensatz zu klassischen Entwicklungsprojekten ist der Testmanager direkter in das Entwicklungsprojekt eingebunden und soll weniger administrative Aufgaben erfüllen. Jedoch ist die Fahrzeugentwicklung reglementiert durch viele Regularien, die bei der Entwicklung berücksichtigt werden müssen. Um agile Methoden und Arbeitsweisen zu ermöglichen, kann der Testmanager den Baukasten anwenden, umso die notwendigen Testaktivitäten ohne erhöhten administrativen Aufwand zu identifizieren und implementieren.

In Abbildung 4.1 wird der Aufbau des Baukastens schematisch dargestellt. Hauptauswahlkriterium zur Bestimmung der Testaktivitäten sind die technischen Eigenschaften des Testobjektes. Für den Baukasten wurden mehrere besondere Merkmale (BsM) (Abbildung 4.1 [A]), die relevant für die Fahrwerkentwicklung sind, identifiziert. Mehr dazu siehe Abschnitt 4.2. Zudem wurden verschiedene Testaktivitäten zur Absicherung von Software auf Basis des Standes der Technik identifiziert und weiter- bzw. neuentwickelt. Dies wird in Abschnitt 4.3.3 vertieft.

Abbildung 4.1: Struktur des Baukastens (schematische Darstellung)

Da die Durchführung jeder Testaktivität für jedes besondere Merkmal nicht zielführend ist, wurde der Baukasten so ausgelegt, dass jedes BsM mit einer Auswahl an Testaktivitäten getestet wird. Die Auswahl dieser erfolgt einerseits anhand gesetzlicher und normativer Anforderungen, andererseits auf Basis von Effizienz, sodass für jedes BsM die Testaktivitäten genutzt werden, die einen Mehrwert für das Testprojekt bieten. Zur Erreichung dieses Ziels wurden verschiedenste internationale Normen aus dem Umfeld der automobilen Entwicklung analysiert, um daraus Anforderungen an den Softwaretest zu extrahieren. In Abschnitt 4.4 wird dies in den Zusammenhang mit agilen Methoden in der Softwareplanung gebracht, um eine

Anwendung des Baukastens zu ermöglichen. Im Vergleich zum Stand der Technik wird mit dieser einheitlichen Methode nur eine Teststrategie für alle technischen Ausprägungen der Fahrwerkfunktionen benötigt und so die Vergleichbarkeit zwischen den Absicherungsmaßnahmen gewährleistet.

4.2 Besondere technische Merkmale von Fahrwerkfunktionen

Schlüsselelement des Baukastens sind besondere technische Merkmale, welche vom Verband der Automobilindustrie (VDA) im Band „Prozessbeschreibung Besondere Merkmale“ beschrieben werden (Verband der Automobilindustrie e. V. (VDA) 2020). Der VDA Band soll helfen, die Merkmale zu identifizieren, klassifizieren und dokumentieren. Dadurch soll die Sorgfaltspflicht von Herstellern und Lieferanten in den Mittelpunkt gerückt werden. Die besonderen Merkmale bilden eine Teilgruppe aller Merkmale, die ein Fahrzeug hat, bedürfen jedoch einer erhöhten Sorgfalt, um die Qualität sicherzustellen oder zu erhöhen (Abbildung 4.2). Es werden drei Merkmale unterschieden: Sicherheitsanforderungen (BM S), Zulassungsrelevanz (BM Z) und funktionale Anforderungen (BM F). Zu BM S werden alle besonderen Merkmale zusammengefasst, die eine Sicherheitsrelevanz für Insassen und Verkehrsteilnehmer darstellen. Wichtig ist, dass der Zusammenhang zwischen Merkmal und Folge vorhersehbar und wahrscheinlich ist. BM Z gruppiert alle Anforderungen, die aus rechtlicher Sicht entstehen und eingehalten werden müssen, um ein Fahrzeug legal in Verkehr bringen zu können, beispielsweise Vorgaben hinsichtlich Emissionen oder Gewährleistung. BM F zielt hingegen mehr auf die funktionalen Anforderungen an Bauteile und Software, beispielsweise Toleranzen, ab. (Verband der Automobilindustrie e. V. (VDA) 2020)

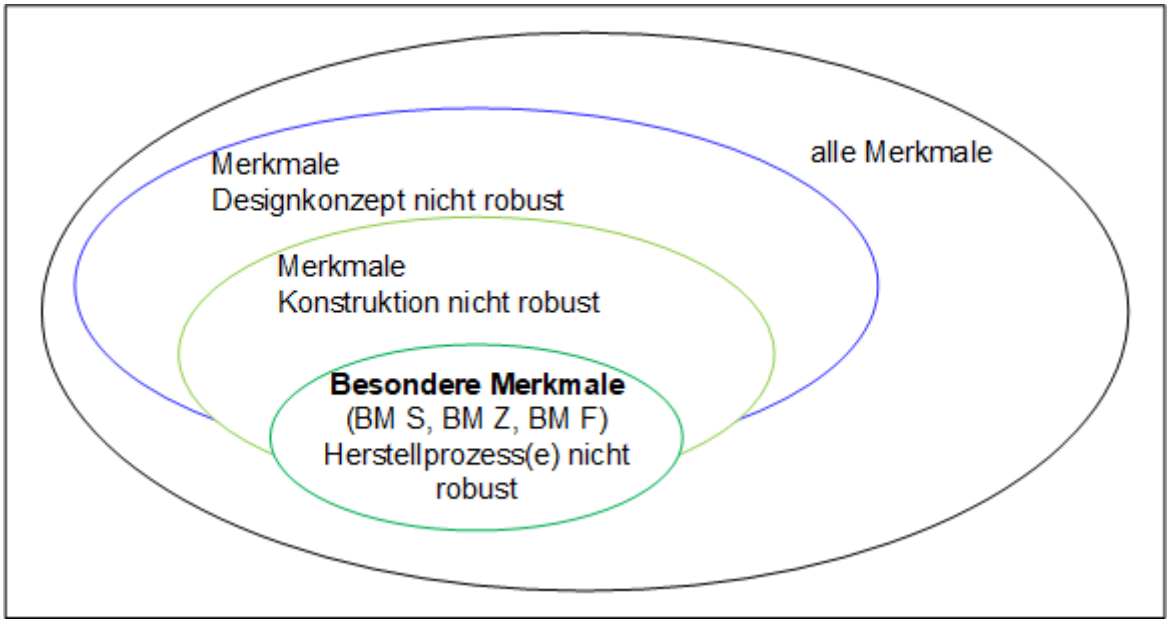

Abbildung 4.2: Menge der Merkmale nach dem VDA (modifiziert nach Verband der Automobilindustrie e. V. (VDA) 2020)

Die Verwendung dieser drei besonderen Merkmale ist in der Praxis anzutreffen, allerdings werden sie häufig weiter spezifiziert. Ein besonders gängiges Beispiel ist die Unterscheidung zwischen funktionaler Sicherheit und Informationssicherheit. Beide Merkmale können BM S zugeordnet werden, haben jedoch einen vollkommen unterschiedlichen Fokus. Unter funktionaler Sicherheit (im englischen „functional safety“) wird die Sicherheit verstanden, die ein Fahrzeug und dessen Funktionen während der bestimmungsgemäßen Verwendung erbringen muss und ist in der Normreihe ISO 26262-x:201x (Road vehicles – Functional safety) geregelt. Informationssicherheit (im englischen Cybersecurity) beschreibt hingegen Schutzmaßnahmen gegenüber Angriffen auf Fahrzeuge mit dem Ziel, das Fahrzeug zu manipulieren oder Daten unerlaubterweise auszulesen. Maßnahmen zum Schutz dieser Daten sind in der ISO/SAE 21434:2021 (Road vehicles – Cybersecurity engineering) beschrieben.

Das Beispiel zeigt, dass eine Unterteilung der vom VDA definierten besonderen Merkmale sinnvoll ist. Effizientes Testen kann nur erfolgen, wenn die besonderen Merkmale im richtigen Maße die technischen Anforderungen widerspiegeln und weder zu umfangreich noch zu granular geschnitten sind. Daher werden zur Entwicklung der BsM für diese Arbeit und der Bewertung der richtigen Granularität folgende drei Punkte untersucht:

- Hardware- und Softwaretechnologien, die in der Fahrwerkentwicklung Anwendung finden.
- Technische Merkmale, die durch Software beeinflusst werden.
- Gültige Regularien, die zur Typzulassung eingehalten werden müssen.

Auch Merkmale, die für die Fahrwerkentwicklung untypisch zu sein scheinen, müssen Berücksichtigung finden. Ein Beispiel dafür sind die sogenannten „Tailpipe Emissions“. Diese beschreiben den Emissionsausstoß am Auspuff. Klassischerweise werden diese von Antriebsfunktionen beeinflusst und waren daher für die Entwicklung und das Testen von Fahrwerkfunktionen unerheblich. Durch den zunehmenden Grad an Vernetzung von Steuergeräten können die Emissionen auch durch eine Fahrwerkfunktionen bzw. Signale einer solchen beeinflusst werden und damit Relevanz für die Fahrwerkentwicklung aufweisen.

Im VW-Konzern werden mindestens 20 BsM unterschieden. Neben Software sind davon auch Hardwarebauteile, Materialien und Stoffe sowie Prozesse betroffen. Da in dieser Arbeit eine Methode zum Testen von Fahrwerkfunktionen entwickelt wurde, sind einige der BsM des VW-Konzerns nicht relevant. Daher wurden die BsM hinsichtlich ihrer Relevanz für diese Arbeit bewertet, überarbeitet und bei Bedarf gruppiert. In Tabelle 4.1 werden die besonderen Merkmale dargestellt. Die Relevanzbewertung für die Fahrwerkentwicklung erfolgte anhand technischer Schlussfolgerungen und unter Einbeziehung verschiedener Experten. Die Begründung ist in der entsprechenden Spalte sachlich dargestellt.

Tabelle 4.1: Bewertung der besonderen Merkmale des VW-Konzerns

Besonderes Merkmal	Relevanz	Begründung
Nachweispflicht	Nein	Strategie berücksichtigt diese Pflichten implizit. Voraussetzung ist, dass alle Test- und Freigabeaktivitäten gesetzeskonform (und unabhängig von BsM) dokumentiert werden.
Kennzeichnungspflicht	Nein	Strategie berücksichtigt diese Pflichten implizit. Voraussetzung ist, dass alle Test- und Freigabeaktivitäten gesetzeskonform (und unabhängig von BsM) gekennzeichnet werden.
Genehmigungspflicht (Typisierung)	Ja	Strategie berücksichtigt diese Pflichten implizit. Voraussetzung ist, dass alle Test- und Freigabeaktivitäten gesetzeskonform (und unabhängig von BsM) typisierbar sind.
Emissionen – AECD/ENDA	Ja	Viele Funktionen können die Abgasreinigung bzw. deren Strategie beeinflussen.
Emissionen - Tailpipe	Ja	Zusammenhängend mit AECD/ENDA.
Emissionen - Leistungsdaten	Ja	Beeinflussung (z. B. cw-Wert, Fahrstrategie) durch Fahrwerkfunktionen gegeben.
Chemische Emissionen	Nein	Chemische Emissionen werden nicht durch Software oder Fahrwerkfunktionen verursacht.
Geräuschemissionen	Nein	Geräuschemissionen sind durch Normen festgelegt und können mit spezifizierten Manövern nachgewiesen werden. Auf diese Manöver haben Fahrwerkfunktionen keinen Einfluss.
Elektromagnetismus	Nein	Anforderungen an Hardware und nicht an die Software von Funktionen.
Fahrwiderstände	Ja	Beeinflussung der Fahrwiderstände (z. B. durch Hochniveau, Lenkung) durch Fahrwerkfunktionen gegeben.
OBD	Ja	Diagnose und Aufleuchten der Motorkontrollleuchte ist aufgrund von fehlerhaften Fahrwerkfunktionen möglich.

Besonderes Merkmal	Relevanz	Begründung
Material Compliance - TOSCA	Nein	Anforderungen an Fluidentwicklung und nicht SW.
Safety - FuSi	Ja	Explizit für mechatronische Systeme und damit auch für Fahrwerkfunktionen.
Safety - Crash	Nein	Crashverhalten und passive Sicherheitselemente werden nicht durch Fahrwerkfunktionen beeinflusst.
Safety – SOTIF	Ja	Explizit für mechatronische Systeme im normalen Anwendungsfall und damit auch für Fahrwerkfunktionen.
Safety - Hochvolt	Nein	HV Sicherheit wird durch VW Konzernnorm „VW80303“ beschrieben. Alle Konzepte sind HW-Konzepte, keine Beeinflussung durch SW möglich (Konzernnorm VW 80303: 2014-06).
Cybersecurity	Ja	Explizit für Softwareanteil mechatronischer Systeme und damit auch für Fahrwerkfunktionen.
Diebstahlschutz	Nein	Gesamtheitliche Betrachtung außerhalb von Fahrwerkfunktionen.
Privacy (DSGVO)	Nein	Gesamtheitliche Betrachtung außerhalb von Fahrwerkfunktionen.

Jedoch sind die relevanten BsM nicht ausreichend für eine Softwareteststrategie, da für die Fahrzeugentwicklung neue und wichtige Trends, die durch Software ermöglicht werden, nicht abgedeckt sind. Diese Softwarefunktionalitäten haben andere Anforderungen an SW-Tests und werden daher als separates Merkmal aufgenommen. Deshalb wurden weitere BsM entwickelt. Diese sind in Tabelle 4.2 dargestellt. Zusätzlich sind auch Hintergründe zur Bedeutung der BsM angegeben.

Tabelle 4.2: Übersicht der ergänzten BsM

Besonderes Merkmal	**Hintergründe**
Over-the-Air-Updates (OTA)	Steigende Bedeutung im Automobilbereich, um einen Wettbewerbsvorteil zu erzielen. Insbesondere im Hinblick auf Sicherheitskritikalität ist dies zu testen und als BsM zu berücksichtigen.
Function as a Service (FaaS)	Steigende Bedeutung im Automobilbereich, um dem Kunden einen Mehrwert anbieten zu können. Insbesondere im Hinblick auf Sicherheitskritikalität ist dies zu testen und als BsM zu berücksichtigen.
Connected functions	Vernetzung im Automobilbereich von erheblicher Bedeutung. Auch Fahrwerkfunktionen sind mit der Umwelt vernetzt und die korrekte Funktionalität muss durch Tests sichergestellt werden und ist als BsM zu berücksichtigen.

Um die bereits angesprochene Granularität zu erzielen, müssen die besonderen Merkmale entsprechend zusammengefasst oder weiter unterteilt werden. Dies ist wichtig, da jedes besondere Merkmal mittels verschiedener Testaktivitäten getestet werden muss. Sollte dies nicht der Fall sein, ist die Granularität nicht gegeben. Diese Informationen flossen in die Entwicklung der BsM für diese Arbeit. Das Ergebnis ist in Abbildung 4.3 dargestellt.

In Abbildung 4.3 werden die besonderen Merkmale des VDAs in zwei Stufen unterteilt. Stufe 1 gibt die Hauptmerkmale an, Stufe 2 die Untermerkmale. Die Notwendigkeit der Unterebene lässt sich am Beispiel der FuSi erläutern. Wäre dies nicht unterteilt, wäre die Absicherung für ASIL A und ASIL D mit denselben Testaktivitäten durchzuführen. Dadurch würde entweder der Aufwand steigen (durch das Testen von ASIL A mit ASIL D Absicherungsmaßnahmen) oder die Vorgaben der ISO 26262 nicht eingehalten (durch das Testen von ASIL D mit ASIL A Absicherungsmaßnahmen). Andererseits ist es auch möglich, dass eine Untergruppierung zwar möglich, aber nicht sinnvoll ist. Dies ist am Beispiel von Remote Updates zu sehen. Sowohl Over-the-Air-Updates als auch Function-as-a-Service nutzen dieselben Technologien und damit auch dieselben Absicherungsmaßnahmen.

Ein weiteres wesentliches Merkmal ist die „Qualitätssicherung“. Dieses Merkmal ist relevant, da es auch Funktionen gibt, die von keinem der besonderen Merkmale betroffen sind. Diese müssen jedoch ebenfalls abgesichert werden, um die Qualität des Gesamtprodukts und der einzelnen Softwarebausteine sicherzustellen. Daher bildet dieses Merkmal, welches dem BM F zugeordnet werden kann, das Fundament für den Testbaukasten. Hier sind alle Absicherungsmaßnahmen enthalten, die nur zur Qualitätssicherung und nicht auf Basis von Regularien getroffen werden müssen.

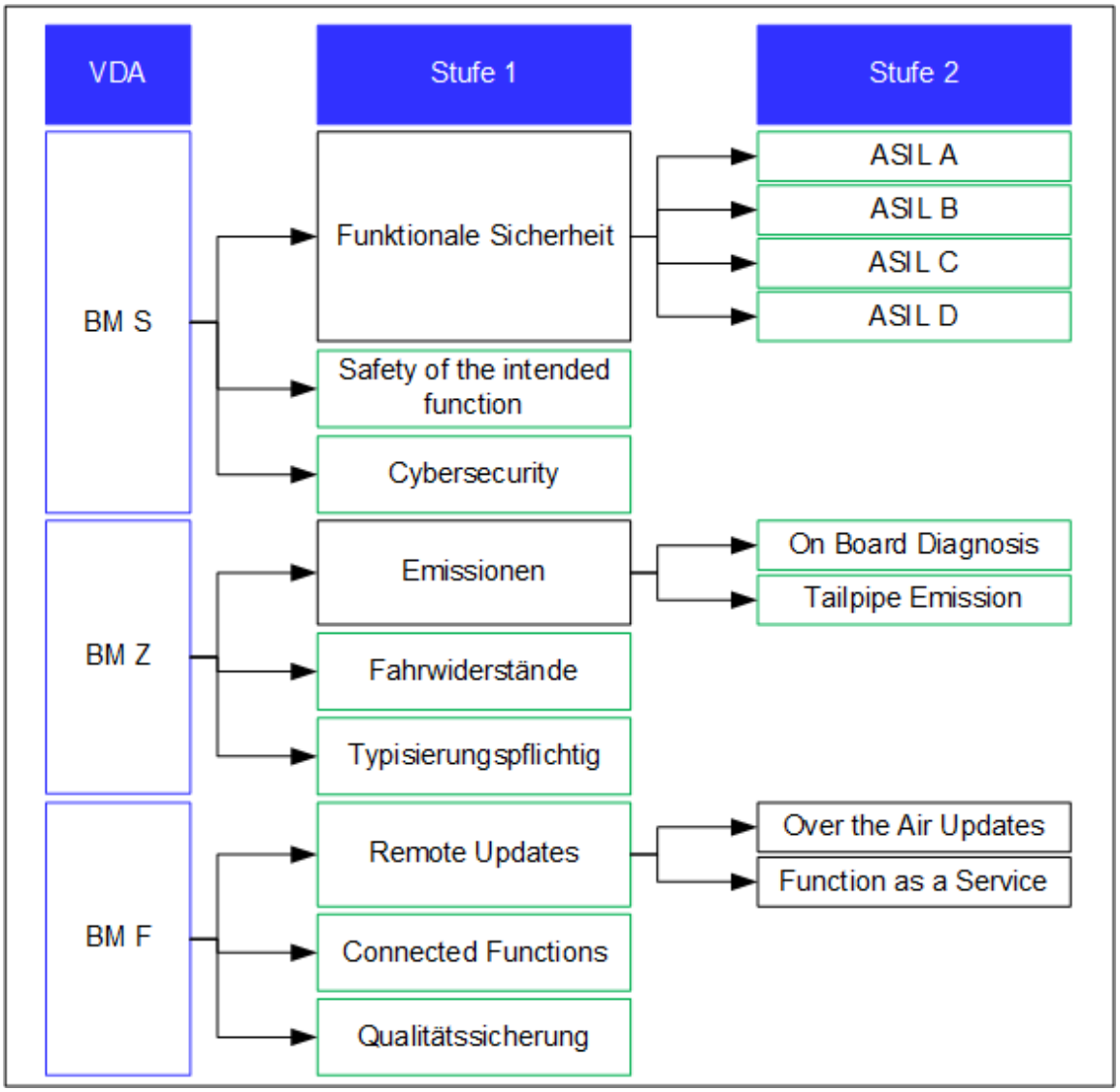

Abbildung 4.3: Besondere Merkmale für den Testbaukasten

Die vorgestellten BsM sind relevant für den MTB und damit für die Absicherung von Fahrwerkfunktionen. Jedes BsM wird durch unterschiedliche Testaktivitäten abgesichert. Mittels methodischer Anforderungserhebung wurden die Testaktivitäten für jedes Merkmal identifiziert und beschrieben. Die Methode zur Entwicklung des Baukastens wird im weiteren Verlauf dargestellt. Aufgrund des großen Umfanges wird die Methode anhand der folgenden Merkmale beschrieben:

1. Merkmal Qualitätssicherung
2. Besonderes Merkmal Funktionale Sicherheit ASIL A
3. Besonderes Merkmal Funktionale Sicherheit ASIL D
4. Besonderes Merkmal SOTIF
5. Besonderes Merkmal Cybersecurity
6. Besonderes Merkmal Remote Updates

4.3 Testaktivitäten in der Fahrwerkentwicklung

Jedes besondere Merkmal wird mit unterschiedlichen Testaktivitäten getestet. Das Ziel ist einerseits, die Tests mit der notwendigen Effektivität durchzuführen und zeitgleich Effizienz sicherzustellen und andererseits die Verwendung von einer Teststrategie für alle technischen Merkmale, die einem Testobjekt zugeschrieben werden können. Um das zu gewährleisten, müssen zunächst Anforderungen an Testmethoden und Testvorgehensweisen je BsM identifiziert werden. Anhand dieser werden Testaktivitäten entwickelt und den einzelnen Testebenen zugeordnet. Dadurch ergibt sich ein Gesamtbild, welches in einen modularen Testbaukasten überführt werden kann.

4.3.1 Methodische Bewertungsmethodik zur Entwicklung der Testmodule

Zunächst müssen normative und gesetzliche Anforderungen an den Baukasten und die Testaktivitäten erhoben werden. Elementar ist die Identifizierung von Richtlinien, die Vorgaben an die Testmethoden und Testaktivitäten zur Absicherung besonderer Merkmale machen. Die Erhebung der Anforderungen erfolgt vollumfänglich und methodisch. Dieser Schritt ist notwendig, um zu gewährleisten, dass der Ansatz des Baukastens anwendbar ist und die Ziele erfüllt werden. Grande (2011) bestätigt diese Notwendigkeit, da ohne Anforderungsmanagement die Ergebnisse der Arbeit unzureichend oder nicht anwendbar für Organisationen sind.

Weiter beschreibt Grande vier Haupttätigkeiten im Zusammenhang mit dem Anforderungsmanagement (Abbildung 4.4). Diese findet in dieser Arbeit in angepasster Form Anwendung. Die Tätigkeiten „finden und ermitteln“, „dokumentieren“ und „bewerten und evaluieren“, werden genutzt, um daraus den Testbaukasten abzuleiten. Dieser muss jedoch fortlaufend gepflegt werden, um Nutzbarkeit und Aktualität zu gewährleisten. Die Pflege ist hingegen kein Aspekt, der in dieser Arbeit vertieft betrachtet wird.

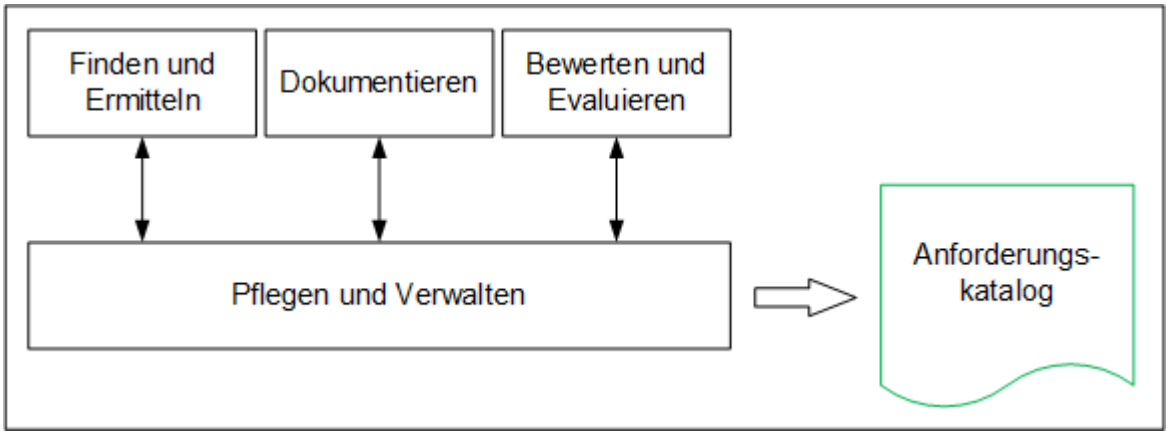

Abbildung 4.4: Methodisches Anforderungsmanagement (modifiziert nach Grande 2011)

Der erste Schritt ist das Finden und Ermitteln der Anforderungen. Anders als von Grande (2011) vorgesehen, beruhen diese nicht auf Ideen für ein Produkt, sondern auf Regularien. Daher war der erste Schritt zur Entwicklung des MTB das Identifizieren von Normen,

Vorschriften und Gesetzen. Dieser Schritt wurde für jedes BsM separat wiederholt. Sowohl etablierte als auch neue oder gar zukünftig gültige Regularien konnten identifiziert werden. Diese werden in Absatz 4.3.4 vorgestellt.

Zur Ableitung von Anforderungen wurden identifizierte Regularien im Rahmen dieser Arbeit genauestens analysiert und in einzelne Paragrafen und Abschnitte zerlegt. Dieses Vorgehen ist auch beschrieben in Jooß et al. (2022). Die Abschnitte wurden anschließend auf die Relevanz hinsichtlich SW-Testing untersucht. Daraus abgeleitete Anforderungen wurden dokumentiert, um sie im nächsten Schritt zu bewerten. Dabei wurde zwischen direkten und indirekten Anforderungen unterschieden. Direkte Anforderungen ergeben sich beispielsweise aus der ISO 26262 oder der ISO/IEC/IEEE 29119, in denen Methoden zur Absicherung vorgegeben sind. In den meisten Fällen handelt es sich jedoch um indirekte Anforderungen, welche mehr Interpretationsspielraum zulassen und daher für einen bestimmten Anwendungsfall genauer analysiert wurden. Dies beeinflusst den Schritt der Prüfung und Evaluierung der Anforderung und damit die Bestimmung der Testaktivitäten.

Bei der Prüfung wurden aus den Anforderungen Testaktivitäten abgeleitet. Im Rahmen des Baukastens stellt ein Testmodul eine Testaktivität zur Absicherung des Testobjektes dar. Einzelne Module werden in Abhängigkeit von den technischen Eigenschaften des Testobjektes flexibel mit anderen Modulen kombiniert. Im Sinne einer Testaktivität beschreibt ein Modul u. a. Testziele, Testmethoden und Testentwurfsverfahren, die einzuhalten sind. Dadurch entsteht ein zusammenhängender Testablauf in Abhängigkeit der besonderen Merkmale, die einem Testobjekt zugeordnet werden.

Die Bewertungsmethodik zur Entwicklung der Testmodule sieht drei Schritte vor (Abbildung 4.5). Im ersten Schritt gilt es verschiedene Regularien hermeneutisch zu untersuchen. Speziell die ISO/IEC/IEEE 29119-4:2015 macht Vorschläge, welche Testaktivitäten zu beachten sind und welche Testziele damit verfolgt werden können. Zusätzlich wurden weitere Regularien untersucht, um daraus Module abzuleiten oder bei Bedarf zu entwickeln. In weiteren Schritten wurden diese durch Experten aus dem Bereich Test hinsichtlich Machbarkeit und mit Experten der BsMs hinsichtlich Sinnhaftigkeit und Zuordnung bewertet. Bei einer erfolgreichen Bewertung wurden die Module in den MTB aufgenommen.

Einige Module wurden in Kategorien eingeteilt. Ausschlaggebend für die Kategorisierung sind vergleichbare Testziele. Da die Testmodule unterschiedliche Fehleridentifizierungsquoten, aber auch unterschiedliche Aufwände in der Testentwicklung und Testdurchführung haben, können hierdurch Effizienzvorteile genutzt werden. Dies bedeutet, dass pro Kategorie bevorzugt das Testmodul mit den geringsten Aufwänden genutzt werden soll, je nach BsM jedoch ein Modul mit größeren Identifizierungsquoten genutzt werden muss. Gründe dafür können beispielsweise Sicherheitsaspekte, rechtliche Anforderungen oder Qualitätsanforderungen sein. Die Bedeutung hiervon wird in Abschnitt 4.4 und 5.3 vertieft betrachtet. In Abschnitt 4.3.3 werden die Module im Detail vorgestellt.

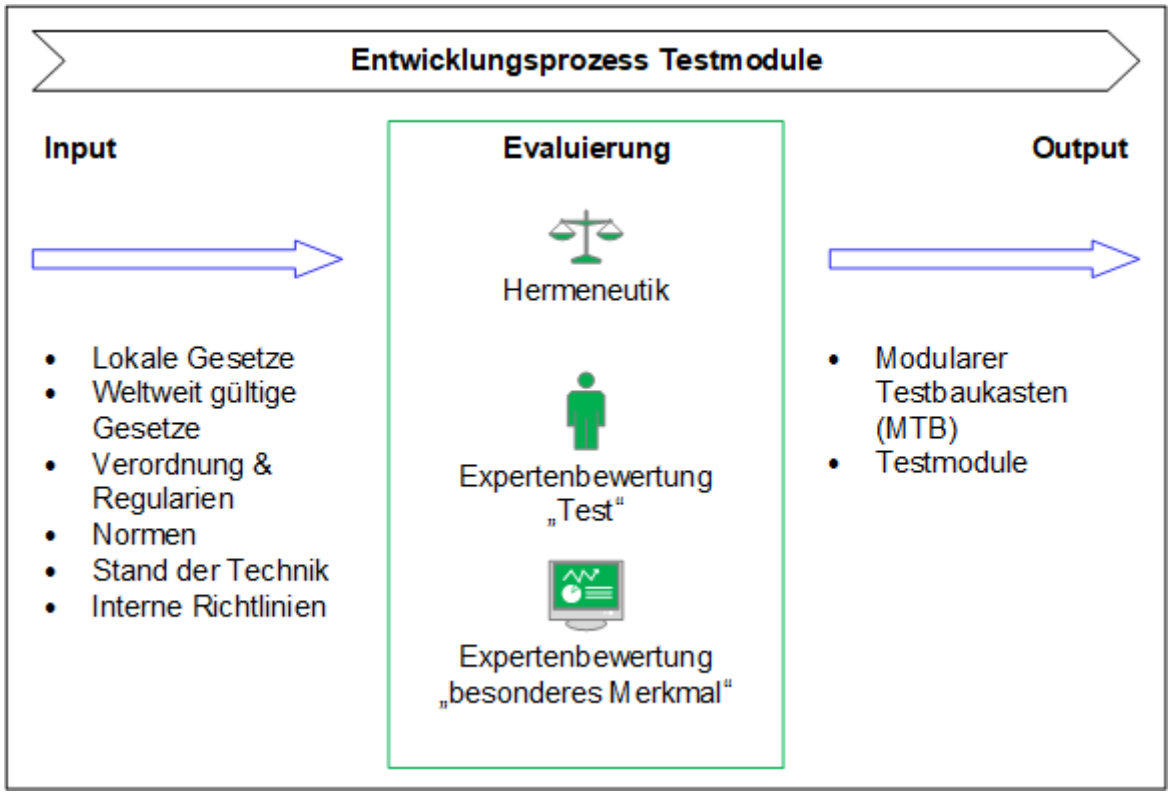

Abbildung 4.5: Entwicklungsprozess der Testmodule

4.3.2 Testebenen für die Baukasten-Entwicklung

Die Absicherung von Software erfolgt auf mehreren Integrationsebenen. Bereits in Abschnitt 3.1 werden diese beschrieben. In vielen Richtlinien, z. B. in ASPICE (VDA 2022), werden Ebenen individuell definiert. Auch für diese Arbeit ist es notwendig, Testebenen zu definieren, da auf den Ebenen, unabhängig von der Testinstanz, verschiedene Testaktivitäten durchzuführen sind. Die folgende Tabelle zeigt die Testebenen, die für diese Arbeit von Bedeutung sind und beschreibt darüber hinaus den Zusammenhang mit den Ebenen von ASPICE.

Tabelle 4.3: Übersicht der betrachteten Ebenen

Ebenen nach ASPICE	**Ebenen für diese Arbeit**
SWE.4: Software Unit Verification SWE.5: Software Integration und Integration Test SWE.6: Software Qualification Test	Softwarekomponente (SWC)
SYS.4: System Integration and Integration Test SYS.5: System Qualification Test	Softwaresystem (SWS)

Die Zusammenfassung der Ebenen erfolgt in erster Linie zur einfacheren Darstellung, da die Module sonst auf mehreren Ebenen aufgezeigt werden müssten. Da es sich in der

Realität jedoch um verschiedene Testobjekte handelt, die integriert werden, müssen die Module teilweise auf einer Ebene für verschiedene Testobjekte wiederholt werden. Für die gewählten Ebenen ist es jedoch von Bedeutung, dass ein individueller Baukasten entwickelt wurde, aus dem hervorgeht, welche Testmodule anzuwenden sind.

In der Ebene Softwarekomponente werden alle Testaktivitäten durchgeführt, die zur Verifikation der Komponenten und deren einzelnen Bestandteilen notwendig sind, was u. a. auch das Testen der Units und der Integration der Units zu einer Komponente umfasst. Ähnlich werden auf der Softwaresystem-Ebene alle Aktivitäten durchgeführt, die zur Absicherung des Softwaresystems notwendig sind. Darunter wird in Zusammenhang mit dieser Arbeit ein System im Sinne einer Wirkkette verstanden, welches aus mehreren SWCs besteht, um eine Funktionalität abbilden zu können.

4.3.3 Testmodule in der Fahrwerkentwicklung

Die beiden Testebenen werden unabhängig von der Testinstanz mit unterschiedlichen Testaktivitäten abgesichert. Zur vollständigen Absicherung ist es notwendig, die einzelnen Testaktivitäten auf verschiedenen Instanzen, bspw. MiL- oder SiL-Simulationen, zu wiederholen. Die Absicherung soll mit standardisierten Prozessen und Verfahren durchgeführt werden. Dazu wurden im Rahmen dieser Arbeit sogenannte Testmodule entwickelt, die Vorgaben hinsichtlich Testfallerstellung machen. Die Testfallerstellung muss für jede Testinstanz individuell geplant und durchgeführt werden.

Die einzelnen Module beinhalten alle für Tester und Testmanager relevanten Informationen zur Planung und Testfallerstellung. In Tabelle 4.4 ist ein Testmodul beispielhaft dargestellt. Wichtigste Information für die Testplanung ist das Testziel. Über dieses erfahren Tester, was mit diesem Test erreicht werden soll. Zeitgleich kann das Testziel Aufschluss zu alternativen Testmodulen mit gleichem Ziel geben. Diese werden zusätzlich explizit angegeben. Zudem können Tester sehen, welche Dokumente und Informationen vorliegen müssen, um mit dem Test zu beginnen. Diese sogenannten Testeingangskriterien haben als Gegenstück die Testendekriterien. Anhand dieser kann ein Tester erkennen, wann ein Test als ausreichend betrachtet und der Test beendet werden kann. Dies ist auch in Kapitel 5 für die Freigabemethode relevant. Ein weiterer für den Tester wesentlicher Inhalt des Testmoduls ist die Beschreibung des Testentwurfsverfahrens. Anhand derer weiß der Tester, wie er konkrete Testfälle aus einem Testmodul ableiten kann.

Tabelle 4.4: Testmodul "Grenzwertanalyse"

ID_004_2	**Grenzwertanalyse**
Testziel	Bestätigung, dass Eingangs- und Ausgangswerte an den Grenzen von Äquivalenzklassen zur korrekten Funktionalität führen, da es an den Grenzen der Wertebereiche vermehrt zu Fehlern im Testobjekt kommt.
Testeingangskriterien / Testbasis	Um den Test durchführen zu können, müssen folgende Elemente vorhanden sein: • Anforderungsbeschreibung. • Modell oder Code der SWC/SWF. • Testumgebung der Ebene. • Übersicht über Grenzwerte.
Testentwurf	Auf Basis des Testendekriteriums werden Testfälle entwickelt durch: 1. Identifizierung und Dokumentation der Grenzwerte für Eingangs- und Ausgangswerte 2. Weitere gültige Werte für alle anderen vom Testfall benötigte Eingabevariablen (willkürlich) festlegen 3. Beschreibung der durchzuführenden Testschritte 4. Bestimmung und Dokumentation der erwarteten Ergebnisse
Testmethode	Analyse der Grenzwerte
Testendekriterien	Pro Grenzwert werden folgende Werte getestet: Grenzwert; erster Wert außerhalb des Wertebereichs; erster Wert innerhalb des Wertebereichs
Alternativen	Äquivalenzklassentest
Normverweis	ISO 29119-4 beschreibt die Grenzwertanalyse und gibt Vorschläge zur Methodik, Testüberdeckung und Testfallerstellung. ISO 26262-6 fordert anforderungsbasierte Tests mittels einer Analyse von Grenzwerten.

Auf SWC-Ebene wurden im Rahmen dieser Arbeit 25 Testmodule entwickelt und in 17 Kategorien eingeteilt. Auf SWS-Ebene sind es 14 Module in 12 Kategorien. Hierfür wurden zunächst verschiedene Normen hermeneutisch untersucht, um Testmodule bzw. Testziele identifizieren zu können. Insbesondere die ISO/IEC/IEEE 29119-4:2015 und die Interpretationshilfe von Daigl und Glunz (2016) waren wichtige Impulsgeber für die Bestimmung der

relevanten Module. Zudem wurden die ISO/IEC 20246:2017 (Software and systems engineering - Work product reviews) und der IEEE Std 1028-2008 (IEEE Standard for Software Reviews and Audits) für die Module der Kategorie “Reviews“ analysiert. Das Vorgehen ist in Abschnitt 4.3.1 beschrieben. Dabei wurden Testverfahren mit gleichen oder ähnlichen Testzielen in Kategorien eingeteilt. Anschließend wurde für jedes Modul analysiert, welche Informationen zum Starten des Tests vorliegen müssen und wie diese genutzt werden, um einen konkreten Testfall abzuleiten. Zudem wurden eindeutige Testendekriterien bestimmt. Dadurch lässt sich der Interpretationsspielraum durch den Tester reduzieren und die Qualität gewährleisten.

Die erwähnten Kategorien werden genutzt, um Module mit gleichen oder sich ergänzenden Testzielen zusammenzufassen. Dies ist notwendig, da alle Testmodule einer Kategorie zur Erreichung eines vergleichbaren Testziels verwendet werden können und so bei Bedarf, z. B. durch zeitliche Engpässe, eine Alternative herangezogen werden kann. Diese Methode zur Testfallpriorisierung wird in Kapitel 5 ausführlich vorgestellt. Die nachfolgende Liste gibt eine Übersicht über die Testmodule, welche für diese Arbeit relevant sind. Hier wird auch über die Zuordnung der Testebene Aufschluss gegeben. Kurzfassungen der Module werden in Anhang A dargestellt.

- **Funktionale Tests**
    - Anforderungsbasierter Test (funktional), Ebene: SWC & SWS
    - Fault Injection Test, Ebene: SWC & SWS
    - Error Guessing, Ebene: SWC & SWS
    - Kategorie: Äquivalenzklassentest & Grenzwertanalyse, Ebene: SWC & SWS
        - Äquivalenzklassentest
        - Grenzwertanalyse
    - Zustandsbasierter Test, Ebene: SWC & SWS
    - Szenariotest, Ebene: SWS
- **Nicht-Funktionale Tests**
    - Vulnerability Scan, Ebene: SWC & SWS
    - Penetration Testing, Ebene: SWS
    - Fuzzy Testing, Ebene: SWC & SWS
    - Statistischer Test, Ebene: SWC & SWS
    - Kategorie: Ressourcennutzungstests (RUT), Ebene: SWC
        - Performance Test

        - Lasttest
        - Stresstest
    - Modelling Guideline Check, Ebene: SWC
    - Static Code Analysis, Ebene: SWC
    - Schnittstellentest, Ebene: SWC & SWS
- **Strukturbasierte Tests**
    - Anweisungstest, Ebene: SWC
    - Kategorie: Entscheidungsausgänge, Ebene: SWC
        - Zweigtest
        - Entscheidungstest
    - Kategorie: Bedingungstests, Ebene: SWC
        - Bedingungstest
        - Modifizierter Bedingungs-/Entscheidungstest (MC/DC)
    - Datenflusstest, Ebene: SWC
- **Reviews**
    - Kategorie: Reviews von Anforderungen (SWC & SWS), Code (SWC), Modell (SWC) oder Testspezifikation (SWC & SWS)
        - Technisches Review
        - Informelles Review
        - Walk-Through
        - Inspection

4.3.4 Zuordnung der Testmodule zu besonderen Merkmalen

In den Abschnitten 4.3.2 und 4.3.3 wurde eine Übersicht zu den für die Fahrwerkentwicklung relevanten Testebenen bzw. Testmodulen gegeben. Gemeinsam mit den besonderen Merkmalen bildet dies die Basis für den modularen Testbaukasten. In Abschnitt 4.3.1 wurde eine Methode zum Analysieren von Normen und Gesetzen vorgestellt, welche die Entwicklung der Testmodule unterstützt. Diese Methode findet auch Anwendung bei der Zuordnung der Testmodule zu besonderen Merkmalen, da diese ebenfalls auf Normen, Gesetzen und Erfahrung von Testern und Entwicklern basieren.

Im ersten Schritt der Methode müssen Anforderungen identifiziert werden. Insgesamt wurden zum aktuellen Zeitpunkt 32 verschiedene Regularien identifiziert. Dies ist jedoch eine veränderliche Anzahl, da die Regularien stets an neue Rahmenbedingungen angepasst werden und so neue hinzukommen oder alte ersetzt werden. Daher müssen permanent die Anforderungsdokumente überwacht werden, um den Testbaukasten auf aktuellem Stand zu halten. In Tabelle 4.5 werden internationale Regularien aufgelistet, die zum aktuellen Zeitpunkt Gültigkeit besitzen oder in naher Zukunft in Kraft treten und für die Einschränkung in Abschnitt 4.2 notwendig sind. (In Anhang B werden alle relevanten Regularien für die Fahrwerkentwicklung aufgezeigt.) Darüber hinaus wird in Tabelle 4.5 ein Zusammenhang zwischen BsM und den Regularien aufgezeigt.

Tabelle 4.5: Übersicht relevanter Regularien für den modularen Testbaukasten (Auszug)

Norm-Nr.	Titel / Inhalt	Merkmal Qualitätssicherung	BsM FuSi ASIL A	BsM FuSi ASIL D	BsM SOTIF	BsM Cybersecurity	BsM Remote Updates
Automotive SPICE	**S**oftware **P**rocess **I**mprovement and **C**apability D**e**termination	Allgemeingültig zur Prozessentwicklung					
AUTOSAR CP R20-11	**AUT**omotive **O**pen **S**ystem **AR**chitecture – Classic Plattform	Allgemeingültige Anforderungen an SW-Entwicklung					
GB 201-5	General Technical Requirements for Software Updates of Vehicles						x
IEEE Std 1028-2008	Software Reviews and Audits	Reviews werden für alle BsM gefordert					
ISO 26262-6:2018	Road vehicles - Functional safety	x	x	x			
ISO/DIS 24089:2022	Road vehicles - Software update engineering						x
ISO/IEC 20246:2017	Software and systems engineering - WP reviews	Reviews werden für alle BsM gefordert					
ISO/PAS 21448:2019	Safety of the intended functionality				x		

Norm-Nr.	Titel / Inhalt	Merkmal Qualitätssicherung	BsM FuSi ASIL A	BsM FuSi ASIL D	BsM SOTIF	BsM Cybersecurity	BsM Remote Updates
ISO/SAE 21434:2021	Road vehicles - Cybersecurity engineering					x	
NHTSA	Cybersecurity Best Practices for Modern Vehicles					x	x
Normreihe ISO/IEC 15408-x:200x; ISO/IEC 15408-3:2008	Information technology - Security techniques - Evaluation criteria for IT security					x	
Normreihe ISO/IEC/IEEE 29119-x:201x	Software Testing	Allgemeingültig für SW-Test					
UN Regelung Nr. 13	Typgenehmigung von Fahrzeugen hinsichtlich der Bremsen				x		
UN Regelung Nr. 13 H	Genehmigung von PKW hinsichtlich der Bremsen				x		
UN Regelung Nr. 79	Genehmigung der Fahrzeuge hinsichtlich der Lenkanlage				x		
UN Regelung Nr. 140	Genehmigung von PKW hinsichtlich der elektronischen Fahrdynamik-Regelsysteme				x		
UN Regelung Nr. 141	Genehmigung von KFZ hinsichtlich Reifendruckkontrollsysteme				x		
UN Regelung Nr. 155	Genehmigung von Fahrzeugen; CS und CS-Managementsystems					x	x
UN Regelung Nr. 156	Genehmigung von KFZ; SW-Aktualisierung und SW-Aktualisierungs-Managementsystems						x

AUTOSAR - Classic Plattform

Dieser Standard, den Automobilhersteller und Zulieferer zur “AUTomotive Open System ARchitektur (AUTOSAR)” entwickelt haben, verfolgt das Ziel, Architekturen von Software und Steuergeräten zu standardisieren. Verschiedene Anforderungen an einzelne Softwareelemente und Prozesse sind genannt. Diese gilt es einzuhalten. Im Release AUTOSAR CP R20-11 werden zudem Anforderungen an Remote Updates gestellt. Diese Anforderungen beeinflussen die Absicherungsprozesse der Software, insbesondere im Hinblick auf das entsprechende BsM.

GB 201-5 - General Technical Requirements for Software Updates of Vehicles

Dieser nationale chinesische Standard beschreibt Cybersecurity Anforderungen, die bei Updates von Software berücksichtig werden müssen, um Fahrzeuge im chinesischen Markt zuzulassen. Diese Anforderungen betreffen ebenfalls den Test und müssen bei der Entwicklung eines Baukastens für die BsM Remote Updates und Cybersecurity beachtet werden.

IEEE Std 1028-2008 - Software Reviews and Audits

In der IEEE 1028-2008 werden fünf verschiede Arten von Reviews und Audits von Software Artefakten definiert. Für jede Art werden Vorgehen, Verantwortlichkeiten, notwendige Ein- und Ausgangsbedingungen sowie Output vorgestellt. Jedoch wird explizit nicht definiert, wann diese Reviews angewandt werden und wie der Prozess zur Bestimmung der Notwendigkeit ist. Diese Informationen werden jedoch in weiteren Normen, beispielsweise der Normreihe ISO 26262-x:201x oder der ISO/IEC 20246:2017 beschrieben. Für die Entwicklung einzelner Testmodule ist der IEEE Std 1028-2008 jedoch von Bedeutung.

Normreihe ISO 26262-x:201x - Road vehicles - Functional safety

Diese Norm definiert den Standard zur funktionalen Sicherheit im Automobilbereich. Als Grundstruktur werden in der ISO 26262-3:2018 vier Risikoklassen, sogenannte Automotive Safety Integrity Level (ASIL), definiert. ASIL A bezeichnet die Klasse mit dem niedrigsten Gefahrenpotenzial, ASIL D die mit dem Höchsten. Das Potenzial einer Funktion wird mittels einer Risikobewertung ermittelt. Die Bewertung erfolgt anhand der Auswirkungen (engl. Severity), der Eintrittswahrscheinlichkeit (engl. Exposure) und der Kontrollierbarkeit durch den Fahrzeugführer (engl. Controlability). Diese drei Kriterien werden einem definierten aufsteigenden Wert (S0-S3, E0-E4 und C0-C3) versehen und miteinander verrechnet, um eine entsprechendes ASIL zu erhalten.

Für alle Level definiert die Normreihe ISO 26262-x:201x in 12 Bänden Entwicklungsmethoden, um diese Gefahren möglichst gering zu halten. Band 6 (Product development at the software level) ist im Rahmen dieser Arbeit besonders relevant. Dieser Band macht Angaben zum Entwicklungsprozess von Softwareelementen. Betrachtungsgegenstand dieses Bandes sind alle Ebenen des V-Modells von der Anforderungserhebung über die Entwicklung und Implementierung bis zum Testabschluss. Die Testmodule auf SWC-Ebene werden in den Kapiteln 9 und 10 definiert, für die SWS-Ebene in den Kapiteln 10 und 11.

ISO/DIS 24089:2022 - Road vehicles - Software update engineering

Die ISO/DIS 24089:2022 befindet sich im Entwurfsstatus und wird internationale Gültigkeit besitzen. In der Norm werden organisatorische, prozessuale sowie technische Anforderungen zur Beherrschung von SW Updates in Fahrzeugen beschrieben. Die Anforderungen sollen dazu dienen, die Entwicklung und Updates von SW so zu gestalten, dass auch unabhängige Parteien diese bewerten können, um eine entsprechende Qualität zu gewährleisten. Dadurch ergeben sich auch Anforderungen, die in dieser Arbeit Berücksichtigung finden. (Burkacky et al. 2020)

ISO/IEC 20246:2017 - Software and systems engineering - Work product reviews

Wie im Titel zu erkennen, beschäftigt sich diese Norm mit dem Review verschiedener „work products“. Darunter wird jedes Erzeugnis verstanden, dass das Ergebnis eines Prozesses während der Entwicklung darstellt. In der Norm werden Reviewmethoden, -prozesse und -aktivitäten vorgestellt. Da dieser Umfang ebenfalls wesentlich für eine Methode zum Testen und Freigeben ist, wird diese Norm in Kombination mit dem IEEE Std 1028-2008 vollumfänglich in dieser Arbeit bedacht.

Normreihe ISO/IEC 15408-x:200x - Information technology - Security techniques - Evaluation criteria for IT security

In der Normreihe ISO/IEC 15408-x:200x werden Vorgaben zur Informationssicherheit von IT-Produkten definiert. In Band 1 wird explizit darauf hingewiesen, dass hierunter kein IT-Produkt im klassischen Sinne, sondern ebenso Software oder Firmware verstanden werden kann. Damit ist auch die Fahrzeugsoftware im Scope der Norm.

Für diese Arbeit ist vor allem Band 3 relevant. In diesem Band werden Anforderungen an die Vertrauenswürdigkeit dargestellt, welche die IT-Produkte einhalten müssen. Zusätzlich wird auf die Notwendigkeit eines ausgereiften und ausreichenden Testprozesses verwiesen und oberflächlich einige Testverfahren vorgeschlagen. Für diese Arbeit bedeutet dies, dass neben der Berücksichtigung der Normreihe ISO/IEC 15408-x:200x in erster Linie die ISO/SAE 21434:2021 zum Tragen kommt und u. a. einen solchen Testprozess beschreibt.

Normreihe ISO/IEC/IEEE 29119-x:201x - Software and systems engineering - Software Testing

Die SW-Testing-Norm dient als wesentlicher Input für diese Arbeit. In den fünf Teilen werden Vorgaben für das Testen von Software gemacht. Teil 1 ist als einziger Teil der Reihe nicht normativ, gibt jedoch wichtigen Input für die Arbeit mit der Norm. In diesem Teil werden Begriffe und allgemeine Konzepte für das Testen von SW definiert. Der zweite Teil gibt einen Überblick über Testprozesse inklusive Rollen und Aufgaben (Abschnitt 3.3.1). Im dritten Teil werden Aufbau und Zweck von Testdokumenten (u. a. Teststrategien, Testkonzepte oder Testberichte) beschrieben. Teil 4 beschäftigt sich mit Testverfahren. Auf dieser Basis wurden

auch viele der in Abschnitt 4.3.3 vorgestellten Testmodule entwickelt. Im fünften Teil wird ausschließlich das „keyword driven Testing“ betrachtet. Dieser Teil ist für diese Arbeit nicht relevant. (Normreihe ISO/IEC/IEEE 29119-x:201x; Daigl und Glunz 2016)

ISO/PAS 21448:2019 - Road vehicles - Safety of the intended functionality

In der Automobilbranche wird die Gebrauchssicherheit (englisch: Safety of the intended Functionality (SOTIF)) durch die ISO/PAS 21448:2019 geregelt. In der Norm werden Methoden zur Anforderungsstellung, Sicherheitslückenidentifizierung sowie zur Reduzierung der Risiken vorgestellt. Ein wichtiges Augenmerk liegt auch auf der Verifikation und Validierung der Systeme. D. h. dass neben den Anforderungen an eine entsprechende Teststrategie (u. a. Beschreibung der Absicherungsmethoden, Testpläne) auch Anforderungen an die Methoden zur Verifikation von Sensoren, Algorithmen (Software), Aktuatoren und integrierten Systemen gestellt. Für diese Arbeit sind hauptsächlich die Abschnitte 9.2 sowie 10.3 der Norm von Bedeutung.

ISO/SAE 21434:2021 - Road vehicles - Cybersecurity engineering

In dieser Norm werden Cybersecurity Maßnahmen für die Fahrzeugentwicklung definiert. Zu diesen Maßnahmen zählen das Definieren von unternehmensspezifischen CS Richtlinien und Prozessen, das langfristige Managen von CS Risiken im Fahrzeugumfeld und die Etablierung einer sogenannten Cybersecurity Kultur im Unternehmen. Auch diese Norm betrachtet das gesamte V-Modell der Entwicklung, wozu wesentliche Aspekte der Teststrategie und Testprozesse zählen. In den Kapiteln 5 und 10 der Norm wird der Einsatz von Testtools, Testmethoden und Testaktivitäten vorgeschlagen. Dieser wichtige Aspekt wird in dieser Arbeit nicht vernachlässigt.

NHTSA - Cybersecurity Best Practices for Modern Vehicles

Die Best Practices des U.S. Departments for Transportation sind rechtlich nicht bindend. Allerdings wird Fahrzeugherstellern und Entwicklern empfohlen, die aufgelisteten ganzheitlichen Anforderungen an Entwicklungsobjekte einzuhalten. Die National Highway Traffic Safety Administration (NHTSA) macht verschiedene Vorschläge, wie das Risiko durch Cybersecurity Attacken im Fahrzeug minimiert werden kann. Sowohl Anforderungen hinsichtlich Umsetzung von Abwehrmechanismen als auch Anforderungen an das Testing der Software und Hardware im Fahrzeug sind in den Best Practices enthalten. Da diese Aspekte relevant für die Absicherung sind, werden diese Best Practices in der Arbeit berücksichtigt, obwohl sie keinen normativen Charakter haben.

UN-Regelungen zur Typgenehmigung

Die UN-Regelungen zur Typgenehmigung (Nr. 0 bis Nr. 163) sind vor allem für den europäischen Markt konzipiert, finden jedoch auch in vielen anderen Märkten Anwendung. Sie geben ein einheitliches System zur Zulassung von Fahrzeugen, -Bauteilen und –Systemen vor. Die Regeln machen Vorgaben, was zur Zulassung eingehalten werden muss und wie ein

Zulassungsprüfverfahren auszusehen hat. Darüber hinaus werden Vorschriften zum Systemdesign und zur Entwicklungsmethodik gemacht, die das Testumfeld beeinflussen. Daher sind einige der Regularien für die Fahrwerkentwicklung und somit für diese Arbeit relevant.

UN Regelung Nr. 155 - CS und CS-Managementsystems &
UN Regelung Nr. 156 - SW-Aktualisierung und SW-aktualisierungs-Managementsystems

Besonders hervorzuheben sind für diese Arbeit die Regelungen Nr. 155 und Nr. 156, welche sich auf Cybersecurity und SW-Updates konzentrieren. Die rechtlich bindenden Regelungen sehen vor, dass Automobilhersteller Systeme einrichten zum Managen von CS und SW-Updates. Diese Systeme müssen unabhängig auditiert werden und beeinflussen damit die Entwicklungsabläufe. Zusätzlich werden auch technische Anforderungen gestellt, welche umgesetzt werden müssen. Damit ergeben sich auch Anforderungen an den Testprozess von Software, welche in dieser Arbeit einbezogen werden.

Die vorgestellten Normen dienen als Grundlage für die Zuordnung der Testmodule zu den genannten besonderen Merkmalen. Dabei wurden diese Normen analysiert und für diese Arbeit hinsichtlich der Testanforderungen interpretiert. Durch die Interpretation der Normen ergeben sich Testmodule und damit Testabläufe für verschiedene besondere Merkmale, welche im Folgenden für die im Abschnitt 4.2 ausgewählten sechs BsM erläutert werden.

Merkmal Qualitätssicherung (QS)

Um die notwendige Effizienz zu erzielen, müssen die Testaktivitäten den entsprechenden BsM zugeordnet werden. Einigen Softwareelementen können jedoch keine besonderen Merkmale zugeordnet werden. Da für diese Elemente ebenfalls Qualitätsstandards eingehalten werden müssen, ist es notwendig, ein Mindestmaß an Softwaretestumfängen zu definieren. Die beiden Tabellen (Tabelle 4.6 und Tabelle 4.7) geben Auskunft über die relevanten Module auf SWC bzw. SWS Ebene. Überwiegend werden Testverfahren angewandt, die einen geringen Aufwand zur Testerstellung haben, um möglichst effizient die Qualität zu gewährleisten. Einige Tests werden nur empfohlen, da es keine normativen Anforderungen gibt, die Qualität damit jedoch gesteigert werden kann. Im Vergleich zwischen den Ebenen fällt auf, dass Tests im Hinblick auf Struktur und Ressourcen möglichst früh getestet werden sollen. Deshalb sind diese auf der SWC Ebene verortet.

Tabelle 4.6: Testmodule auf SWC-Ebene Qualitätssicherung

Testmodul		**Normative Forderung**
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO/IEC/IEEE 29119-4:2015 Abs. 5.2
Technisches	Verpflichtend	IEEE Std 1028-2008 Abs. 5

Testmodul		Normative Forderung
Anforderungsreview		ISO/IEC 20246:2017Annex F.2
Anforderungsreview Informal	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Informal	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Äquivalenzklassentest	Verpflichtend	ISO/IEC/IEEE 29119-4:2015 Abs. 5.2.1
Technisches Modellreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Modellreview Informal	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Technisches Codereview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Codereview Informal	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Performance Test	Verpflichtend	QS ohne normative Anforderung
Anweisungstest	Verpflichtend	ISO/IEC/IEEE 29119-4:2015 Abs. 5.3.1
Bedingungstest	Verpflichtend	ISO/IEC/IEEE 29119-4:2015 Abs. 5.3.4
Modelling Guideline Check	Verpflichtend	QS ohne normative Anforderung
Static Code Analysis	Verpflichtend	QS ohne normative Anforderung
Schnittstellentest	Verpflichtend	ISO 26262-6:2018 Annex E

Tabelle 4.7: Testmodule auf SWS-Ebene Qualitätssicherung

Testmodul		Normative Forderung
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO/IEC/IEEE 29119-4:2015 Abs. 5.2
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Informal	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Informal	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Äquivalenzklassentest	Verpflichtend	ISO/IEC/IEEE 29119-4:2015 Abs. 5.2.1
Schnittstellentest	Verpflichtend	ISO 26262-6:2018 Annex E

Die Qualitätssicherung bildet die Grundlage des Testens. Daher basiert Zuordnung der Testmodule auf der Softwaretestnorm. Der notwendige Input zur Beschreibung der Module erfolgt auf Basis der ISO/IEC/IEEE 29119-4:2015. Zudem werden in der ISO/IEC 20246:2017 und dem IEEE Std 1028-2008 die Reviews beschrieben, die notwendig sind, um die Qualität zu sichern. Da hier weniger Anforderungen an die Sicherheit gemacht werden als bei anderen besonderen Merkmalen, sind hier weniger formale Methoden ausreichend. Auch die ISO 26262-6:2018 gibt Vorgaben an die Absicherung der Qualität v. a. an die Absicherung von Schnittstellen zwischen FuSi und nicht-Fusi Elementen. Darüber hinaus wurden auch Testaktivitäten auf der Erfahrung von Testern und Entwicklern entwickelt und aufgenommen.

Besonderes Merkmal Funktionale Sicherheit ASIL A

Die Anforderungen hinsichtlich funktionaler Sicherheit werden in der Fahrzeugentwicklung bereits seit mehreren Jahren bedacht und durch verschiedene Prozesse und Methoden unterstützt. Das ist wesentlich, um die notwendige Sicherheit von elektrischen und elektronischen Fahrzeugfunktionen gewährleisten zu können. Da die in der ISO 26262-6:2018 geforderten Testaktivitäten abhängig vom ASIL sind, werden hier ASIL A und ASIL D gezeigt, um die Unterschiede zu verdeutlichen. Tabelle 4.8 zeigt die Testmodule auf SWC- Tabelle 4.9 auf SWS-Ebene für ASIL A.

Tabelle 4.8: Testmodule auf SWC-Ebene ASIL A

Testmodul		**Normative Forderung**
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO 26262-6:2018 Table 7 1j + Table 10 1a
Fault Injection Test	Empfohlen	ISO 26262-6:2018 Table 7 1l + Tabel 10 1c
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Walk-through	Verpflichtend	ISO 26262-8:2018Table 2 1a ISO/IEC 20246:2017 Annex F.2
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Walk-through	Verpflichtend	ISO 26262-6:2018 Table 7 1a ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Empfohlen	ISO 26262-6:2018 Table 8 1c
Technisches Modellreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Modellreview Walk-through	Verpflichtend	ISO 26262-6:2018 Table 7 1a ISO/IEC 20246:2017 Annex F.2

Testmodul		Normative Forderung
Technisches Codereview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Codereview Walk-through	Verpflichtend	ISO 26262-6:2018 Table 7 1a ISO/IEC 20246:2017 Annex F.2
Lasttest	Verpflichtend	ISO 26262-6:2018 Table 7 1m
Anweisungstest	Verpflichtend	ISO 26262-6:2018 Table 9 1a
Modelling Guideline Check	Verpflichtend	ISO 26262-6:2018 Table 1
Static Code Analysis	Verpflichtend	ISO 26262-6:2018 Table 1 + Table 7 1h
Schnittstellentest	Verpflichtend	ISO 26262-6:2018 Table 7 1k

Tabelle 4.9: Testmodule auf SWS-Ebene ASIL A

Testmodul		Normative Forderung
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO 26262-6:2018 Table 14 1a
Fault Injection Test	Empfohlen	ISO 26262-6:2018 Table 14 1b
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Walk-through	Verpflichtend	ISO 26262-8:2018 Table 2 1a ISO/IEC 20246:2017 Annex F.2
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Walk-through	Verpflichtend	ISO 26262-6:2018 Table 7 1a ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Verpflichtend	ISO 26262-6:2018 Table 15 1c
Schnittstellentest	Verpflichtend	ISO 26262-6:2018 Table 10 1b

Die Normreihe ISO 26262-x:201x beschreibt den Umgang mit funktionaler Sicherheit für die Fahrzeugentwicklung. Band 6 hat für diese Arbeit die größte Bedeutung, da dieser Teil die Softwareentwicklung behandelt. Die einzelnen Kapitel behandeln verschiedene Ebene. Da sich diese nicht direkt auf die in dieser Arbeit verwendeten Ebene mappen lässt, ist eine Interpretation notwendig. Dadurch kommt es auch zu Überlappungen. So beschäftigt sich Kapitel 9 mit der Software Unit Verifikation, Kapitel 10 mit der Software Integration und Verifikation. In Kapitel 11 geht es um das Testen von Embedded Software. Insbesondere Kapitel 10 hat damit auf beide Ebenen Auswirkungen, da die SWC-Ebene die Integration von Modulen und die SWS-Ebene die Integration zu Gesamtsoftwareverbünden beinhaltet.

Zusätzlich zur ISO 26262 greifen auch hier die beiden Normen hinsichtlich Reviews. Dabei wird nicht nur von der FuSi-Norm, sondern auch von der ISO/IEC 20246:2017 vorgegeben, dass mit größerem Sicherheitsrisiko ein Verfahren mit höherem Standardisierungsgrad genutzt werden soll. Damit ist diese Vorgabe für alle besonderen Merkmale mit Sicherheitsrelevanz von Bedeutung.

Besonderes Merkmal Funktionale Sicherheit ASIL D

ASIL D basiert auf denselben Normen und Normbänden. In den einzelnen Tabellen, die als Referenz herangezogen werden, werden Empfehlungen abhängig vom ASIL gemacht. Der Unterschied zwischen den Leveln wird in Tabelle 4.10 für die SWC- und Tabelle 4.11 für die SWS-Ebene dargestellt.

Tabelle 4.10: Testmodule auf SWC-Ebene ASIL D

Testmodul		**Normative Forderung**
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO 26262-6:2018 Table 7 1j + Table 10 1a
Fault Injection Test	Verpflichtend	ISO 26262-6:2018 Table 7 1l + Table 10 1c
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Inspection	Verpflichtend	ISO 26262-8:2018 Table 2 1b ISO/IEC 20246:2017 Annex F.2
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Inspection	Verpflichtend	ISO 26262-6:2018 Table 7 1c ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Verpflichtend	ISO 26262-6:2018 Table 8 1c
Technisches Modellreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Modellreview Inspection	Verpflichtend	ISO 26262-6:2018 Table 7 1c ISO/IEC 20246:2017 Annex F.2
Technisches Codereview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Codereview Inspection	Verpflichtend	ISO 26262-6:2018 Table 7 1c ISO/IEC 20246:2017 Annex F.2
Lasttest	Verpflichtend	ISO 26262-6:2018 Table 7 1m
Zweigtest	Verpflichtend	ISO 26262-6:2018 Table 7 1f
Datenflusstest	Verpflichtend	ISO 26262-6:2018 Table 7 1g
MC/DC	Verpflichtend	ISO 26262-6:2018 Table 9 1c

Testmodul		Normative Forderung
Modelling Guideline Check	Verpflichtend	ISO 26262-6:2018 Table 1
Static Code Analysis	Verpflichtend	ISO 26262-6:2018 Table 1 + Table 7 1h
Schnittstellentest	Verpflichtend	ISO 26262-6:2018 Table 7 1k

Tabelle 4.11: Testmodule auf SWS-Ebene ASIL D

Testmodul		Normative Forderung
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO 26262-6:2018 Table 14 1a
Fault Injection Test	Verpflichtend	ISO 26262-6:2018 Table 14 1b
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Inspection	Verpflichtend	ISO 26262-8:2018 Table 2 1b ISO/IEC 20246:2017 Annex F.2
Error Guessing	Verpflichtend	ISO 26262-6:2018 Table 15 1d
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Inspection	Verpflichtend	ISO 26262-6:2018 Table 7 1c ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Verpflichtend	ISO 26262-6:2018 Table 15 1c
Szenariotest	Verpflichtend	ISO 26262-6:2018 Table 15 1f
Schnittstellentest	Verpflichtend	ISO 26262-6:2018 Table 10 1b

Besonderes Merkmal Safety of the intended Function

Unter SOTIF wird die Sicherheit bei einem vorhersehbaren Gebrauch eines Fahrzeuges für den Nutzer verstanden. Das Gesamtfahrzeug kann diese Sicherheit nur bieten, wenn sie für die Einzelteile und deren Integration zu einem Gesamtprodukt gewährleistet ist. Dadurch lassen sich Anforderungen an die Absicherung von SW ableiten. Die resultierenden Testmodule und deren Anforderungsbasis sind in Tabelle 4.12 und Tabelle 4.13 genannt.

Tabelle 4.12: Testmodule auf SWC Ebene SOTIF

Testmodul		Normative Forderung
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO/PAS 21448:2019 Table 4 A + Table 6 B
Fault Injection Test	Verpflichtend	ISO/PAS 21448:2019 Table 4 G
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2

Testmodul		Normative Forderung
Anforderungsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Error Guessing	Verpflichtend	ISO/PAS 21448:2019 Table 4 E + I
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Verpflichtend	ISO/PAS 21448:2019 Table 4 D
Zustandsbasierter Test	Verpflichtend	ISO/PAS 21448:2019 Table 4 H
Technisches Modellreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Modellreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Technisches Codereview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Codereview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Entscheidungstest	Verpflichtend	ISO/PAS 21448:2019 Table 4 H + L
Schnittstellentest	Verpflichtend	ISO/PAS 21448:2019 Table 4 B

Tabelle 4.13: Testmodule auf SWS Ebene SOTIF

Testmodul		Normative Forderung
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO/PAS 21448:2019 Table 4 A + Table 6 B UN Regelung Nr. 13 + UN Regelung Nr. 13 H Anhang 21, Anlage 2, Abs. 1.2 UN Regelung Nr. 79 Anhang 6, Abs. 3.2 UN Regelung Nr. 140 Abs. 6 UN Regelung Nr. 141 Abs. 5
Fault Injection Test	Verpflichtend	ISO/PAS 21448:2019 Table 4 G UN Regelung Nr. 79 Anhang 6, Abs.4.1
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Error Guessing	Verpflichtend	ISO/PAS 21448:2019 Table 4 E + I

Testmodul		Normative Forderung
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Verpflichtend	ISO/PAS 21448:2019 Table 4 D
Zustandsbasierter Test	Verpflichtend	ISO/PAS 21448:2019 Table 4 H
Szenariotest	Verpflichtend	ISO/PAS 21448:2019 Annex E UN Regelung Nr. 13 + UN Regelung Nr. 13 H Abs. 5 + Anhang 21, Abs. 2 UN Regelung Nr. 79 Anhang 6, Abs. 4.1 UN Regelung Nr. 140 Abs. 9.9 in Verbindung mit Anhang 3
Schnittstellentest	Verpflichtend	ISO/PAS 21448:2019 Table 4 B

Die meisten Anforderungen an die Absicherung des BsM SOTIF lassen sich aus der ISO/PAS 21448:2019 ableiten. Wesentlicher Inhalt der Norm sind Szenarien, während derer die Sicherheit gewährleistet sein muss. Diese Szenarien sollten ebenso die Testaktivitäten beeinflussen wie die in Kapitel 9 geforderten Absicherungsmechanismen. Hier werden verschiedene Methoden unabhängig von der Testebene angegeben, die Berücksichtigung finden. Auch Kapitel 10 stellt Anforderungen an den Test und die Absicherung des Gesamtsystems.

Für die Absicherung der SWS-Ebene müssen zudem UN-Regelungen in Betracht gezogen werden. Diese Regelungen stellen Anforderungen an Gesamtsysteme, bspw. Bremsanlage (Nr. 13 & 13H), Lenkanlage (Nr. 79) Fahrdynamikregelsysteme (Nr. 140) und Reifendruckkontrollsysteme (Nr. 141). Aufgrund der Systembetrachtung beeinflusst dies die SWS-Ebene. Anforderungen, die von den Regelungen an die Systeme gestellt werden, gilt es abzusichern. Alternativ kann die Gebrauchssicherheit an das System mittels verschiedener Szenarien nachgewiesen werden.

Besonderes Merkmal Cybersecurity

Das BsM Cybersecurity ist relevant, da durch eine zunehmende Vernetzung der Fahrzeuge mit der Umwelt auch Angriffe durch Hacker wahrscheinlicher werden. Auch Funktionen, die dem Fahrwerk zugerechnet sind, sind teilweise über eine digitale Schnittstelle mit einem Back-End verbunden und haben daher hohe Anforderungen an die IT-Sicherheit. Neben Maßnahmen auf Gesamtverbundebene werden in verschiedenen Normen auch Maßnahmen auf SWC- oder SWS-Ebene gefordert. Daraus lassen sich die in Tabelle 4.14 für die SWC- und Tabelle 4.15 für die SWS-Ebene vorgestellten Testmodule ableiten.

Tabelle 4.14: Testmodule auf SWC Ebene CS

Testmodul		**Normative Forderung**
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO/IEC 15408-3:2008 Abs. 5.2.4 + Tabelle 8 ISO/SAE 21434:2021 [RQ-05-11] + [RQ10-10] NHTSA ganzheitlich UN Regelung Nr. 155 Abs. 7.2.2.2.
Technisches Anforderungsreview	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Anforderungsreview Walk-Through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Error Guessing	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Technisches Testspezifikationsreview	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Testspezifikationsreview Walk-Through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Grenzwertanalyse	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Fuzzy Testing	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Statistischer Test	Empfohlen	UN Regelung Nr. 155 Tabelle A1 – 26.1 bis 26.3
Technisches Modellreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Modellreview Walk-Through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Technisches Codereview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Codereview Walk-Through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36

Testmodul		Normative Forderung
Performance Test	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Datenflusstest	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10] NHTSA T.9
MC/DC	Verpflichtend	ISO/IEC 15408-3:2008 Tabelle 8 ISO/SAE 21434:2021 [RQ-10-11]
Modelling Guideline Check	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-08] + [RQ-10-10]
Static Code Analysis	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-08] + [RQ-10-10]
Schnittstellentest	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Vulnerability Scan	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10] NHTSA G.15 UN Regelung Nr. 155 Tabelle A1 - 29.1. + 29.2

Tabelle 4.15: Testmodule auf SWS Ebene CS

Testmodul		Normative Forderung
Anforderungsbasierter Test (funktional)	Verpflichtend	ISO/IEC 15408-3:2008 Abs. 5.2.4 + Tabelle 8 ISO/SAE 21434:2021 [RQ-05-11] + [RQ10-10] NHTSA ganzheitlich UN Regelung Nr. 155 Abs. 7.2.2.2. + Tabelle A1 - 28.1.
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Anforderungsreview Walk-Through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Error Guessing	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36

Testmodul		**Normative Forderung**
Testspezifikationsreview Walk-Through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2 ISO/SAE 21434:2021 [RQ-10-08] NHTSA G.36
Grenzwertanalyse	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Fuzzy Testing	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10] UN Regelung Nr. 155 Tabelle A1 - 28.1.
Statistischer Test	Empfohlen	UN Regelung Nr. 155 Tabelle A1 – 26.1 bis 26.3
Penetration Testing	Verpflichtend	ISO/IEC 15408-3:2008 Abs. 5.2.4 ISO/SAE 21434:2021 [RQ-10-10] NHTSA G.13 UN Regelung Nr. 155 Tabelle A1 – 5.1 bis 5.5
Schnittstellentest	Verpflichtend	ISO/SAE 21434:2021 [RQ-10-10]
Vulnerability Scan	Verpflichtend	ISO/IEC 15408-3:2008 Abs. 5.2.4 ISO/SAE 21434:2021 [RQ-10-10] NHTSA G.15 UN Regelung Nr. 155 Tabelle A1 - 29.1. + 29.2

Für CS sind verschiedene Normen von Bedeutung. Hauptsächlich spielt die ISO/SAE 21434:2021 (Road vehicles — Cybersecurity engineering") eine Rolle, da sie Anforderungen an den Entwicklungsprozess von Funktionen mit CS Relevanz stellt. In Abschnitt 10.4 der Norm werden Anforderungen an verschiedene Bereiche des Entwicklungsprozesses gestellt. Die Anforderung [RQ-10-10] fordert eine Spezifizierung der Verifikationsaktivitäten, sodass CS-Anforderungen erfüllt werden und listet dazu verschieden Testaktivitäten auf. In [RQ-05-11] wird zudem gefordert, dass ein Anforderungsmanagementsystem in der Organisation installiert wird. Dies führt zu der Schlussfolgerung, dass die Einhaltung der Anforderungen entsprechend durch SW-Tests verifiziert wird. Zudem soll nach [RQ10-08] die Vollständigkeit mit verschiedenen Methoden, in erster Linie Reviews oder statischen Analysen abgesichert werden.

Des Weiteren stellt Band 3 der Normreihe ISO/IEC 15408-x:200x wichtige Anforderungen an die Absicherung des BsM CS. Diese Norm stellt einige Anforderungen auf Spezifikationsseite, welche es zu testen gilt. Darüber hinaus wird ein ausgereifter und dem Zweck dienlicher Testprozess gefordert und Vorschläge für einige Testverfahren gemacht (Absatz 5.4.2 und Tabelle 8). Solch ein Testprozess kann auf der ISO/SAE 21434:2021 basieren.

In der UN Regelung Nr. 155 werden Anforderungen an das Managen von CS dargelegt. In Anhang 5 Tabelle A1 werden verschiedene Schwachstellen und Angriffsmethoden beschrieben. Daraus lassen sich diverse Anforderungen ableiten, die das Testobjekt einhalten muss. Diese Einhaltung kann mit verschiedenen Testverfahren nachgewiesen werden. Beispielsweise beschreibt die Bedrohung mit der ID 26 den mangelnden Einsatz oder die Beeinträchtigung kryptografischer Technologien und drei passende Angriffsmethoden mit den IDs 26.1 (schlechte Kombination von Verschlüsselungscodes), 26.2 (Unzureichender Einsatz) und 26.3 (veraltete Algorithmen). Mittels eines statistischen Tests kann nachgewiesen werden, dass die angedachten Technologien die statistisch notwendige Sicherheit bieten.

Abschließend beschäftigt sich die NHTSA (2020) mit Best Practices für CS. Einerseits werden technische Anforderungen an die Entwicklungsobjekte beschrieben und andererseits auch konkrete Testverfahren gefordert. Beispielsweise wird in [G.13] der Einsatz geeigneter Testprozesse und damit verbunden auch Penetration Tests gefordert. Auch diese Sammlung von Best Practices fordert geeignete Reviewmethoden ([G.36]).

Besonderes Merkmal Remote Updates

Die Vernetzung von Fahrzeugen ist ein immer wichtigeres Thema für Fahrzeughersteller und bietet neue Möglichkeiten, die Kundenzufriedenheit zu steigern und die Qualität der Funktionen und Fahrzeuge auch nach Verkauf zu verbessern, ohne Werkstattaufenthalte zu haben. Jedoch bergen solche Updates auch Gefahren, welche bereits während der Entwicklungsphase berücksichtigt werden müssen. Um diese Gefahren zu managen, sind auch hier Cybersecurity und funktionale Sicherheit von großer Bedeutung. Aber auch die Datensicherheit spielt hier eine wichtige Rolle. Um dies gewährleisten zu können, müssen verschiedene Testmethoden angewandt werden. (Jooß et al. 2022). Tabelle 4.16 und Tabelle 4.17 geben Aufschluss über die relevanten Testmodule für SW-Elemente für Remote Updates. Diese beiden Tabellen basieren auf den Ergebnissen aus Jooß et al. (2022).

Tabelle 4.16: Testmodule auf SWC Ebene Remote Updates

Testmodul		**Normative Forderung**
Anforderungsbasierter Test (funktional)	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_00006 + RS_FOTA_00016 + RS_FOTA_00030 GB 201-5 Abs. 4.3 + 5.1.5. ISO/DIS 24089:2022 Abs. 7.3.5.3 + 9.3.3.9 NHTSA ganzheitlich

Testmodul		**Normative Forderung**
Fault Injection Test	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_CONSTR_00002 + RS_FOTA_00013 + RS_FOTA_00006 GB 201-5 Abs. 4.4.4 ISO/DIS 24089:2022 Abs. 6.3.5.1 + 9.3.3.8 UN Regelung Nr. 156 Abs. 7.2.2.1.1
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Verpflichtend	GB 201-5 Abs. 4.2.3 + 4.2.5 + 4.2.6 ISO/DIS 24089:2022 Abs. 6.3.4.4 + 7.3.4.1 + 8.3.3.4 UN Regelung Nr. 156 Abs. 7.2.2.1.2
Fuzzy Testing	Verpflichtend	UN Regelung Nr. 155 Tabelle A1 - 28.1.
Statistischer Test	Empfohlen	UN Regelung Nr. 155 Tabelle A1 – 26.1 + 26.2 + 26.3
Technisches Modellreview	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_00006 GB 201-5 Abs. 4.2.7. IEEE Std 1028-2008 Abs. 5 ISO/DIS 24089:2022 Abs. 9.3.3.10 + 8.3.3.2 ISO/IEC 20246:2017 Annex F.2 UN Regelung Nr. 156 Abs. 7.1.1.7.
Modellreview Inspection	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_00005 GB 201-5 Abs. 4.2.4 + 4.4.7 ISO 26262-6:2018 Table 7 1c ISO/DIS 24089:2022 Abs. 7.3.5.1 ISO/IEC 20246:2017 Annex F.2 UN Regelung Nr. 156 Abs. 7.1.4.1 + 7.2.2.1.3 + 7.2.2.3 a), b) + 7.2.2.5

Testmodul		**Normative Forderung**
Technisches Codereview	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_00006 GB 201-5 Abs. 4.2.7. IEEE Std 1028-2008 Abs. 5 ISO/DIS 24089:2022 Abs. 9.3.3.10 + 8.3.3.2 ISO/IEC 20246:2017 Annex F.2 UN Regelung Nr. 156 Abs. 7.1.1.7.
Codereview Inspection	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_00005 GB 201-5 Abs. 4.2.4 + 4.4.7 ISO 26262-6:2018 Table 7 1c ISO/DIS 24089:2022 Abs. 7.3.5.1 ISO/IEC 20246:2017 Annex F.2 UN Regelung Nr. 156 Abs. 7.1.4.1 + 7.2.2.1.3 + 7.2.2.3 a), b) + 7.2.2.5
Performance Test	Verpflichtend	GB 201-5 Abs. 4.2.3 + 4.2.5 + 4.2.6 ISO/DIS 24089:2022 Abs. 6.3.4.4 + 7.3.4.1 + 8.3.3.4 UN Regelung Nr. 156 Abs. 7.2.2.1.2
Stresstest	Verpflichtend	UN Regelung Nr. 155 Tabelle A1 - 8.1 + 13.1 + 24.1
Entscheidungstest	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_00030 GB 201-5 Abs. 4.1.2 + 4.2.1 + 4.4.1 + 4.4.3 + 4.4.4 a) + 4.4.5 ISO/DIS 24089:2022 Abs. 6.3.2.1 f. + 6.3.3.2 + 6.3.4.1 ff. + 7.3.2.1 f. + 7.3.3.2 + 9.3.4.1 + 9.3.3.3. a) – e) + 9.3.3.6 + 9.3.4.3 + 9.3.4.5 + 9.3.6.1 UN Regelung Nr. 156 Abs. 7.1.1.11 + 7.2.2.2 a) – e) + 7.2.2.4 a), b)

Testmodul		Normative Forderung
MC/DC	Verpflichtend	GB 201-5 Abs. 5.1.1 + 5.1.2 + 5.1.3. + 5.2.1. ISO 26262-6:2018 Table 9 1c ISO/DIS 24089:2022 Abs. 5.3.4.2 a) – d) + 7.3.4.3 + 9.3.3.5 NHTSA T.1 + T.21 UN Regelung Nr. 155 Tabelle A1 - 29.1. + 29.2 UN Regelung Nr. 156 Abs. 7.2.1.1
Static Code Analysis	Verpflichtend	QS ohne normative Anforderung
Schnittstellentest	Verpflichtend	ISO/DIS 24089:2022 Abs. 6.3.2.3 + 8.3.3.3 + 9.3.2.8 UN Regelung Nr. 156 Abs. 7.1.1.5 + 7.1.1.10
Vulnerability Scan	Verpflichtend	UN Regelung Nr. 155 Tabelle A1 - 29.1. + 29.2

Tabelle 4.17: Testmodule auf SWS Ebene Remote Updates

Testmodul		Normative Forderung
Anforderungsbasierter Test (funktional)	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_00006 + RS_FOTA_00005 + RS_FOTA_00030 GB 201-5 Abs. 4.1.1 + 4.1.2 + 4.2.1 + 4.2.4 + 4.4.3 +4.4.4 a) + 4.4.5 + 4.4.7 + 5.1.1 + 5.1.2 + 5.1.3. + 5.2.1. ISO 26262-6:2018 Table 14 1a ISO/DIS 24089:2022 Abs. 5.3.4.2 a) – d) + 6.3.2.1 f. + 6.3.3.2 + 6.3.4.1 ff. + 7.3.2.1 f. + 7.3.3.2 + 7.3.4.3 + 7.3.5.1 + 7.3.5.3 + 9.3.3.3. a) – e) + 9.3.3.5 + 9.3.3.6 + 9.3.3.9 + 9.3.4.1 + 9.3.4.3 + 9.3.4.5 + 9.3.6.1 NHTSA ganzheitlich UN Regelung Nr. 155 Tabelle A1 - 28.1. UN Regelung Nr. 156 Abs. 7.1.1.11 + 7.1.4.1 + 7.2.1.1 + 7.2.2.1.3 + 7.2.2.2 a) – e) + 7.2.2.3 a), b) + 7.2.2.4 a), b) + 7.2.2.5

Testmodul		**Normative Forderung**
Fault Injection Test	Verpflichtend	AUTOSAR CP R20-11 RS_FOTA_CONSTR_00002 + RS_FOTA_00013 + RS_FOTA_00006 + RS_FOTA_00030 GB 201-5 Abs. 4.4.4 ISO/DIS 24089:2022 Abs. 6.3.5.1 + 7.3.5.3 + 9.3.3.8 + 9.3.3.9 UN Regelung Nr. 156 Abs. 7.2.2.1.1
Technisches Anforderungsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Anforderungsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Technisches Testspezifikationsreview	Verpflichtend	IEEE Std 1028-2008 Abs. 5 ISO/IEC 20246:2017 Annex F.2
Testspezifikationsreview Walk-through	Verpflichtend	ISO/IEC 20246:2017 Annex F.2
Grenzwertanalyse	Verpflichtend	GB 201-5 Abs. 4.2.3 + 4.2.5 + 4.2.6 ISO/DIS 24089:2022 Abs. 6.3.4.4 + 7.3.4.1 + 8.3.3.4 UN Regelung Nr. 156 Abs. 7.2.2.1.2
Fuzzy Testing	Verpflichtend	GB 201-5 Abs. 5.1.5. UN Regelung Nr. 155 Tabelle A1 - 28.1.
Statistischer Test	Verpflichtend	UN Regelung Nr. 155 Tabelle A1 - 26.1 + 26.2 + 26.3
Penetration Testing	Verpflichtend	UN Regelung Nr. 155 Tabelle A1 - 5.1 + 5.2 + 5.3 + 5.4 + 5.5
Schnittstellentest	Verpflichtend	GB 201-5 Abs. 4.1.2 + 4.2.1 + 4.4.1 + 4.4.3 ISO/DIS 24089:2022 Abs. 6.3.2.1 f. + 6.3.2.3 + 6.3.3.2 + 6.3.4.1 ff. + 7.3.2.1 f. + 7.3.3.2 + 8.3.3.3 + 9.3.2.8 + 9.3.3.3. a) – e) + 9.3.3.6 + 9.3.4.1 + 9.3.4.5 UN Regelung Nr. 156 Abs. 7.1.1.5 + 7.1.1.10 + 7.1.1.11 + 7.2.2.2 a) – e)
Vulnerability Scan	Verpflichtend	UN Regelung Nr. 155 Tabelle A1 - 29.1. + 29.2

Immer mehr Normen und Best Pratices behandeln das Thema Software-Updates Over-the-Air. Diese dienen als Anforderungsbasis, um verschiedene Testaktivitäten abzuleiten. Beispielsweise werden im Best Practices des NHTSA (2020) ganzheitlich Anforderungen an die Software vorgeschlagen, die es umzusetzen und abzusichern gilt. Aber auch die Veröffentlichung des AUTOSAR-Standards zum Thema OTA-Updates stellt solche Anforderungen. Beispielsweise werden in diesem Standard anforderungsbasierte Tests gefordert (AUTOSAR CP R20-11).

Die UN Regelung Nr. 155 stellt in Tabelle A1 Schwachstellen und Angriffsmethoden im Zusammenhang mit Software und deren Updates dar. Dies stellt eine Relevanz für das BsM dar und ermöglicht die Verknüpfung zu Testmodulen. Ähnlich werden in der UN Regelung Nr. 156 verschiedene Verfahren im Zusammenhang mit Software-Updates gefordert. Diese geforderten Verfahren bedürfen einer Absicherung, sodass verschiedene Module auf allen Ebenen abgeleitet werden können.

Anders als beispielsweise die Normreihe ISO 26262-x:201x stellen diese Normen weniger direkte Anforderungen an den Absicherungsmechanismus, sondern vermehrt Anforderungen an die Software, welche eingehalten werden müssen und so indirekt zu Testmodulen führen.

Anhand ausgewählter BsM wurde in diesem Kapitel gezeigt, wie die Testmodule mittels hermeneutischer Untersuchungen verschiedener Normen und Standards im Zusammenhang mit bestimmten Merkmalen entwickelt und zugeordnet wurden. Diese Methodik stellt die Grundlage für die folgenden Kapitel im Zusammenhang mit effizienten und agilen Testvorgehen dar.

4.4 Agile Testplanungsmethodik

Die klassische Fahrzeugentwicklung ist stark durch Termine geprägt (Abschnitt 3.1). Agile Entwicklungsabläufe sind hingegen weniger stark terminorientiert, sondern folgen dem Versuchen, ein funktionsfähiges SW-Element am Ende eines Zyklus zu liefern. Dies bedeutet, dass die strukturierte Abarbeitung der Entwicklungstätigkeiten im Vordergrund steht. Für einen festgelegten Zeitraum, häufig 3 Monate, werden alle durchzuführenden Aktivitäten geplant. So wird im agilen SAFe Framework ein sogenanntes Product increment (PI) Planning durchgeführt, um die Aufgaben für die kommenden drei Monate zu planen. Dieser Zeitraum wird PI genannt. Ziel ist die zeitliche, budgetäre und personelle Planung, um ein Entwicklungsartefakt vollständig abzuschließen. Ein PI wird weiter in Sprints von zwei bis drei Wochen unterteilt, in denen einzelne Inkremente des Produktes weiter einwickelt und getestet werden. (Scaled Agile, Inc. 2021)

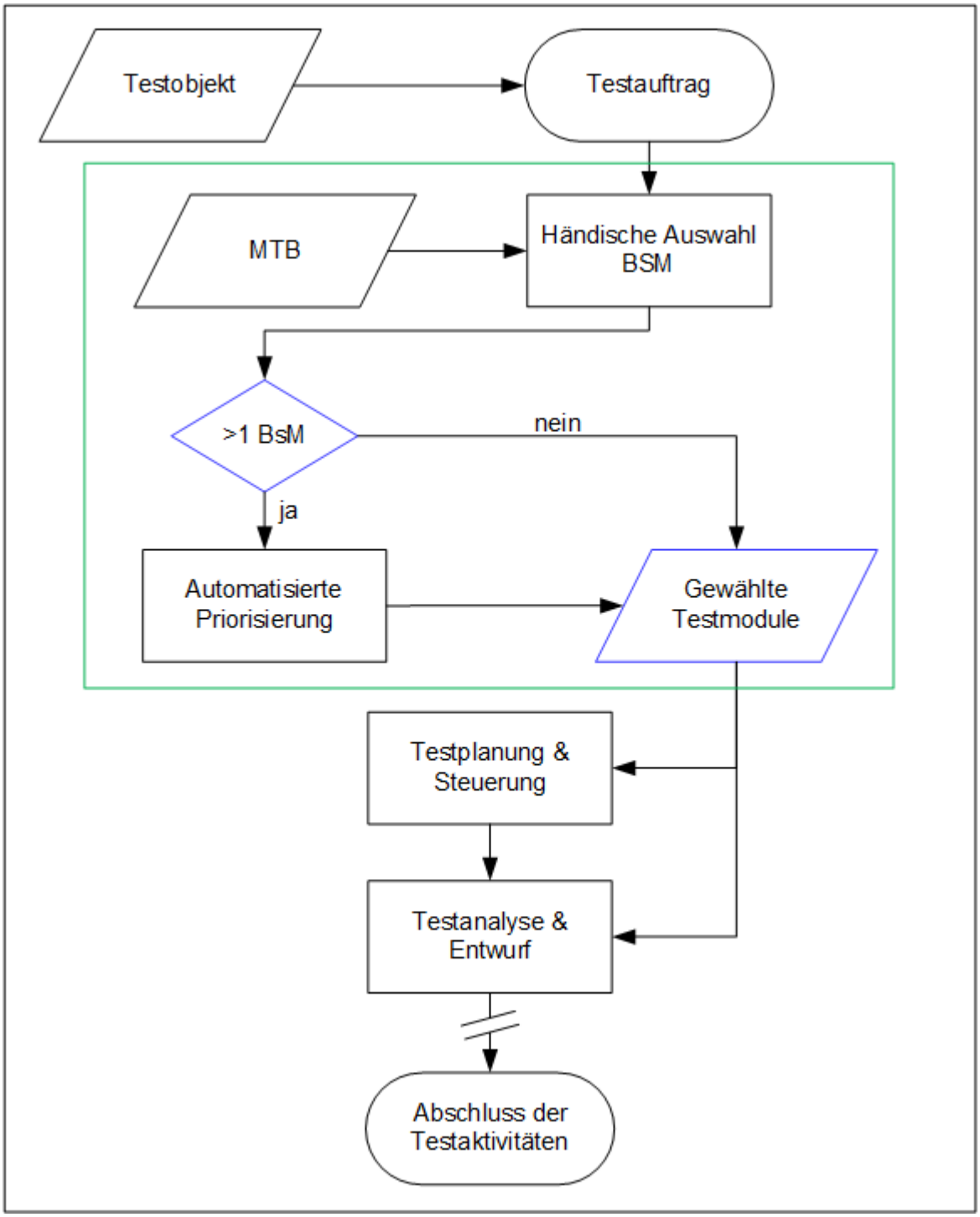

Abbildung 4.6: Anwendung des Baukastens im Testablaufprozess

Zu den zu planenden Aktivitäten zählen u. a. die Testumfänge. Deshalb muss bereits vor einem PI Planning der zu testende Umfang in Form einer Teststrategie bekannt sein. In dieser Planungsphase wird der modulare Testbaukasten (MTB) als Testplanungswerkzeug eingesetzt. In Abbildung 4.6 ist der Ablauf der Testplanung im fundamentalen Testprozess dargestellt. Zunächst wird ein Testauftrag erstellt bzw. geplant und dem Testobjekt die relevanten besonderen Merkmale zugeordnet. Wie im Abschnitt 4.3.4 beschrieben, erfolgt auf dieser Basis die Auswahl der korrekten Testmodule. In Abhängigkeit von den zugeordneten BsM und dadurch evtl. entstehender Redundanzen in den Testmodulen wird lediglich das Modul

mit der höheren Aussagekraft unabhängig vom Aufwand eingeplant und durchgeführt. Dadurch wird die maximale Effizienz durch den Einsatz des Baukastens sichergestellt. Zudem wird ein Überblick über die durchzuführenden Testmodule frühzeitig im Projekt generiert. Der Aufwand, der pro Modul benötigt wird, muss entsprechend beziffert werden, um richtig eingeplant zu werden. Diese Informationen werden alle während der Testplanung und -steuerung benötigt. Zudem kann der Testmanager damit auch die Aufwände für die kommenden Phasen, insbesondere den Testentwurf, besser einschätzen. So kann frühzeitig mit der Erstellung von Testfällen begonnen werden und notwendige Ressourcen beschafft werden.

Zur Aufwandsabschätzung ist es notwendig zu wissen, wann ausreichend getestet wurde. Das bezieht sich auf mögliche Testendekriterien. Häufig hängen diese von Zielterminen ab. Dies bedeutet, dass das eigentliche Testendekriterium nicht erreicht werden kann, sondern durch den Mangel an Zeit der Test frühzeitig beendet wird. Dieses Risiko besteht bei agilen Vorgehensweisen ebenso wie bei klassischen Entwicklungsmethoden, da man immer für einen festgelegten Zeitraum die Tests plant. Dieser Umstand wird in Expertenkreisen immer wieder diskutiert. So erörtert zum Beispiel Specht (2021), dass zeitlich bedingte Testendekriterien kritisch sind, da offensichtlich nicht ausreichend Informationen gesammelt wurden, um den Test vor einer Deadline abzuschließen. Darüber hinaus argumentiert er, dass die Messung von Abdeckungen (bspw. Anforderungs- oder Codecoverage) sinnvoll, aber in den kurzen Sprints kaum erreichbar sind. Er plädiert daher für systematisches Testen, um das Testobjekt möglichst genau untersuchen zu können und Informationen zu sammeln, die es ermöglichen, eine Abschätzung über Risiken oder Fehler geben zu können. Er sieht diese Informationen als ausreichendes Testendekriterium an.

Jedoch ist auch dieses Vorgehen problembehaftet. Einerseits kann auch hierbei die Zeit für die Tests am Ende eines Sprints zu knapp sein, um ausreichend Informationen zu sammeln und zum anderen muss die Komplexität eines Fahrzeuges einbezogen werden. Da Fahrzeuge hohe Anforderungen an die Sicherheit haben, sind die Absicherung aller Elemente und der Nachweis über die Durchführung notwendig. So wird auch in verschiedenen Normen, beispielsweise der Normreihe ISO 26262-x:201x, die Dokumentation aller Absicherungsmaßnahmen gefordert. Für die funktionale Sicherheit werden in Band 6 Absatz 9.4.4 die Erreichung verschiedener Abdeckungselemente gefordert. Aus diesen Gründen ist das von Specht (2021) vorgeschlagene Testendekriterium alleine nicht ausreichend.

Zur normgerechten Entwicklung müssen im Vorfeld Testendekriterien definiert und auch erreicht werden. Die Testendekriterien für die einzelnen Module werden in Abschnitt 4.3.3 genannt. Diese müssen erreicht werden. Deshalb wird es bereits bei der Testplanung einbezogen, um den zeitlichen Aufwand korrekt abschätzen zu können. Zudem müssen die Testebenen betrachtet werden.

Mit diesen Voraussetzungen ergibt sich ein Ablauf pro PI wie in Abbildung 4.7 dargestellt. Die Entwicklungsphase beginnt auf unterster Integrationsebene (Sprint 1), der Test wird

möglichst zeitnah begonnen, um Informationen zu generieren. Sobald die Tests positiv abgeschlossen sind (Sprint 3), wird im nächsten Sprint integriert und diese Ebene zeitnah getestet (Sprint 4). Je nach Ergebnis dieser Tests müssen im darauffolgenden Sprint (5) auf der vorangegangen, derselben oder der nächsten Integrationsebene Bugs gefixt und getestet werden. Erfolgsfaktor ist die genaue Planung und Aufwandsabschätzung im Rahmen eines PI Plannings. In jedem Sprint wird getestet oder auch ganze Sprints zum vollständigen Testen (Sprint 2) verwendet. Bei kleineren Bugs sind Regressionstests während eines Sprints ausreichend.

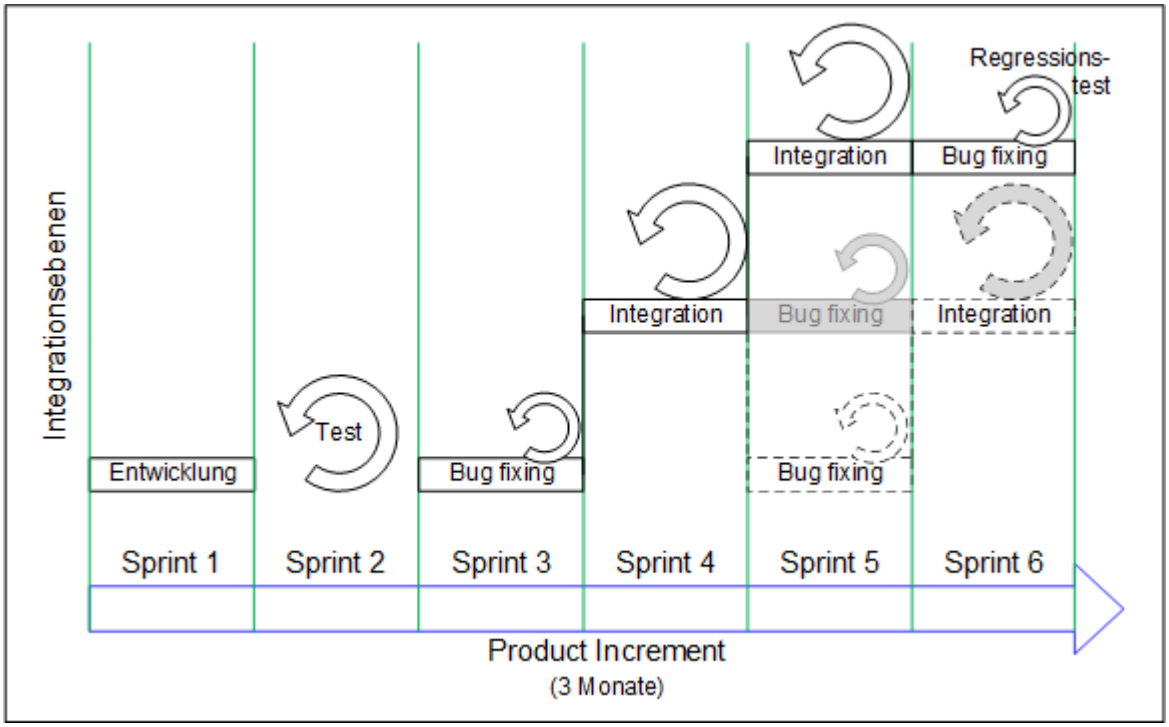

Abbildung 4.7: Entwicklungs- und Testablauf im agilen Umfeld (schematische Darstellung)

Zusammengefasst bedeutet dies, dass mittels einer Teststrategie frühzeitig festgelegt wird, welche Tests in welchem Sprint durchzuführen sind, um auch Meilensteine der Fahrzeugentwicklung bedienen zu können. Dies wird in Abschnitt 5.2 genauer betrachtet. Während eines PIs können die Tests, abhängig vom Reifegrad und unter Berücksichtigung agiler Rituale durchgeführt werden, um so zu gewährleisten, dass am Ende eines PIs die notwendige Qualität des Lieferproduktes vorliegt.

4.4.1 IT gestützte Testplanung

Die Planung und Durchführung aller Testaktivitäten kann sehr aufwendig und zeitintensiv sein. Um dies möglichst effizient darstellen zu können, ist ein IT gestütztes Vorgehen sinnvoll. In diesem Abschnitt wird ein Konzept zu IT gestützter Testplanung vorgestellt.

Laut dem fundamentalen Testprozess ist das Ziel der Testplanung die Erstellung oder Anpassung des Testkonzeptes, um die Testziele und Testaktivitäten festzulegen. Durch die Testüberwachung wird die Erreichung der Ziele fortlaufend über den gesamten Testprozess

geprüft, um Status und Abweichungen im Blick zu haben und bei Bedarf Maßnahmen einleiten zu können. (ATB et al. 2011)

Um dies möglichst effizient zu erreichen, muss das IT-Konzept folgende Ziele erfüllen:

- Gesamtheitlich in Toolkette integriert.
- Hoher Automatisierungsgrad.
- Nachweis der Normeinhaltung.

Die Gesamtintegration in eine Toolkette bietet mehrere Vorteile: So ist eine kontinuierliche Überwachung realisierbar, Maßnahmen können bei Abweichung schnell eingeleitet werden und der Aufwand während des gesamten Testprozesses wird effizient gestaltet. Aus diesem Grund ist auch ein hoher Automatisierungsgrad notwendig. Je mehr manuelle Arbeit anfällt, desto aufwendiger ist die Testplanung, was im agilen Umfeld viel Zeit für wenig Wertschöpfung in Anspruch nimmt. Dies führt auch zum dritten Ziel, dem Nachweis zur Normeinhaltung. Werden bei der Testplanung bereits relevante Normen oder Gesetze beachtet, kann durch vollumfängliche Traces zwischen den einzelnen Anforderungen, Testzielen, Testfällen und Testergebnissen der Nachweis schnell und aufwandsarm erbracht werden.

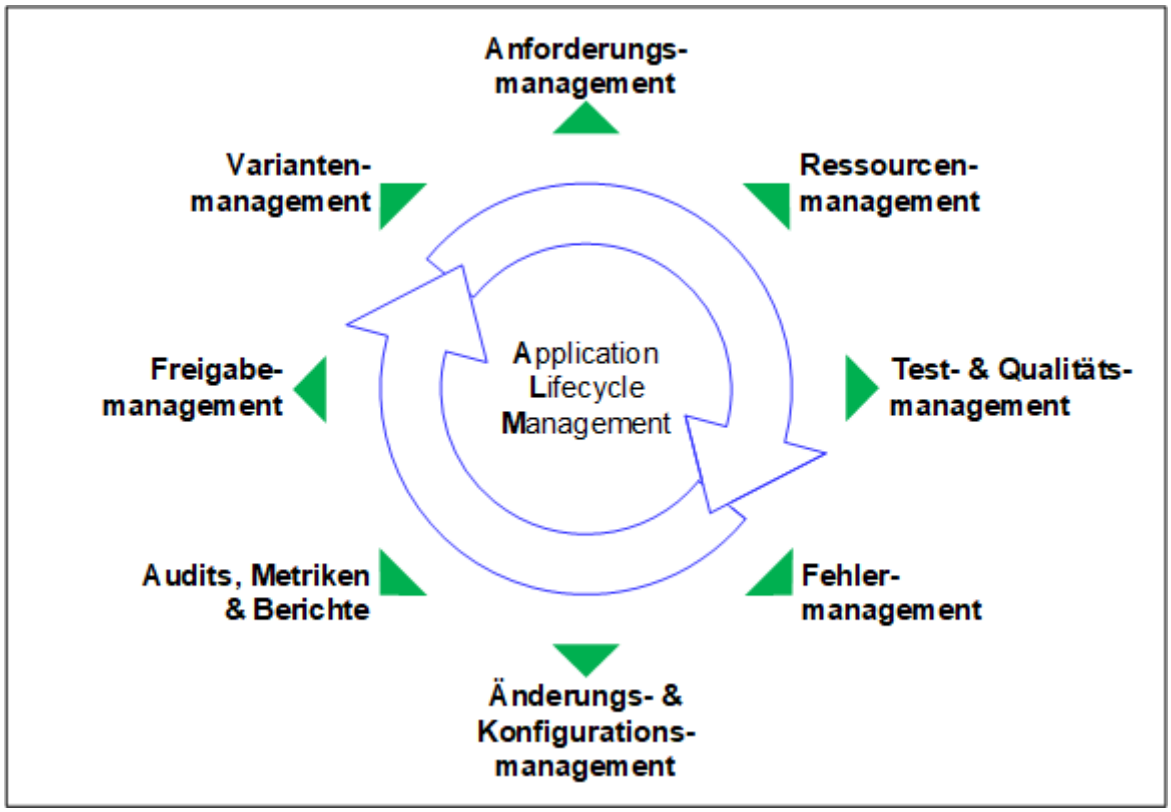

Abbildung 4.8: Application Lifecycle Management (modifiziert nach Delos Santos 2022)

Mit diesen Prämissen wurde ein Konzept zur IT-gestützten Testplanung erstellt. Als Grundlage für das Konzept dient der Application Lifecycle Management (ALM) Ansatz. Darunter versteht man einen Ansatz, bei dem ein Produkt, bspw. ein Softwaretool oder eine Fahrwerkfunktion, über den gesamten Lebenszyklus gemanagt wird. Darunter fallen anfängliche

Planungen für das Produkt über Änderungen im Zeitraum der Herstellung bis hin zur Außerbetriebnahme bzw. dem Ende des Supports durch den Hersteller. (TechTarget, Inc. 2015). Abbildung 4.8 zeigt die durch ALM unterstützten Disziplinen vom Anforderungsmanagement über das Testmanagement bis hin zum Freigabemanagement.

Durch eine einheitliche Toolunterstützung können alle entstehenden Artefakte (z. B. Anforderungen, Testfälle, Auffälligkeiten) miteinander verknüpft werden, um automatisierte Traces zwischen den Artefakten zu erzeugen. Durch diese ist es so möglich, einen schnellen Überblick über den Status eines Produktes und ggf. notwendige Anpassungen zu erhalten. Zeitgleich werden alle Normen erfüllt, die solche Traces fordern. Aufgrund der Komplexität von Fahrzeugen ist dies zur gesamthaften Bestimmung des Entwicklungs- und Reifegradstatus notwendig.

Das Konzept zum Testplanungstool (siehe Abbildung 4.9), welches im Rahmen dieser Arbeit entwickelt wurde, setzt an dem Punkt der Trace-Erstellung an und sieht eine Integration in ein ALM Tool und damit der Verknüpfung mit der gesamten Toolkette vor. Für ein ALM-Tool, wie z. B. CodeBeamer der Intland Software GmbH bedeutet dies, dass eine Datenbank integriert oder angebunden sein muss, in der die Testmodule hinterlegt sind. Die Datenbank enthält Informationen zu Testzielen, Testeingangs- und -ausgangskriterien sowie zum Testentwurfsverfahren.

Im Rahmen der Testplanung werden dem Testobjekt unter Zuhilfenahme der Anforderungen (link mittels Trace im ALM-Tool vorahnend) die BsM zugeordnet. Für jede Testebene werden dadurch automatisiert die Testmodule zugeordnet werden, um im nächsten Schritt des fundamentalen Testprozesses „Testanalyse & Entwurf“ die Testfälle pro Testinstanz zu entwickeln und zu pflegen. Dazu muss die Testbasis vollumfänglich vorhanden und die Testeingangskriterien erfüllt sein. Dies kann durch ein ALM-Tool automatisiert angezeigt werden.

Ein weiterer wesentlicher Aspekt, den das Konzept vorsieht, ist die Aufwandsabschätzung zum Zeitpunkt der Testplanung. Dabei ist für jedes Testmodul ein geschätzter Aufwand (Durchführungszeit, Personenstunden für Entwurf, Durchführung und Analyse) angegeben. Dies wird für die Planung der Testdurchführung während der Sprints verwendet, um genauere Abschätzungen zu den möglichen Testfällen machen zu können. Die Aufwände werden während der Testentwurfsphase der Testdurchführung und Testbewertung erhoben und in das ALM-Tool zurück gespiegelt. Dadurch können zukünftigen Sprints mit einer höheren Planungssicherheit hinsichtlich des tatsächlich benötigten Aufwands durchgeführt werden.

ALM-Tools ermöglichen standardmäßig die Konfiguration der Anforderungs- und Testebenen. Dies ist auch notwendig, um nachzuweisen, dass die Testmethoden und Testverfahren, welche in den Normen gefordert werden Anwendung finden. Dieser wichtige Aspekt des Konzepts ermöglicht einen Nachweis der eingehaltenen Normen zu jedem Zeitpunkt im Projektverlauf.

Durch eine gute Verwaltung und Konfiguration der hinterlegten Daten, werden während der Testplanung viele Informationen automatisiert ausgegeben und berechnet. Um auch während der folgenden Phasen einen hohen Automatisierungsgrad zu erhalten, ist es wichtig, dass das ALM-Tool Schnittstellen zu den Testtools bspw. TPT der PikeTec GmbH oder TestWeaver von Synopsis bietet. Durch eine hohe Integration des ALM-Tools in die Testtoolketten können Traces von den Testmodulen zu den Testfälle und Testergebnisse gezogen werden und so der Testfortschritt kontinuierlich überwacht werden. Mit einem hohen Automatisierungsgrad der gesamten Toolkette wird die Effizienz gesteigert und die für einen Sprint geplanten Testaktivitäten können mit größerer Wahrscheinlichkeit vollumfänglich durchgeführt werden.

Zudem ist eine Schnittstelle zwischen den Tools wichtig, um Testberichte automatisiert und standardisiert auszuleiten. Dazu werden während der Bewertungsphase die Testergebnisse analysiert und bewertet. Diese Bewertung wird in das ALM-Tool überführt, um einen Testbericht zu erzeugen. Diese Berichte müssen im ALM-Tool an das Freigabemanagement übergeben werden, um Meilensteine des PEP zu erreichen.

Mit diesem Konzept werden die gesetzten Ziele erreicht. Allerdings ist die Umsetzung stark abhängig von der eingesetzten Toolkette. So müssen potenzielle ALM Tools die entsprechenden Schnittstellen zur Verfügung stellen, um die benötigen Informationen weitergeben und verwalten zu können. Zudem spielen auch die Schnittstellen zu den einzelnen Testtools eine große Rolle. Um den generischen Ansatz der Arbeit nicht außeracht zu lassen, wurde in dieser Arbeit ein Konzept, welches projekt- oder unternehmensspezifisch umzusetzen ist, erarbeitet.

Die erzielten Ergebnisse des Tests und der Nachweis zur Einhaltung verschiedener Normen und Gesetze ist relevant für die Freigabe der Funktionen. Darauf wird im folgenden Kapitel vertieft eingegangen.

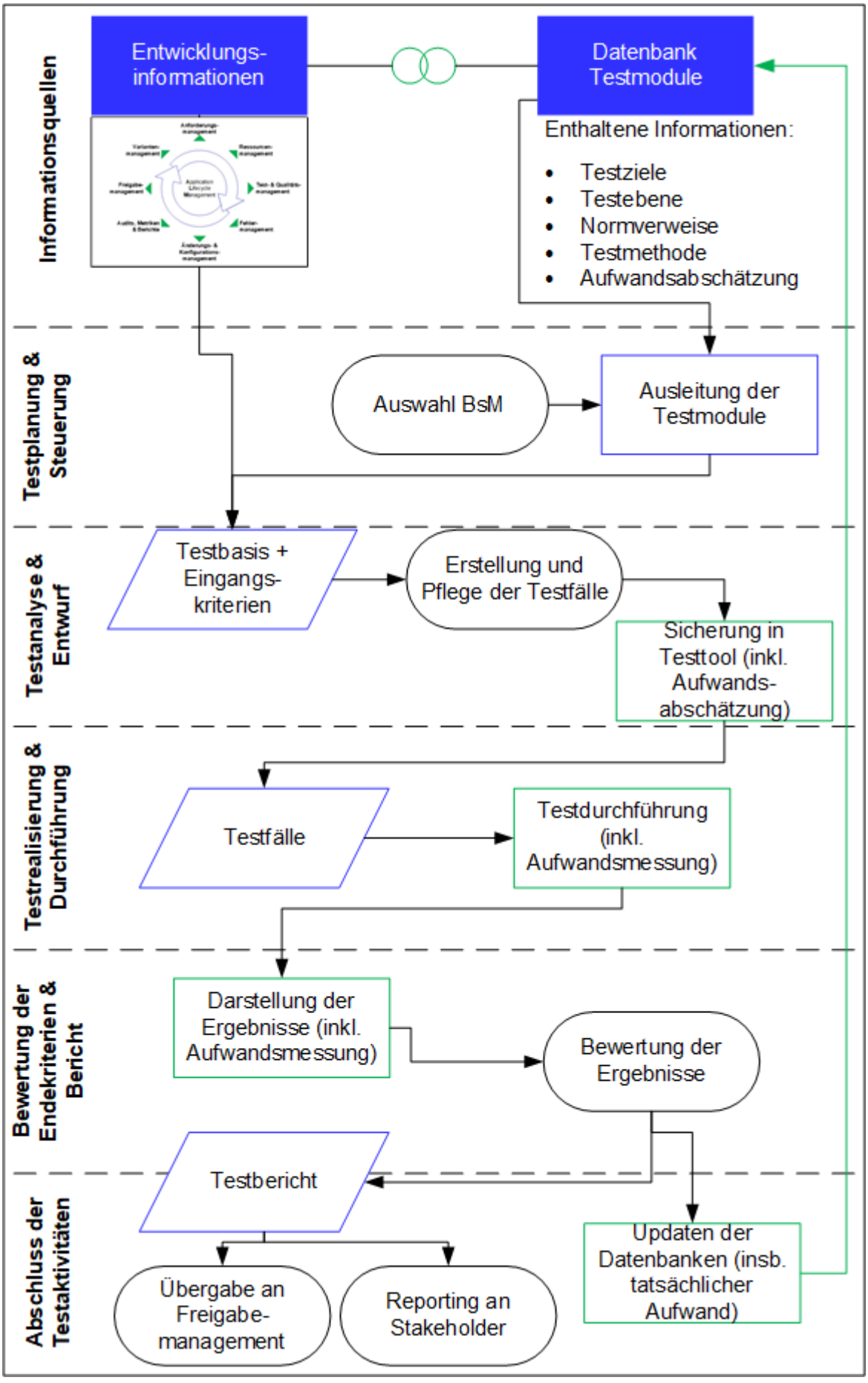

Abbildung 4.9: Konzept Testplanungstool

5 Freigaben im Entwicklungsprozess

Tests dienen nicht allein dem Selbstzweck, sondern werden im Entwicklungsprozess für die Freigaben benötigt. Um möglichst effizient zu testen, ist es sinnvoll, die Testendekriterien an die Freigabelevel zu knüpfen. Zudem kann es im Verlauf der Entwicklung zu zeitlichen Engpässen beim Testen kommen. In solchen Fällen ist es notwendig, zielgerichtet die wichtigsten und aussagekräftigsten Tests durchzuführen, um auf dieser Basis eine Freigabe erteilen zu können. In diesem Kapitel werden die Grundlagen der Freigaben und eine damit verbundene neuartige Methode im agilen Umfeld vorgestellt. Anschließend wird die Notwendigkeit für die Testfallpriorisierung erläutert und entsprechende Maßnahmen vorgestellt.

5.1 Grundlagen der Freigabemethodik

Die theoretischen Grundlagen der Freigabe wurden in Abschnitt 3.4 hinreichend beschrieben. In diesem Abschnitt wird hingegen erörtert, welche Informationen benötigt werden, um ein Entwicklungsartefakt bspw. eine Fahrwerkfunktion freigeben zu können. Davor werden potenzielle Freigabelevel und Integrationsstufen in Abhängigkeit der bisher vorgestellten Teststrategie vorgestellt.

Freigaben werden in der Praxis meist für verschiedene Verwendungszwecke und auf verschiedenen Integrationsebenen erteilt. Eine pauschale Aussage über diese Zwecke und Ebenen lässt sich schwer treffen, Abbildung 5.1 stellt daher schematisch verschiedene Ebenen (y-Achse) und Zwecke (x-Achse) dar.

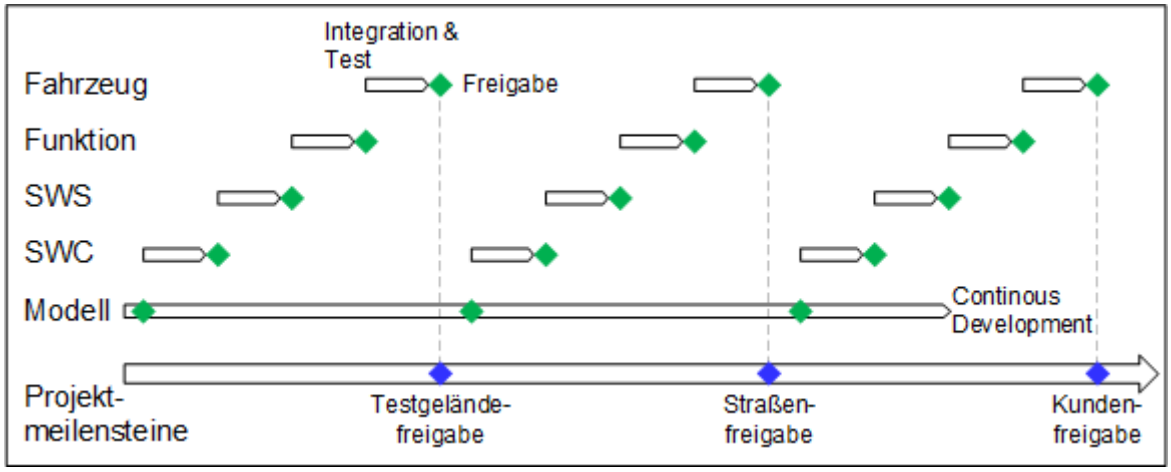

Abbildung 5.1: Freigabeebenen über einen zeitlichen Verlauf (schematische Darstellung)

Zunächst wird die Zielsetzung genauer betrachtet. Darunter wird hier der Verwendungszweck für das Gesamtfahrzeug verstanden. Dies bedeutet, dass jede Freigabe eines Artefakts für genau diesen Zweck ausgelegt ist. Typischerweise umfasst dies die Berechtigung, ein Fahrzeug mit dem freigegebene Entwicklungsstand auf einem abgeschlossenen

Testgelände oder der öffentlichen Straße durch ausgebildete Fahrer und Entwickler zu bewegen. Der wichtigste Freigabezweck ist die Bestätigung, dass die Fahrzeuge einen Serienstand erreicht haben und für Endkunden nach dem Stand der Technik entwickelt wurden. Dieser als Kundenfreigabe bezeichnete Meilenstein ist in der Regel kurz vor SOP und schließt die Entwicklung für ein bestimmtes Fahrzeugderivat ab. Die meisten Normen sind so ausgelegt, dass deren Erfüllung zu diesem Zeitpunkt erreicht werden muss. Beispielsweise müssen die von der Normreihe ISO 26262-x:201x geforderten Entwicklungsschritte und -aktivitäten bis zu diesem Zeitpunkt vollständig erfüllt und abgeschlossen sein.

Weiterhin muss die zeitliche Komponente bei einer Freigabe betrachtet werden. In Abschnitt 3.4.1 wird die Freigabe definiert als Abschluss einer Entwicklungsphase und die Bestätigung, dass das Entwicklungsobjekt in der nächsten Entwicklungsphase für einen bestimmten Zweck eingesetzt werden kann. Daraus ergibt sich, dass Freigaben zu einem Meilenstein im PEP, z. B. BST oder SOP, erteilt werden, um die Folgephase beginnen zu können. Für die in dieser Arbeit entwickelte Methode bedeutet dies, dass Freigaben der unteren Integrationsebene entsprechend vorgelagert sein müssen, um die Gesamtfreigabe zu diesen Meilensteinen einzuhalten. Dadurch ergibt sich ein abgestufter Zeitplan wie in Abbildung 5.1 zu sehen.

Anschließend werden die Integrationsebenen vorgestellt. Die Freigabe bzw. Integrationsebenen unterscheiden sich von denen in Abschnitt 4.3.2 vorgestellten Ebenen des Baukastens. Diese Ebenen dienen lediglich der Zuordnung von Testmodulen zu einem bestimmten Testobjekt. Die getesteten Objekte müssen jedoch für die nachfolgende Integration freigegeben werden. Generell werden daher alle einzelnen Elemente freigegeben - für diese Arbeit sind folglich Softwareunits (SWU) oder -module die kleinsten freizugebenden Einheiten. Die freigegeben Units werden anschließend zu SWCs integriert, getestet und freigegeben. Anschließend erfolgt die Integration mehrerer SWCs in Softwaresysteme und damit häufig auch in das Steuergerät. Gemeinsam mit verschiedenen mechatronischen Elementen ist dies die Integrationsebene der Funktion auch als System bezeichnet. Alle Funktionen und Bauteile werden in ein Gesamtfahrzeug integriert. Dies stellt die oberste Test- und Freigabeebene dar. Grundlage einer Freigabe sind vorangegangene Freigaben der darunterliegenden Integrationsebene. Diese werden durch ergänzende Tests für die jeweilige Ebene bestätigt.

Zu den Integrationsebenen SWU, SWC und SWS muss erwähnt werden, dass sie, wie in Abschnitt 1.1.1 vorgestellt, modellbasiert entwickelt und integriert werden können. Alternativ ist es möglich, aus den Modellen auf jeder Ebene Code zu generieren, welcher anschließend integrieret wird. Im Verlauf dieser Arbeit wird davon ausgegangen, dass die SWUs als Modell zu einer SWC integriert wird. Aus diesem Modell auf SWC-Ebene wird dann der Code generiert, um diesen dann als SWS in das Steuergerät zu integrieren.

Abschließend werden die Inhalte bzw. Entwicklungsinformationen, die zu einer Freigabe benötigt werden, beschrieben. Für eine Freigabe müssen verschiedene Informationen zusammengetragen werden. Grundlegend sind die in Abbildung 5.2 aufgezeigten Traces. Mit

diesen sind verschiedene Entwicklungsdokumente miteinander verknüpft, sodass es nachvollziehbar ist, welche Teile eines Modells oder Codes welche Anforderung abdeckt und mit welchem Testfall getestet wird.

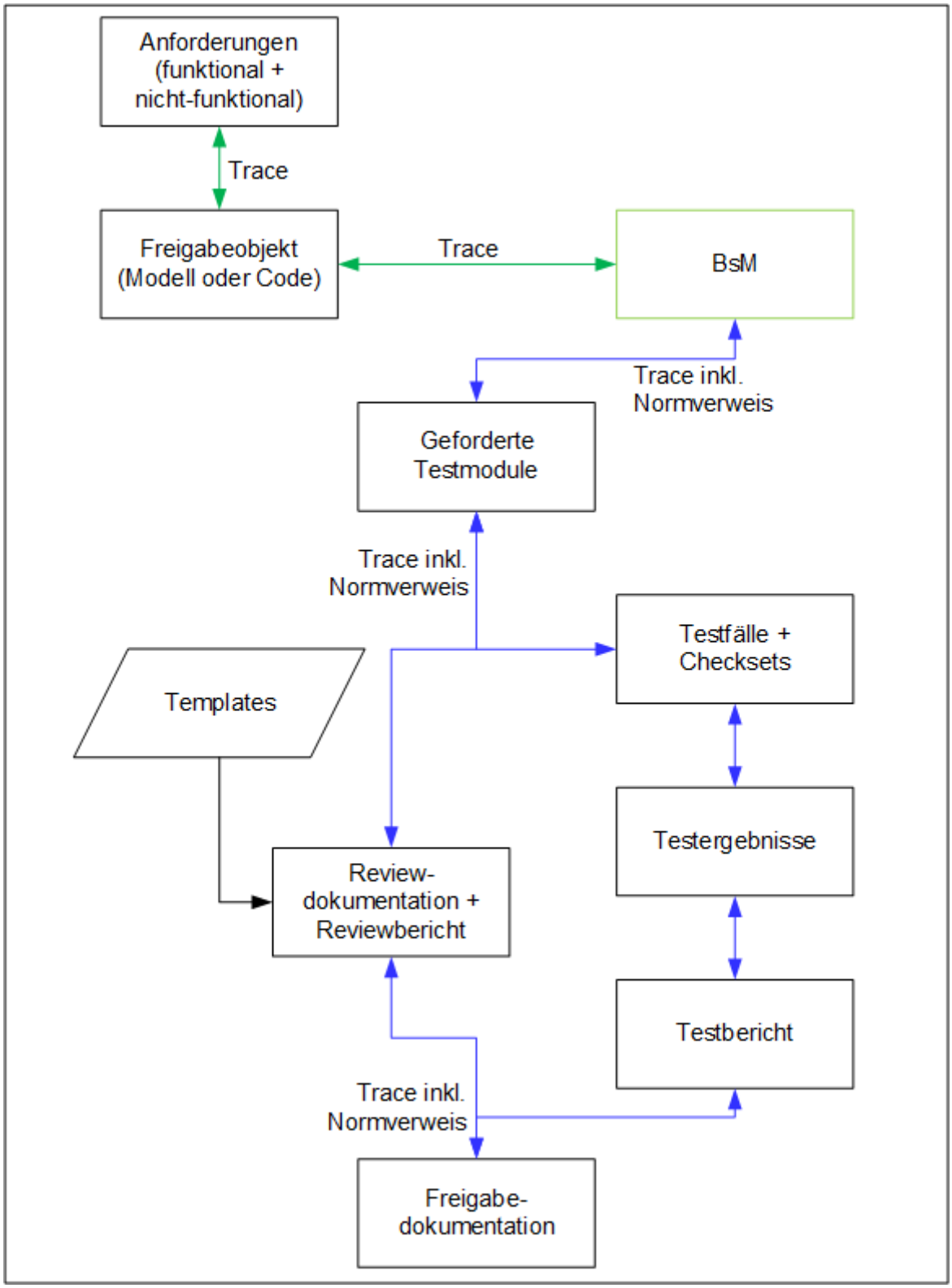

Abbildung 5.2: Traceabilitykonzept zur Freigabe

In dieser Arbeit wird das Konzept zusätzlich um die besonderen Merkmale ergänzt. Damit wird übersichtlich dargestellt, welche Normen und Gesetze ein Freigabeartefakt erfüllen

muss. Zeitgleich ist ein direkter und generischer Nachweis erbracht, ob Normanforderungen hinsichtlich Tests eingehalten wurden. Dies dient als übersichtliche Bewertungsgrundlage für die Freigabe. Alle gesammelten Informationen werden in dem in Abschnitt 4.4.1 vorgestellten Konzept zur IT gestützten Testplanung und über ein erweitertes ALM Tool bereitgestellt.

Die hier vorgestellten Grundlagen dienen als Basis für die agile Freigabemethodik, die im folgenden Absatz vorgestellt wird.

5.2 Agile Freigabemethodik

Agile und klassische Entwicklungsvorgehen unterscheiden sich in vielen Punkten von agilen Modellen. Insbesondere die starren Zeitpläne (PEP) in der klassischen Fahrzeugentwicklung passen nicht zu den agilen Vorgehen in der Softwareentwicklung. Im Folgenden wird eine Methode vorgestellt, die versucht, diesen Konflikt aufzulösen.

Zunächst ist es wichtig, Freigabezwecke und Meilensteine zu synchronisieren. Ausgangsbasis bildet der PEP eines Fahrzeugderivates, wie in Abbildung 5.3 dargestellt. Dieser wird durch verschiedene Meilensteine gegliedert, welche hier generisch als Freigabemeilensteine (FMS) bezeichnet werden. Beispielhaft sind in der Abbildung Freigabezwecke aufgeführt und enden mit dem FMS 3 zum Zweck der Kundenfreigabe. Da die meisten Fahrwerkfunktionen in mehreren Derivaten eines Modells oder mit Anpassungen auch in verschiedenen Modellen eingesetzt werden, ist hier schematisch noch ein weiterer PEP eines zweiten Derivats dargestellt. Dieser ist zeitliche versetzt, um die unterschiedlichen Zeitpläne zu verdeutlichen.

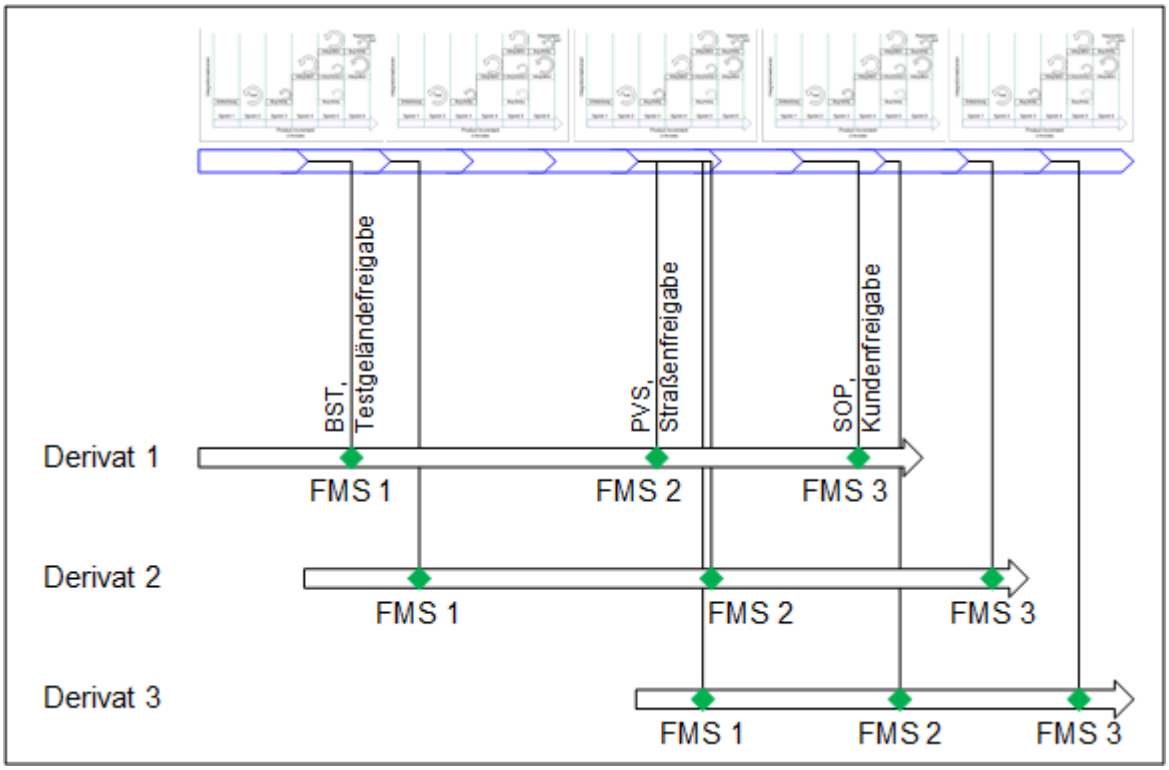

Abbildung 5.3: Freigabeablauf in der agilen Entwicklung

In den PI-Plannings werden diese FMS berücksichtigt. Wesentlich ist, dass der Abschluss eines PI's zeitlich ausreichend vor einem FMS liegt. Dies ist notwendig, um weitere Tests auf Fahrzeugebene durchführen zu können. Zudem ist zu erkennen, dass Freigaben für beide Derivate oder nur für ein Derivat gültig sein können.

Die in Abschnitt 4.4 gezeigte Testplanungsmethodik wird während der PIs durchgeführt, um entsprechende Testdurchführungen durchlaufen zu können. In frühen Projektphasen eines Fahrzeugprojektes unterscheiden sich die Erfüllungsgrade der Testabschlusskriterien im Vergleich zu den späten Projektphasen und sind abhängig vom Freigabezweck.

Dies bedeutet, dass die Erfüllungsgrade der Testabschlusskriterien sowohl abhängig vom Einsatzzweck in der nächsten Entwicklungsphase als auch von den besonderen Merkmalen sind. So müssen sicherheitskritische Freigabeartefakte auch bei einer Fahrt auf dem Testgelände mit erfahrenen Testfahrern funktionieren. Funktionen ohne Sicherheitskritikalität müssen die an sie gestellten Anforderungen hingegen erst zu einer Kundenfreigabe vollumfänglich erfüllen.

Die Testendekriterien sind jedem Testmodul zugeordnet (Abschnitt 4.3.3). Dort wird der Erfüllungsgrad zu einer Kundenfreigabe angegeben (entspricht 100 %). Für Freigaben ist ein in Abhängigkeit vom Sicherheitsrisiko angepasstes Testendekriterium zu erreichen. Eine standardisierte Methode zur Bestimmung des Testendekriteriums wurde entwickelt. In einem ersten Schritt wurden Risiken identifiziert, welche bei unzureichenden Tests für einen Freigabezweck auftreten können:

- **Sicherheitsgefährdung**
  Darunter wird die Sicherheit für das Leben der Fahrer und weiterer Verkehrsteilnehmer verstanden
- **Finanzieller Verlust**
  Durch unzureichende Tests können verschiedene finanzielle Verluste für einen OEM auftreten, bspw. durch Verlust von Prototypen oder Reputationszahlungen
- **Imageverlust**
  Fehler, die auftreten, können zu Imageverlust führen, sei es aufgrund von Qualitätsmängeln oder aufgrund von Unfällen, die damit in Verbindung gebracht werden
- **Informationsverlust**
  Darunter werden sowohl Informationen zum Entwicklungsobjekt als auch von Kundendaten verstanden
- **Gefährdung des Zeitplans**
  Durch Fehler, die durch unzureichende Tests nicht rechtzeitig identifiziert werden, wird der Entwicklungszeitplan gefährdet.

Mittels eines paarweisen Vergleichs wurden diese fünf Risiken gewichtet (Tabelle 5.1).

Tabelle 5.1: Paarweiser Vergleich zur Risikogewichtung

	Sicherheitsgefährdung	Finanzieller Verlust	Imageverlust	Informationsverlust	Gefährdung des Zeitplans	Summe	**Gewichtung**
Sicherheitsgefährdung		2	2	2	2	8	40 %
Finanzieller Verlust	0		1	1	1	3	15 %
Imageverlust	0	1		0	2	3	15 %
Informationsverlust	0	1	2		2	5	25 %
Gefährdung des Zeitplans	0	1	0	0		1	5 %
					∑	20	100 %

Im nächsten Schritt wurde die Eintrittswahrscheinlichkeit für den Freigabezweck mittels einer Nutzwertanalyse bewertet. Die VDI Richtlinie VDI2225 Blatt 3 beschreibt eine Bewertungsmethode für Konstruktionen, bei der zwischen null und vier Punkten vergeben werden können, wobei null Punkte für eine unbefriedigende Lösung und vier Punkte für die Ideallösung vergeben werden. Zur Ermittlung der Punkte wird eine Ideallösung angenommen und mit den angegebenen Punkten wird bewertet, wie nah die Entwürfe an diese herankommen. Dieses Prinzip lässt sich auch auf die Eintrittswahrscheinlichkeit übertragen. Hierbei bedeuten null Punkte, jedoch keine Eintrittswahrscheinlichkeit und vier Punkte eine sehr hohe Eintrittswahrscheinlichkeit. Tabelle 5.2 zeigt die Eintrittswahrscheinlichkeiten und einen Eintrittsfaktor für das weitere Vorgehen für drei beispielhafte Freigabezwecke. Zur Gesetztes- und Normerfüllung müssen für den Zweck Kundenfreigabe 100 % erreicht werden.

Tabelle 5.2: Eintrittswahrscheinlichkeit der Risiken für verschiedene Freigabezwecke

	Gewichtung	Testgelände	Straße	Kunde
Sicherheitsgefährdung	40 %	2	3	4
Finanzieller Verlust	15 %	0	2	4
Imageverlust	15 %	0	2	4
Informationsverlust	25 %	1	3	4
Gefährdung des Zeitplans	5 %	4	4	4
		1,25	2,75	4
	Eintrittsfaktor	**31 %**	**69 %**	**100 %**

In einem letzten Schritt wurde das Gefahrenniveau für die besonderen Merkmale bestimmt. Diese sind hier durch die sechs Merkmale aus Abschnitt 4.2 dargestellt. Das Gefahrenniveau wird in Prozentwerten angegeben (Tabelle 5.3). Dieses wird mit dem Eintrittsfaktor multipliziert, um so einen Flexfaktor zu erhalten.

$$Gefahrenniveau * Eintrittsfaktor = Flexfaktor$$

Zur Berechnung des Testüberdeckungsgrades wird ein fixer Anteil festgelegt. Der fixe Anteil beträgt für die Testgeländefreigabe 70 %, die Straßenfreigabe 75 % und die Kundenfreigabe 100 %. Dieser stellt das niedrigste angepasste Testendekriterium dar und dient der Sicherstellung, dass ein Mindestumfang getestet wird. Der variable Anteil wird mit dem Flexfaktor multipliziert und dem fixen Anteil subtrahiert, um das angepasste Testendekriterium zu erhalten.

$$Angepasstes\ Testendekriterium = fixer\ Anteil + \ variabler\ Anteil * Flexfaktor$$

Tabelle 5.3 stellt die Ergebnisse für das gewählte Beispiel dar.

Tabelle 5.3: Angepasste Testendekriterien

	Gefahrenniveau	Angepasstes Testendekriterium Testgelände	Angepasstes Testendekriterium Straße
Qualitätssicherung	20 %	72 %	78 %
ASIL A	60 %	76 %	85 %
ASIL D	100 %	79 %	92 %
SOTIF	100 %	79 %	92 %
Cybersecurity	80 %	78 %	89 %
Remote Updates	80 %	78 %	89 %

Ziel der Freigabemethode ist die Erreichung von Testendekriterien mit einer Abstufung für Freigabemeilensteine vor Kundenfreigabe. Damit wird dem Umstand genüge getan, dass eine Entwicklung inkrementell abläuft und nicht von Beginn an alle Funktionalitäten im Fahrzeug verfügbar sind. Mit dieser Abstufung lässt sich zu Beginn der Testplanung entsprechend der Testaufwand einplanen und steuern.

5.3 Notwendigkeit und Ziele der Testfallpriorisierung

Die Priorisierung verschiedener Artefakte im Testprozess ist ein wesentliches Element in der de facto Norm des ISTQB. Demnach sollen bereits während der Testanalyse die Testbedingungen priorisiert werden, um abstrakte Testfälle zu entwerfen. Diese müssen wiederrum während der Testrealisierungsphase priorisiert werden, um einen möglichst effizienten Testablauf zu gewährleisten. (ATB et al. 2020)

Die Notwendigkeit besteht, da im Rahmen des Testprozesses und der Testplanung mittels des Testbaukastens eine Vielzahl verschiedener Testfälle entwickelt wird. Um die begrenzten Ressourcen an Prüfständen und Testtools effizient zu nutzen, sollten diese priorisiert werden.
Zusätzlich besteht die Notwendigkeit, da die Softwareentwicklung im Fahrzeugumfeld abhängig von starren Zeitplänen des PEPs ist. Dies bedeutet, dass trotz der Flexibilität in agilen Vorgehensmodellen Meilensteine erreicht werden müssen, um die nächste Entwicklungsphase zu erreichen. Daraus ergibt sich der Bedarf, verschiedene Tests zu priorisieren.

Das Ziel der Priorisierung ist folglich die zeitliche Ausplanung der Durchführungsreihenfolge anhand definierter Kriterien, um Testziele zu erreichen (Rothermel et al. 1999). Nach den Autoren einer empirischen Studie zur Testfallpriorisierung können damit folgende fünf Ziele erreicht werden:

- Schnellere Fehleridentifizierung und Steigerung der Fehleridentifizierungsrate.
- Frühzeitige Erkennung von Fehlern mit hohen Risiken.
- Erhöhung der Fehleridentifizierung im Rahmen von Regressionstests.
- Erhöhung verschiedener Überdeckungsraten, z. B. Codeüberdeckung.
- Erhöhung des Vertrauens in das Testobjekt in kurzer Zeit.

Zusammengefasst soll folglich die Effizienz beim Testen gesteigert werden, um ein oder mehrere Ziele in möglichst geringer Zeit zu erreichen. (Rothermel et al. 1999)

5.3.1 Maßnahmen zur Testfallpriorisierung

Eine umfassende Testpriorisierungsstrategie ist jedoch sehr zeitaufwendig und widerspricht teilweise dem Ziel der Effizienzsteigerung. Daher basieren die Methoden zu Priorisierung meist auf Faustregeln. Diese sollen mit dem begrenzten Wissen über die Testobjekte eine Entscheidungshilfe bieten (Rothermel et al. 1999). In der Literatur werden verschiedene heuristische Methoden vorgestellt, welche mit mehr oder minder großem Aufwand umsetzbar sind.

So haben Quoc Tuan Le et al. (2009) eine mehrstufige Priorisierungsmethode entwickelt, bei der eine Vielzahl an Einflussfaktoren, die in Zusammenhang mit der Funktion stehen, mittels des Analytic Hierachy Process (AHP) bewertet, um eine mutlikriterielle Entscheidung zu treffen. Mittels des AHP werden die Einflussfaktoren hinsichtlich ihrer Kritikalität priorisiert. Diese Einflussfaktoren werden im nächsten Schritt auf die Testfälle gemappt, um diese anschließend anhand von Regeln zu priorisieren. Diese werden wiederum durch AHP feinpriorisiert. Dieser Prozess scheint sehr ressourcenintensiv und steht damit auf den ersten Blick im

Widerspruch zur Effizienzsteigerung durch die Priorisierung. Allerdings werden hier wesentliche Punkte wie Sicherheitsaspekte, Vernetzungsgrade oder Image- und Kostenrisiken als Einflussfaktoren genannt. Dies greift den heuristischen Gedanken auf.

Dieser wird auch von weiteren Experten aufgegriffen. Beispielsweise werden Methoden erläutert, die darauf aufbauen, dass anhand verschiedener Coverages Testfälle priorisiert werden. Die Priorisierung erfolgt anhand der Testfälle, welche die höchste Abdeckung haben. (GeeksforGeeks 2020)

Sattler (2015) hingegen hat eine nach eigenen Aussagen „unaufwendige“ Methode auf der Basis von Kriterien entwickelt. Die Testfälle werden anhand verschiedener Anforderungen an ein System in eine Reihenfolge gebracht werden. Das wichtigste Kriterium ist die ASIL-Einstufung. Darüber hinaus sollte aber auch die Kritikalität i. S. v. Auswirkungen bei Fehlverhalten, Zentralität, Neuheitsgrad und Aufwand berücksichtigt werden. An einem Beispiel erklärt sie, dass durch den Fokus auf die ASIL-Einstufung lediglich 27 % der Testfälle Relevanz aus der Perspektive der ISO 26262 haben. Auch wenn dies nur beispielhaft ist, lässt sich daran erkennen, dass die Priorisierung deutlich vereinfacht werden kann und die wichtigsten Testfälle zuerst durchgeführt werden.

Sattler konzentriert sich in Ihrer Arbeit auf das Testen von ASIL-Systemen. Trotz des Blickwinkels auf lediglich ein BsM, ist der Gedanke einer einfachen und unkomplizierten Methode auch für die vorliegende Arbeit zielführend. Die Arbeit von Quoc Tuan Le et al. (2009) ist deutlich zeitaufwendiger. Jedoch erhoffen sich die Autoren einen Vorteil durch eine schnelle Testplanung im weiteren Projektverlauf, insbesondere durch einen heuristischen Ansatz, der in Zusammenhang mit Testfallpriorisierung immer wieder auftritt.

Zwischen den beiden vorgestellten Methoden und weiteren Methoden bestehen Gemeinsamkeiten. Meistens werden die Testfälle anhand verschiedener Kriterien oder Faktoren priorisiert. Auch in dieser Arbeit wird dies aufgegriffen, um möglichst effizient Testfälle zu priorisieren und eine Ablaufreihenfolge zu bestimmen.

Die Kriterien, die Sattler (2015) in ihrer Arbeit vorschlägt, erscheinen hinreichend sinnvoll. Da die vorliegende Arbeit jedoch generisch ist und mehrere BsM Beachtung finden, ist eine Anpassung der Kriterien notwendig. Unter anderem werden unterschiedliche Sicherheitsrisiken berücksichtigt, die nicht vergleichbar sind. Daher kann anhand dieser Risiken keine Priorisierung festgelegt werden. Aus diesem Grund ist es nicht ausreichend, eines davon in den Fokus zu stellen und dies als Priorisierungsmerkmal zur Etablierung einer Testdurchführungsreihenfolge zu nutzen. Aufgrund dessen werden folgende angepasste Kriterien zur Testfallpriorisierung vorgeschlagen:

1. Gefahrenniveau:
   In Abschnitt 5.2 wurde methodisch das Gefahrenniveau für die besonderen Merkmale bestimmt, um eine Testabdeckung für eine Freigabe erteilen zu können. Dieses gibt eine Indikation für verschiedene BsM hinsichtlich Risiko, welche auch zur Testfallpriorisierung genutzt werden.
2. Ausfallrisiko:
   Funktionen und Softwareelemente, die mit hoher Wahrscheinlichkeit zu einem Ausfall des Systems oder der Funktion führen, werden mit höherer Priorität getestet, um das Ausfallrisiko zu reduzieren.
3. Qualitätsrisiko:
   Um die Qualität sicherzustellen, ist es notwendig, dass alle Elemente einer Funktion getestet werden. Funktionen, die im Projektverlauf wenig oder überhaupt nicht getestet wurden, müssen priorisiert werden, um eine Testabdeckung vor SOP zu gewährleisten.
4. Abdeckungsrate:
   Testfälle, die größere Teile eines Softwareelements testen, sind solchen, die nur kleine Teile abdecken, vorzuziehen. Dadurch kann mittels eines einzelnen Testfalls schneller eine größere Abdeckung erzielt werden.
5. Durchführungsdauer:
   Neben der Abdeckungsrate ist auch die Durchführungsdauer eine wichtige Eigenschaft des Testfalls. Testfälle werden unter Berücksichtigung der verbliebenen Zeit priorisiert. Testfälle mit hohen Ausführungszeiten sollten nur priorisiert werden, wenn ausreichend Zeit verblieben ist.
6. Veränderungsrate
   Dieses Kriterium bezieht sich wieder auf die Funktion. Funktionen mit vielen Veränderungen sind zu priorisieren, um Fehler zu erkennen und zu vermeiden. Auch dieses Kriterium ist vergleichbar mit dem „Neuheitsgrad“ in der Arbeit von Sattler (2015), bedenkt jedoch nicht nur neue, sondern auch stark veränderte Funktionen.

Diese Kriterien sollen bei der Testplanung als Anhaltspunkte genutzt werden, um in einem agilen Zeitplan Testfälle zu priorisieren und die fünf Ziele der Priorisierung zu erreichen. Selbstverständlich können nicht immer alle Kriterien gleichzeitig Anwendung finden. In der Praxis müssen daher jene Kriterien gewählt werden, welche zielversprechend sind und helfen, eine Freigabestufe im Zeitplan zu erreichen.

6 Einsatz der Methode

Die Methode konzentriert sich auf den rechten Ast des V-Modells. Dabei werden sowohl Aspekte des Testprozesses als auch des Freigabeprozesses berücksichtigt. Da die Automobilbranche aktuell im Umbruch ist, z. B. durch neue softwarebasierte Technologien und agile Methoden, müssen die Prozesse an diese neuen Strukturen angepasst werden. In diesem Kapitel wird daher gezeigt, wie die Methode während der Entwicklungsabläufe zum Einsatz kommt.

6.1 Einordnung der Methode

In Kapitel 3 wurde der Produktentstehungsprozesse, damit verbundene Vorgehensmodelle und im Detail der Test- und der Freigabeprozess vorgestellt. Auf diese beiden Teilprozesse konzentrieren sich auch die Ergebnisse der Arbeit (siehe Abbildung 6.1) Während des Produktentstehungsprozesses werden sowohl Tests- als auch Freigaben mehrfach wiederholt bzw. ausgesprochen. Insbesondere die Freigabe basiert dabei auf den vorangegangenen Schritten. Die dabei erzeugten Arbeitsergebnisse müssen bei der Erteilung der Freigabe berücksichtigt werden. Besonders elementar sind dabei die Testergebnisse. Diese dienen als Nachweis, um zu bewerten, ob die Freigabeziele eingehalten wurden und ob ein Entwicklungsobjekt für die nächste Entwicklungsphase genutzt werden kann.

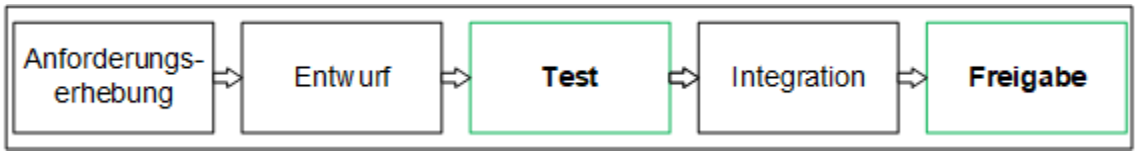

Abbildung 6.1: Einsatzbereiche der Methode im Entwicklungsablauf

Da die Methoden, die in dieser Arbeit entwickelt wurden, für verschiedene Phasen angedacht sind, wird deren Einsatz in den folgenden Abschnitten getrennt betrachtet. Im Abschnitt 6.1.1 wird zunächst der Einsatz der Testmethode beschrieben, in Abschnitt 6.1.2 der Einsatz der Freigabemethode und das Zusammenwirken der beiden Methoden.

6.1.1 Einsatz der Testmethode

Die Methode zielt darauf ab, Testmanager und Tester bei der Testplanung und Testfallerstellung zu unterstützen. Abbildung 6.2 zeigt auf, in welchen Schritten des fundamentalen Testprozesses die Methode Anwendung findet. Insbesondere in den Phasen der „Testplanung“ und „Testanalyse & Entwurf“ werden verschiedene Aspekte der Methode eingesetzt. Ganzheitlich ist die Methode als übergeordnete Teststrategie zu verstehen. Aus dieser Strategie werden in der ersten Phase des Prozesses projektspezifische Testkonzepte abgeleitet.

Ziel ist insbesondere die Festlegung der durchzuführenden Testmodule und damit der Testziele. Zudem können zu diesem Zeitpunkt auch die notwendigen Ressourcen bestimmt werden.

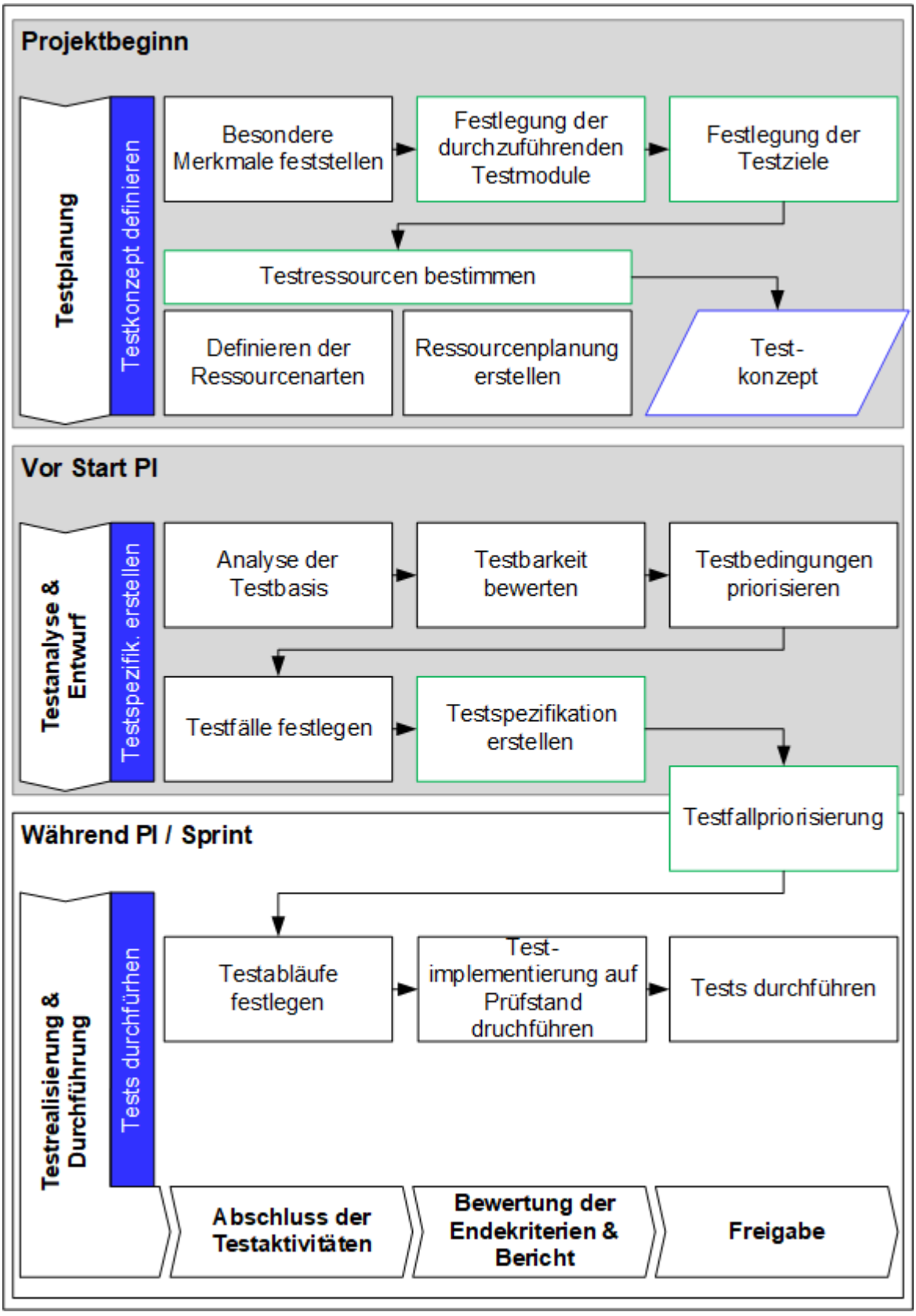

Abbildung 6.2: Einsatz der Testmethode im Testprozess

Im Vergleich zu bereits etablierten Teststrategien bietet die Methode den Vorteil, dass mehrere besondere technische Merkmale berücksichtigt werden und somit alle Aspekte der Fahrwerkentwicklung. Dadurch werden alle Tests nach denselben Vorgaben der Teststrategie durchgeführt. In der Literatur sind dafür bisher mehrere unterschiedliche Strategien vorgesehen. Mit der Verwendung einer einheitlichen Strategie ist eine gleichbleibende Qualität der Testumfänge über das gesamte Entwicklungsprojekt gewährleistet.

In einem agilen Kontext betrachtet, erfolgt diese frühe Phase einmalig zu Projektbeginn. Dabei werden Testziele festgelegt, die auf Basis der technischen Merkmale entschieden werden. Für den Fall, dass sich Änderungen im Entwicklungs- und Testablauf ergeben, werden die Testziele im Rahmen der Teststeuerung angepasst. Die frühe Planung ist unter anderem notwendig, um Ressourcen beschaffen zu können. Dazu zählen Testinfrastruktur und qualifiziertes Personal. Zur effizienten und strukturierten Testplanung wird dabei das ALM-Tool mit einer Anbindung zur Datenbank des MTB, wie in Abschnitt 4.4.1 beschrieben, eingesetzt.

In der zweiten Phase des Testprozesses hilft die Festlegung der Testziele dabei, die Testspezifikation zu erstellen. Die geplanten Module enthalten ferner eine Beschreibung des Testentwurfsverfahrens, welches Testspezifikateure dazu befähigt, für jedes Module passende Testspezifikationen abzuleiten. Dieser Schritt wird im Entwicklungsablauf mehrfach wiederholt, da Anpassungen der Testspezifikationen essenziell sind. Hintergründe sind mögliche Änderungen des Entwicklungsobjektes oder von Anforderungen. Konkret wird dieser Schritt vor jedem PI durchgeführt. Mit diesem Vorgehen ist sichergestellt, dass in einem PI mit dem aktuellen Stand der Testspezifikationen gearbeitet wird.

Im Übergang zur nächsten Phase der „Testrealisierung & Durchführung“ findet die Testfallpriorisierung statt. Dazu wurden Maßnahmen in Abschnitt 5.3.1 vorgestellt, die hier zur Anwendung kommen. Die Testdurchführung wird im Laufe eines PIs zu jedem Sprint wiederholt. Die Terminkritikalität kann sich stets verändern. Dies führt dazu, dass eine vollumfängliche Testabdeckung nicht erreicht wird. Deshalb ist es wesentlich, die Priorisierung frühzeitig festzulegen. So ist es für den Tester auch in zeitlich engen Phasen möglich, schnell die Tests mit der größten Priorität durchzuführen. Auf Basis der Testspezifikationen in Verbindung mit der Priorisierung werden Testfälle auf den entsprechenden Prüfständen implementiert, Testabläufe erstellt und anschließend die einzelnen Testfälle abgearbeitet. Damit ist diese Phase ebenso wie die restlichen Phasen nicht direkt von der Methode betroffen. Lediglich der Einsatz des ALM-Tools und das automatisierte Ziehen von Traces zu anderen Arbeitsprodukten des Entwicklungsprozesses ist in den letzten drei Phasen des fundamentalen Testprozesses vorgesehen und zählt als Teil der in dieser Arbeit entwickelten Methode.

Zusammenfassend ist festzuhalten, dass die Testmethode auf die Optimierung des Testprozesses durch frühzeitige, strukturierte und transparente Planung baut und die in Abbildung 6.2 grün hervorgehobenen Schritte des Testprozesses unterstützt.

6.1.2 Einsatz der Freigabemethode

Tests bilden gemeinsam mit den Anforderungen die Basis für die Freigabe einzelner Entwicklungsobjekte. Da die Teilprozesse Test und Freigabe eng miteinander verzahnt sind, wurde in dieser Arbeit zusätzlich zur Testmethode eine Freigabemethode entwickelt. Beide Methoden zielen ganzheitlich auf die effiziente Gestaltung des Entwicklungsablaufes ab. Freigaben werden ebenso wie Tests auf verschiedenen Ebenen erteilt. Dies wird durch den freizugebenden Gegenstand bspw. eine SWC oder eine Funktion bestimmt. In Abbildung 6.3 ist dargestellt, dass durch die Freigabeebene die Testebene und damit die Testmodule festgelegt werden. Im vorangegangenen Abschnitt 6.1.1 wurde der Einsatz der Methode zur Testfallpriorisierung bereits beschrieben. Die Priorisierung hat ebenfalls Auswirkungen auf die Freigabe. Dadurch entstehende Risiken müssen während der Freigabe berücksichtigt werden. Sollten falsche Maßnahmen zur Priorisierung ergriffen worden sein oder unzureichende Tests durchgeführt werden, ist davon die Freigabe betroffen und kann verweigert werden. Durch die Verwendung des Konzeptes zum ALM-Tool werden die entsprechenden Informationen transparent kommuniziert.

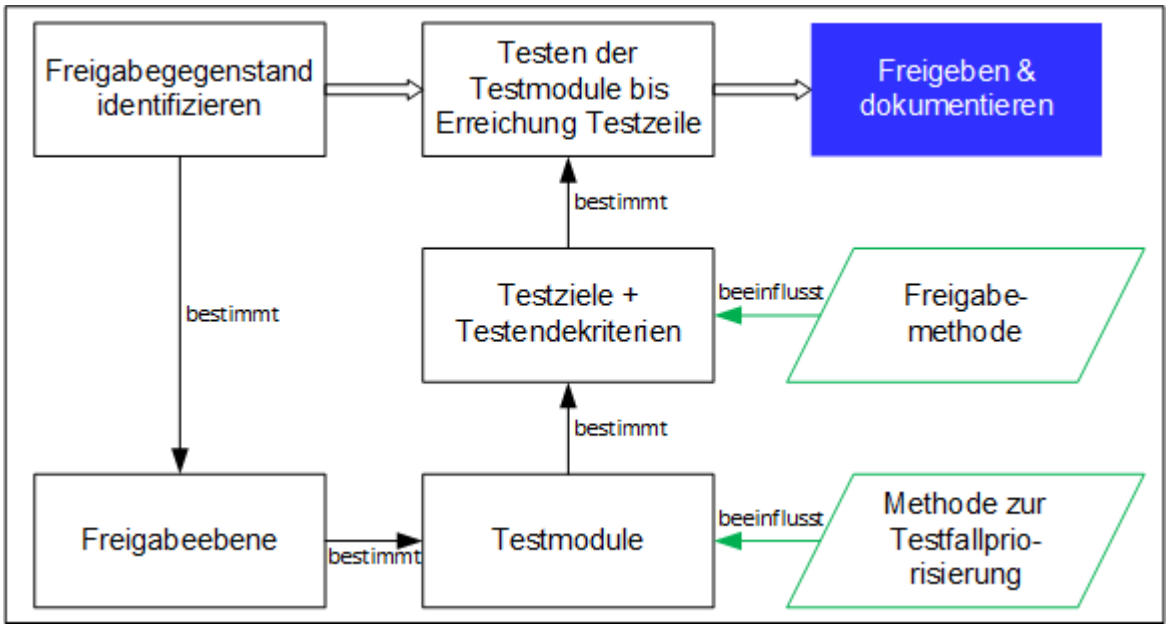

Abbildung 6.3: Einsatz der Freigabemethode im Freigabeablauf

Durch die (priorisierten) Testmodule werden auch die Testziele und Testendekriterien festgelegt. Die Notwendigkeit der Erreichung der Ziele wird durch die Freigabemethode beeinflusst. In Abschnitt 5.2 wird aufgezeigt, dass die Testendekriterien in Abhängigkeit vom Freigabezweck angepasst werden. Dazu werden verschiedene Risiken definiert und gewichtet sowie deren Eintrittswahrscheinlichkeit bestimmt. Dabei spielt auch das BsM eine Rolle, sodass für jedes Merkmal angepasste Testendekriterien für die Freigabezwecke „Einsatz auf dem Prüfgelände“ und „Einsatz auf der Straße durch eingewiesenen Entwickler“ abgeleitet werden. Die Kundenfreigabe ist davon nicht betroffen, da mit dieser finalen Freigabe bestätigt wird, dass die Sicherheit und Qualität der Funktionen für die Nutzung durch Kunden möglich ist.

Dieses Vorgehen kalkuliert die kurzen und iterativen Sprintzyklen der agilen Vorgehensweise ein. Zudem basiert die Methode auf der Interpretation von Normen, sodass die Einhaltung der geltenden Gesetze zur Kundenfreigabe gesichert ist.

Auch für die Freigabemethode bietet das Toolkonzept zur Testplanung aus Abschnitt 4.4.1 entscheidenden Vorteile. Durch die Traces zwischen den einzelnen Arbeitsprodukten ist einfach nachvollziehbar, welche Anforderungen mit welchen Tests abgedeckt wurden. Zudem können alle durchgeführten Testmodule inkl. der Maßnahmen zur Priorisierung erfasst werden. Auf dieser Basis können Verantwortliche schnell nachvollziehen, welche Risiken für eine Freigabe bestehen und ob auf dieser Basis eine Freigabe als letzter Schritt erteilt werden kann. Diese Informationen werden ebenso im ALM-Tool dokumentiert und an alle Stakeholder verteilt.

Die enge Verknüpfung zwischen einer Testmethode und Freigabemethode wurde in diesem Abschnitt nochmals verdeutlicht. Ein vollumfängliches Konzept, wie in dieser Arbeit vorgestellt, unterstützt bei effizienten Abläufen in Entwicklungsprojekten und ermöglicht damit ein Arbeiten nach agilen Vorgehensweisen ohne Produktentstehungsprozesse, wie sie im Automobilbau gängig sind, außer Acht zu lassen.

7 Fazit

In dieser Arbeit wurde eine generische Methode zur optimierten Absicherung und Freigabe von Fahrwerkfunktionen vorgestellt. In diesem abschließenden Kapitel wird die Arbeit zusammengefasst, der wissenschaftliche Beitrag dargestellt und ein Ausblick zu Weiterentwicklung und potenziellen Forschungsarbeiten gegeben.

7.1 Zusammenfassung

In Fahrzeugen gibt es zunehmend komplexe Systeme, die überwiegend auf Mechatronik basieren. Zu Innovationen und Entwicklung neuer Funktionen trägt im Wesentlichen der Informatikanteil mechatronischer Systeme bei (Abbildung 1.1). Von diesem Einfluss sind alle Domänen der Fahrzeugentwicklung betroffen. Diese Arbeit konzentriert sich jedoch ausschließlich auf Fahrwerkfunktionen.

Der Informatikanteil bzw. die Software ist für Automobilhersteller ein wesentlicher Faktor zur Wettbewerbsdifferenzierung, der die Entwicklungsaktivitäten beeinflusst. Aus diesem Grund ist es wichtig, dass alle Prozesse, die mit der Entwicklung von Software zusammenhängen, beherrscht werden. Unter Software werden alle Elemente verstanden, die notwendig sind, um eine logische Rechenoperation auf einem Steuergerät auszuführen. Im Rahmen dieser Arbeit zählen dazu Modelle von SW-Units oder SW-Komponenten und daraus generierter Code. Jedoch sind auf einigen Testebenen die Basis-Software, das Run-Time-Environment oder das Steuergerät relevant, um eine Funktion zu testen.

Testen zählt zu den wichtigsten Prozessen im Zusammenhang mit SW-Entwicklung. So sollte bereits auf unterster Ebene (im Falle von modellbasierter Entwicklung ist es das Modell) mit dem Testen begonnen werden, um Folgekosten so gering wie möglich zu halten. Die anschließende Integration zu größeren SW-Elementen, Funktionen oder Systemen und damit verbundene Tests sind unter anderem notwendig, um die Qualität sicherzustellen, das Risiko von Rückrufen aufgrund von Softwarefehlern zu minimieren sowie die Kundenzufriedenheit zu gewährleisten. Jedoch unterliegen Entwicklungs- und Testprozesse im Automobilumfeld einem Wandel.

Dieser kann auf drei Hauptursachen zurückgeführt werden:

1. **Technologien**: Eine steigende Anzahl an Funktionalitäten wird im Automobilbau umgesetzt. Dies führt einerseits zu einer Zunahme an abzusichernder Software, andererseits wird die Vernetzung zwischen Steuergeräten und Funktionen komplexer. Teilweise liegt dies daran, dass die Steuergeräte rechenstärker werden und so die Anzahl an Funktionen pro Steuergerät steigt.
2. **Regularien**: Durch weltweite Veränderungen und neue Technologien werden Gesetze und Normen angepasst. Diese beeinflussen die Entwicklung und müssen zur

Homologation von Fahrzeugen eingehalten werden. Zudem werden bestehende Regularien umfangreicher, wie das Beispiel in Abschnitt 1.1.2 verdeutlicht.

3. **Entwicklungsmethoden**: Bisher waren relativ starre Meilensteinpläne die zeitliche Hauptvorgabe für Entwicklungsprojekte. Da es bei der Entwicklung von SW regelmäßig zu unerwarteten Ereignissen kommt, wird mehr Flexibilität benötigt, als es die Zeitpläne zulassen. Um dem entgegenzuwirken, werden vermehrt agile Entwicklungsmethoden eingesetzt, um reaktiver auf unerwartete Vorkommnisse eingehen zu können.

Ziel dieser Arbeit war die Entwicklung einer generischen Methode zur optimierten Absicherung und Freigabe von Fahrwerkfunktionen. Bisherige Teststrategien zielen meist auf die Absicherung eines technischen Merkmals, beispielsweise FuSi, Security oder Qualitätssicherung (obere Hälfte Abbildung 7.1). Ziel dieser Arbeit war hingegen, eine Methode zu entwickeln, die verschiedene technische Aspekte von Software berücksichtigt, umso eine generische Anwendung für die Fahrwerkentwicklung zu ermöglichen. Um dieses Ziel zu erreichen, wurde ein sogenannter modularer Testbaukasten entwickelt (untere Hälfte Abbildung 7.1), der verschiedene Teststrategien zu einer einzigen vereint. Dadurch soll vermieden werden, dass ein Mehraufwand während der Testdurchführung entsteht. Bei der parallelen Betrachtung mehrere Teststrategien kann dieser durch die Durchführung verschiedener Testaktivitäten mit demselben Testziel entstehen. Der modulare Aufbau der in dieser Arbeit vorgestellten Methode vermeidet dies, da Testaktivitäten pro Testobjekt abgeglichen werden und so das Testziel nur einmalig abgesichert wird.

Zunächst wurden 13 besondere technische Merkmale entwickelt. Die Basis bilden die drei besonderen Merkmale des VDA. Jedoch wurden diese deutlich verfeinert, um ausreichend Granularität für eine modulare Strategie zu bieten. In einem nächsten Schritt wurden die Testmodule identifiziert. In einem Testmodul wurden alle wesentlichen Informationen beschrieben, die ein Testmanager zur Testplanung benötigt. Dies umfasst u. a. Testziele, Testmethoden und Testendekriterien. Zur Identifizierung wurden verschieden Normen hermeneutisch untersucht. Die wichtigste Norm war die Normreihe ISO/IEC/IEEE 29119-x:201x. Um zu gewährleisten, dass jedes BsM hinreichend und nach Stand der Technik abgesichert werden kann, wurden 31 weitere Normen und Gesetzte untersucht und hinsichtlich Testvorgaben interpretiert. Das Ergebnis ist ein modularer Baukasten, der Anwendung während der Testplanung agiler Projekte findet. Grundlage bilden die besonderen Merkmale eines Testobjektes. Mit jedem besonderen Merkmal sind verschiedene Testmodule verbunden, aus denen Testfälle abgeleitet werden. So kann eine norm- und gesetzeskonforme Absicherung erfolgen, da die Module und die Zuordnung zu den BsM auf der hermeneutischen Untersuchung basieren. Um dies möglichst effizient zu gestalten, wurde zusätzlich ein Vorschlag zur IT gestützten Testplanung mittels eines ALM-Tools gemacht.

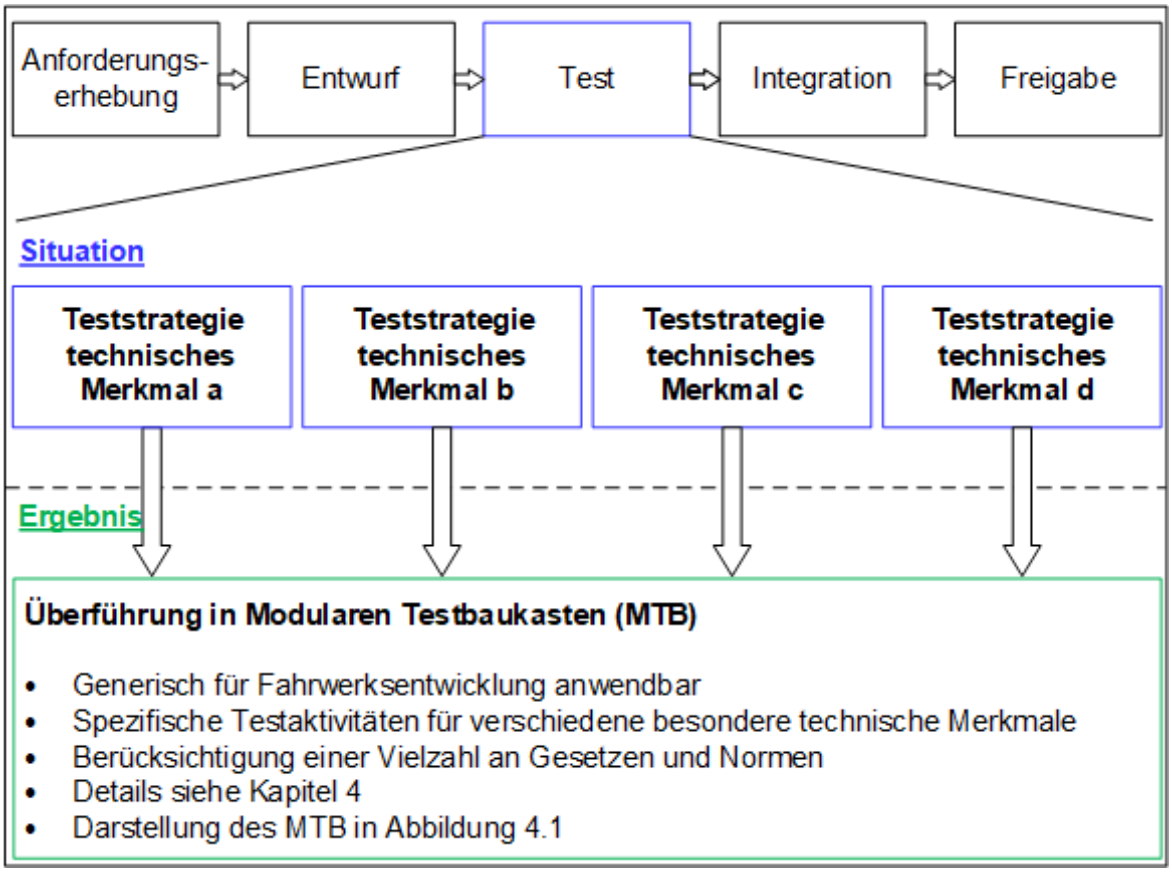

Abbildung 7.1: Zielerreichung durch den MTB

Für eine vollständige Absicherungs- und Freigabe-Methode sind noch weitere Aspekte zu berücksichtigen. Abbildung 7.2 stellt die Zusammenhänge zwischen der Testmethode, der Freigabe und dem Testprozess grafisch dar. Die SW-Testplanung beachtet neben den starren Produktentstehungsprozessen der Fahrzeugprojekte auch agile Zeitpläne. Gemeinsam mit dem MTB und weiteren Testartefakten werden diese für die Planung der Tests in einer frühen Projektphase herangezogen. Durch zeitliche Engpässe ist es jedoch möglich, dass nicht alle Testmodule durchgeführt werden können. Um diesen Zielkonflikt aufzulösen, wurde eine Methode zur effizienten Testfallpriorisierung vorgestellt. Insgesamt wurden sechs Kriterien gezeigt, anhand derer Testfälle priorisiert werden können. Dazu zählen: Gefahrenniveau, Ausfallrisiko, Qualitätsrisiko, Abdeckungsrate, Durchführungsdauer und Veränderungsrate des Testobjektes.

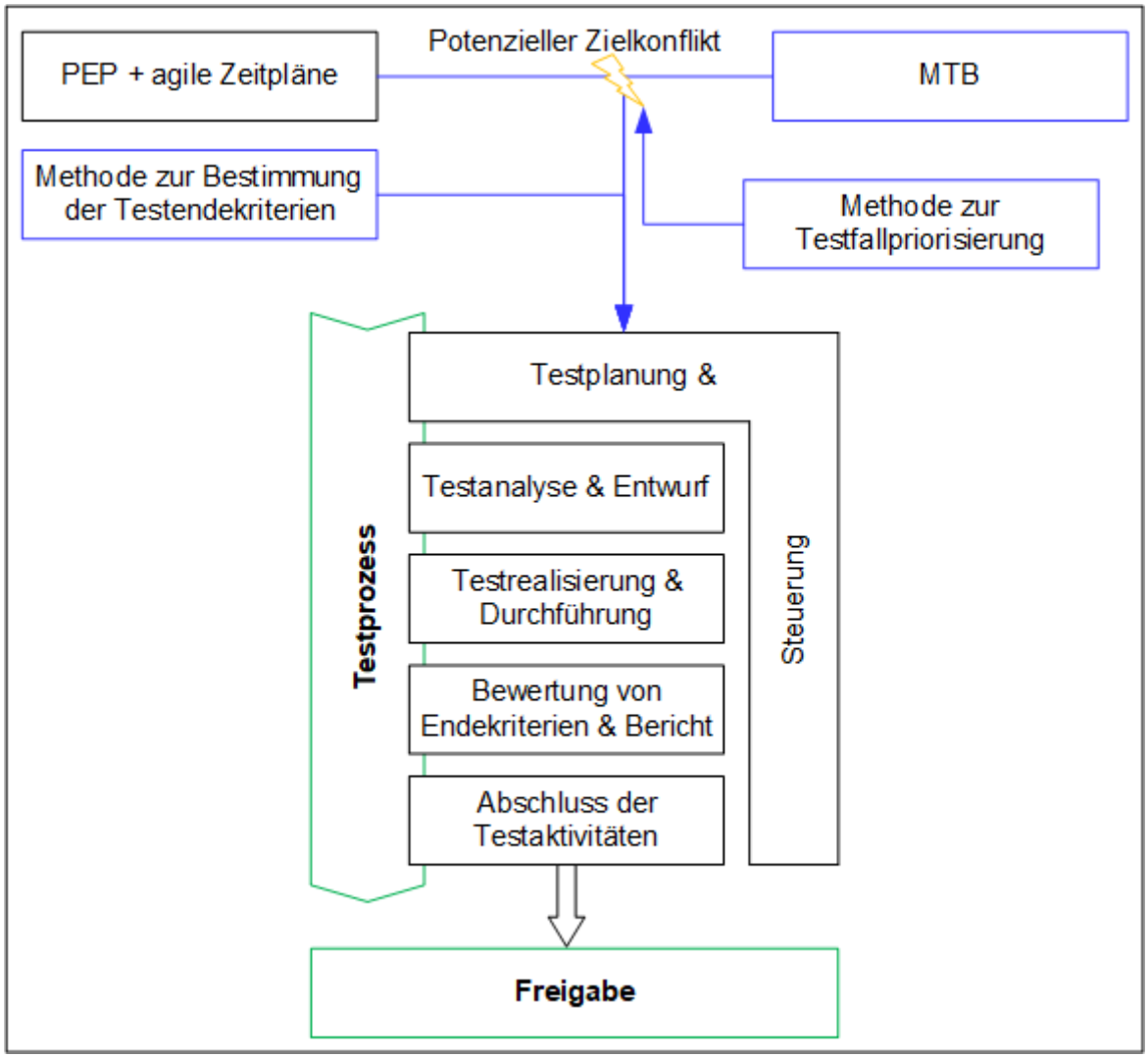

Abbildung 7.2: Zusammenfassung

Darüber hinaus wurde eine Freigabemethodik für die agile Softwareentwicklung vorgestellt. Die zielt darauf ab, dass Testendekriterien abhängig von besonderen Merkmalen und Freigabelevels bestimmt werden. Verschiedene Risiken, die im Zusammenhang mit der Freigabe stehen, wurden identifiziert, deren Auswirkungen gewichtet und anschließend deren Eintrittswahrscheinlichkeit in Abhängigkeit vom Freigabelevel bestimmt. Der Risikogewichtungsfaktor und die Eintrittswahrscheinlichkeit wurden genutzt, um einen Eintrittsfaktor zu berechnen. Für jedes BsM wurde anschließend ein Gefahrenniveau bestimmt. Dieses wird mit dem Eintrittsfaktor verrechnet, um ein levelspezifisches Testendekriterium für jedes besondere Merkmal zu erhalten. Die ganzheitliche Betrachtung von Test und Freigabe ermöglicht effiziente Abläufe in agilen Entwicklungsprojekten unter Berücksichtigung des PEP.

7.2 Wissenschaftlicher Beitrag

Testprozesse und Teststrategien sind unter SW-Testern etabliert und bilden das Fundament für die meisten Testprojekte. In der Fahrzeugentwicklung ist die Absicherung von

Funktionen zur Sicherstellung der Qualität und Einhaltung der funktionalen Sicherheit von besonderer Bedeutung. Neben diesen Aspekten werden noch weitere technische Merkmale abgesichert. Dabei werden all diese Merkmale mit unterschiedlichen Strategien und Vorgehensweisen getestet. In der Wissenschaft sind verschiedene Arbeiten, die die Absicherung einzelner Merkmale in den Fokus stellen, bekannt (siehe Abschnitt 1.2). Allerdings gibt es keine Arbeiten, die die Entwicklung einer einheitlichen Teststrategie für eine Domäne der Fahrzeugentwicklung in den Mittelpunkt der Betrachtung stellt.

In dieser Arbeit wurde solch ein Fokus gelegt und eine entsprechende Methode vorgestellt, um so den Stand der Technik und zur Wissenschaft zu erweitern. Bei einer ganzheitlichen Betrachtung der Arbeit wurde ein Verfahren zur Absicherung und Freigabe von Fahrwerkfunktionen in einem agilen Entwicklungsumfeld vorgestellt. Auch andere Arbeiten haben sich mit dem Ansatz der optimierten Testplanung auseinandergesetzt. Jedoch gibt es kein Vorgehen, das neben dem Test auch explizit auf die Freigabe eingeht. Dies wird häufig als Nebenprodukt betrachtet. Daher wurde hier eine Methode erarbeitet, die den Testprozess von der Testplanung bis zur Freigabe abdeckt. Dazu wurden verschiedene Ebenen berücksichtigt, was in vielen anderen Arbeiten ebenfalls vernachlässigt wird.

Konkret bedeutet dies, dass eine modulare Teststrategie vorgestellt wurde, die für verschiedene Ebenen verschiedene Testaktivitäten vorsieht. Diese sind jedoch nicht ausschließlich von Ebenen, sondern ebenso von besonderen technischen Merkmalen abhängig. Mit der korrekten Granularität der BsM wird eine einheitliche Vorgehensweise etabliert. Dazu wurden zunächst alle für die Fahrwerkentwicklung relevanten BsM identifiziert. Anschließend wurde diesen unter der Berücksichtigung des Standes der Technik und verschiedener Normen Testaktivitäten zugeordnet. Eine Testaktivität ist in einem Testmodul beschrieben und beinhaltet verschiedene Aspekte der Testplanung und Testfallerstellung. Dieser neuartige Ansatz ermöglicht es Testobjekten, mehrere besondere Merkmale zuzuordnen und toolgestützt das effizienteste Testvorgehen zu bestimmen.

Darüber hinaus wurde eine neuartige Freigabemethode entwickelt, die auf Risiken basiert, um Freigaben mit weniger kritischen Anforderungen effizient gestalten zu können. Dabei hilft sowohl ein Traceabilitykonzept, welches in ein ALM-Tool integriert wird, als auch Maßnahmen zur Priorisierung von Testfällen zur Freigabe von Entwicklungsartefakten. Der Aspekt der Freigabe wird in der Literatur meist gänzlich vernachlässigt. Daher wird der Stand der Wissenschaft um das hier vorgestellte Vorgehen erweitert. In der Praxis ist dieses ebenso neuartig, da agile Abläufe berücksichtigt werden. Agile Arbeitsweisen gelten in der Automobilentwicklung anders als in der reinen Softwareentwicklung als neuartig und sind daher von großer Bedeutung für diese Arbeit. Dieser ganzheitliche Ansatz von Test und Freigabemethoden wird in Zukunft noch entscheidender werden, da die steigende Komplexität im Automobilbau nur mit durchgängigen Verfahren und Methoden beherrscht werden kann.

Sich verändernde Strukturen in Fahrzeugen und der Fahrzeugentwicklung führt auch zu angepassten rechtlichen Rahmenbedingungen. Um den Veränderungen gerecht zu werden, wurden in dieser Arbeit 32 Normen und Gesetzte hinsichtlich ihrer Anforderungen an Testprozesse und -aktivitäten analysiert und interpretiert. Eine Interpretation in diesem Umfang ist in der bisherigen Literatur nicht anzufinden und stellt einen Beitrag zur Wissenschaft dar.

7.3 Ausblick

Der nächste logische Schritt ist ganzheitliche die Anwendung der Methode in der Praxis. Zur vollständigen Evaluierung sollte die entwickelte Methode in einem Fahrzeugentwicklungsprojekt von Beginn bis zum SOP angewendet werden. Solche Entwicklungsprojekte betragen zwischen sechs und sieben Jahren, weshalb dies im Rahmen der Arbeit nicht möglich war. Allerdings sind die Ergebnisse so dargestellt, dass sie in einem Projekt in dieser Form angewandt werden können. Dazu wurden ein ganzheitliches Konzept vorgestellt, welches genau aufzeigt, zu welchem Zeitpunkt welcher Aspekt der Methode zum Einsatz kommt und wie eine Toolunterstützung aussehen muss.

Allerdings muss erwähnt werden, dass die Ergebnisse dieser Forschungsarbeit eine Momentaufnahme darstellen. Die Entwicklung und das Testen von Software bringt immer neue Erkenntnisse, sodass eine fortlaufende Anpassung des MTB in der Praxis notwendig ist. Durch die Weiterentwicklung scheint die Wahrscheinlichkeit, dass sich die Relevanz besonderer Merkmale in der Fahrwerkentwicklung ändert, sehr hoch. Dies bedeutet eine stetige Anpassung, um dem Stand der Technik zu entsprechen. In Zukunft werden nicht nur neue Merkmale hinzukommen, sondern ebenso werden Merkmale an Bedeutung verlieren und dementsprechend gelöscht.

Zeitgleich verändern sich weltweit die Anforderungen an Fahrzeuge und dadurch auch rechtliche Rahmenbedingungen. Ein fortlaufendes Gesetzesmonitoring ist essenziell für den Erfolg einer solchen Methode. Nur so können Relevanz und Bedeutung von neuen Normen und Gesetzen bestimmt werden. Bei Bedarf müssen diese in den Baukasten aufgenommen und entsprechende Maßnahmen abgleitet werden. Zusätzlich sollten auch weitere Ergebnisse von Forschung und Entwicklung rund um das Softwaretesten berücksichtigt werden. Neue Erkenntnisse ermöglichen die Adaption des Baukastens hinsichtlich neuer Testarten, -verfahren oder -methoden. Damit kann eine Alterung vermieden und die Vorteile des Baukastens auch in Zukunft genutzt werden.

Die in dieser Arbeit erhobenen Ergebnisse bieten darüber hinaus Ansatzpunkte für weitere Forschungsarbeiten. So zielen die Ergebnisse auf die Entwicklung von Fahrwerkfunktionen ab. Mögliche Forschungen könnten sich auf die Adaption der Methode für Funktionen anderer Domänen, bspw. Antrieb oder Karosserie, beziehen. Da die Anforderungen auch in

diesen Bereichen für höhere Komplexität der Entwicklungsartefakte sorgen, ist eine Modularität zur Effizienzsteigerung sinnvoll. Weiterhin können auch einzelne Aspekte dieser Arbeit genauer untersucht werden. Die für die Testfallpriorisierung gezeigte Methode basiert auf wenigen Kriterien, um die Priorisierung so einfach wie möglich zu gestalten. Dies bietet weiteres Potenzial zur Forschung. Die meisten bisherigen Methoden sind entweder sehr komplex und zeitintensiv oder, wie in diesem Fall sehr einfach. Hier könnten folglich weitere Untersuchungen stattfinden, um eine präzisere, jedoch wenig zeitintensive Methode zu entwickeln.

Auch die Freigabe im agilen Entwicklungsumfeld ist erfolgversprechend für weitere Forschungsarbeiten. Da diese meist bereits etablierten und historisch gewachsenen Abläufen entspricht, können Forscher neue Methoden und Ansätze für effizientere Freigabeabläufe entwickeln. Für einen standardisierten Prozess für die Automobilindustrie müssen etablierte Vorgehensmodelle, z. B. Produktentstehungsprozesse als auch neue agile Methoden betrachtet werden, um eine effiziente Freigabe zu ermöglichen.

Globale Veränderungen wie der Klimawandel oder wirtschaftliche Aufstiege einzelner Länder sowie die Trends „Elektromobilität“ und „hochautomatisierte Fahrzeuge“ bewirken, dass Gesetzgebungen und Normen fortlaufend angepasst werden. Deshalb müssen auch Entwicklungsmethoden stets angepasst werden und die hier vorgestellten Forschungsergebnisse stellen einen momentanen Stand dar. In Zukunft müssen neue Teststrategien, Testmethoden und Testverfahren entwickelt werden, welche die steigende Komplexität, neue Technologien und neue bzw. angepasste Rechtsgrundlagen berücksichtigen. Nur so ist es möglich, auch in Zukunft sichere Fahrzeuge zu entwickeln, die den veränderten Anforderungen gerecht werden. Dieser Herausforderung müssen sich alle Fahrzeughersteller widmen, um konkurrenzfähig zu bleiben.

Anhang

A Testmodule

Im Folgenden werden die Kurzfassungen aller relevanten Testmodule aufgezeigt.

Funktionale Tests

Tabelle A.1: Anforderungsbasierter Test (funktional)

ID_001	**Anforderungsbasierter Test (funktional)**
Testebene	SWC & SWS
Testziel	Überprüfung, ob das Testobjekt die funktionalen Anforderungen erfüllt
Testmethoden	Analyse der funktionalen Anforderungen
Testendekriterium	Jede Anforderung durch mind. 1 Testfall abgedeckt

Tabelle A.2: Fault Injection Test

ID_002	**Fault Injection Test**
Testebene	SWC & SWS
Testziel	Analyse des Verhaltens und der Fehleridentifizierung eines Testobjekts durch absichtliche Fehleraufschaltungen
Testmethoden	Anforderungsanalyse Analyse möglicher Fehlerfälle
Testendekriterium	Jeder identifizierte mögliche Fehler durch mind. ein Testfall abgedeckt

Tabelle A.3: Error Guessing

ID_003	**Error Guessing**
Testebene	SWC & SWS
Testziel	Systematische Fehleraufdeckung durch Übertragung von Erfahrungen der Tester und Entwickler auf aktuelles Testprojekt
Testmethoden	Erfahrungswerte der Tester und Entwickler Analyse von Lessons Learned Dokumenten
Testendekriterium	-

Tabelle A.4: Testkategorie Äquivalenzklassen

ID_004	**Äquivalenzklassentest & Grenzwertanalyse**
Testebene	SWC & SWS
Testziel	Bestätigung, dass Eingangs- oder Ausgangswerte (oder eine Kombination) zu einer richtigen Funktionalität führen

Tabelle A.5: Äquivalenzklassentest

ID_004_1	**Äquivalenzklassentest**
Testziel	Einteilung der Werte in Äquivalenzklassen und Überprüfung, dass jede Äquivalenzklasse das erwartet Verhalten aufweist
Testmethoden	Anforderungsanalyse zur Ableitung von Äquivalenzklassen
Testendekriterium	Jede Äquivalenzklasse wird durch mind. einen Repräsentanten getestet

Tabelle A.6: Grenzwertanalyse

ID_004_2	**Grenzwertanalyse**
Testziel	Verwendung der Grenzwerte von Äquivalenzklassen zur Überprüfung des erwarteten Verhaltens an den Grenzen der Klassen
Testmethoden	Analyse der Grenzwerte
Testendekriterium	Pro Grenzwert werden folgende Werte getestet: Grenzwert; erster Wert außerhalb des Wertebereichs; erster Wert innerhalb des Wertebereichs

Tabelle A.7: Zustandsbasierter Test

ID_005	**Zustandsbasierter Test**
Testebene	SWC & SWS
Testziel	Nachweis, dass definierte Reihenfolgen von Eingabewerten (Zustandsübergänge) zu erwartetem Ergebnis führen.
Testmethoden	Analyse der funktionalen Abhängigkeiten
Testendekriterium	Jeder Zustandsübergang wird getestet durch aneinander schalten mehrerer Übergänge in einem Testfall (multiple transitions)

Tabelle A.8: Szenariotest

ID_006	**Szenariotest**
Testebene	SWS
Testziel	Fehleridentifizierung durch Verwendung von Szenarien bzw. Anwendungsfällen (Abfolge von Interaktionen zwischen dem Testobjekt und anderen Systemen)
Testmethoden	Analyse von Anforderungen, Grenzbedingungen, Reihenfolgen, Fehlerquellen, Umweltbedingungen, Anwendungsfällen und Events die Fehler hervorrufen
Testendekriterium	Mind. ein Testfall pro Szenario

Nicht-Funktionale Tests

Tabelle A.9: Vulnerability Scan

ID_007	**Vulnerability Scan**
Testebene	SWC & SWS
Testziel	Prüfung des Testobjektes auf das Vorhandensein bekannter Schwachstellen
Testmethoden	Automatisierter toolgestützter Scan
Testendekriterium	-

Tabelle A.10: Penetration Testing

ID_008	**Penetration Testing**
Testebene	SWS
Testziel	Identifizierung möglicher Sicherheitsschwachstellen durch Penetrierung des Testobjektes (Versuch des unerlaubten Zugriffs)
Testmethoden	Analyse möglicher Sicherheitsschwachstellen
Testendekriterium	Projektindividuell

Tabelle A.11: Fuzzy Testing

ID_009	**Fuzzy Testing**
Testebene	SWC & SWS
Testziel	Identifizierung möglicher Sicherheitsschwachstellen durch Eingabe zufälliger Daten in großen Mengen
Testmethoden	Automatisierte toolgestützte Methode zur Erstellung der Eingabewerte
Testendekriterium	-

Tabelle A.12: Statistischer Test

ID_010	**Statistischer Test**
Testebene	SWC & SWS
Testziel	Statistischer Nachweis der vollständigen Testabdeckung
Testmethoden	Automatisierte toolgestützte Methode zur Erstellung statistisch vollständiger Testvektoren
Testendekriterium	-

Tabelle A.13: Testkategorie RUT

ID_011	**Ressourcennutzungstests (RUT)**
Testebene	SWC
Testziel	Prüfung, ob Zeit-, Ressourcen oder Kapazitätsnutzung des Testobjektes und festgelegte Grenzwerte (Anforderungen) bei Ausführung nicht überschritten werden

Tabelle A.14: Performance Test

ID_011_1	**Performance Test**
Testziel	Prüfung während der Ausführung des Testobjektes.
Testmethoden	Analyse nicht-funktionaler Anforderungen
Testendekriterium	-

Tabelle A.15: Lasttest

ID_011_2	**Lasttest**
Testziel	Prüfung unter wechselnder Last (niedrige, zu erwartende und Spitzenlasten)
Testmethoden	Analyse nicht-funktionaler Anforderungen
Testendekriterium	-

Tabelle A.16: Stresstest

ID_011_3	**Stresstest**
Testziel	Prüfung an oder über den spezifizierten Lastgrenzen
Testmethoden	Analyse nicht-funktionaler Anforderungen
Testendekriterium	-

Tabelle A.17: Modelling Guideline Check

ID_012	**Modelling Guideline Check**
Testebene	SWC
Testziel	Statische Prüfung der Einhaltung der Richtlinien auf Modellebene (toolbasiert)
Testmethoden	Richtlinienanalyse
Testendekriterium	Alle Checks zur Einhaltung wurden durchgeführt

Tabelle A.18: Static Code Analysis

ID_013	**Static Code Analysis**
Testebene	SWC
Testziel	Statische Prüfung der Einhaltung der Richtlinien auf Codeebene (toolbasiert)
Testmethoden	Richtlinienanalyse
Testendekriterium	Alle Checks zur Einhaltung wurden durchgeführt

Tabelle A.19: Schnittstellentest

ID_014	**Schnittstellentest**
Testebene	SWC & SWS
Testziel	Nachweis, dass alle spezifizierten Schnittstellen Daten und Signale korrekt übertragen
Testmethoden	Analyse nicht-funktionaler Anforderungen Analyse der internen und externen Schnittstellen
Testendekriterium	Jede Schnittstelle durch mind. einen Testfall abgedeckt

Strukturbasierte Tests

Tabelle A.20: Anweisungstest

ID_015	**Anweisungstest**
Testebene	SWC
Testziel	Nachweis, dass die Struktur der SWC jede Anweisung ermöglicht und richtig befolgt wird
Testmethoden	Strukturanalyse
Testendekriterium	Für jede durchführbare Anweisung ist mind. ein Testfall vorhanden

Tabelle A.21: Testkategorie Entscheidungsausgänge

ID_016	**Entscheidungsausgänge**
Testebene	SWC
Testziel	Prüfung der Entscheidungsausgänge

Tabelle A.22: Zweigtest

ID_016_1	**Zweigtest**
Testziel	Prüfung der Entscheidungsausgänge, durch das Ausführung jedes Zweigs in einem Kontrollflussdiagramm
Testmethoden	Strukturanalyse
Testendekriterium	Jeder Zweig durch mind. einen Testfall abgedeckt

Tabelle A.23: Entscheidungstest

ID_016_2	**Entscheidungstest**
Testziel	Prüfung der Entscheidungsausgänge, durch Ausführung jeder möglichen Entscheidung (mit mindestens zwei verschiedenen Ausgängen) eines Knotens
Testmethoden	Strukturanalyse Analyse von Entscheidungspfaden
Testendekriterium	Jeder Entscheidungsausgang durch mind. einen Testfall abgedeckt

Tabelle A.24: Testkategorie Bedingungstests

ID_017	**Bedingungstests**
Testebene	SWC
Testziel	Atomare Bedingungen innerhalb des Testobjektes zu testen, um so Fehlerzustände zu identifizieren

Tabelle A.25: Bedingungstest

ID_017_1	Bedingungstest
Testziel	Test und Bewertung der Bedingungen ("wahr" oder "falsch"), die zu einer Entscheidung führen
Testmethoden	Strukturanalyse
Testendekriterium	Jede Kombination von Wahrheitswerten innerhalb der Entscheidung durch mind. einen Testfall abgedeckt

Tabelle A.26: Modifizierter Bedingungs-/Entscheidungstest

ID_017_2	**Modifizierter Bedingungs-/Entscheidungstest (MC/DC)**
Testziel	Test und Bewertung der Bedingungen innerhalb einer Entscheidung, die den Ausgang der Entscheidung beeinflussen können
Testmethoden	Strukturanalyse
Testendekriterium	Jede Bedingung, deren Wahrheitswerte unabhängig von anderen Bedingungen den Ausgang der Entscheidung beeinflussen kann, durch mind. einen Testfall abgedeckt

Tabelle A.27: Datenflusstest

ID_018	**Datenflusstest**
Testebene	SWC
Testziel	Test aller Variablen im Code unter Berücksichtigung der Definition der Variablen und ihre Verwendung
Testmethoden	Strukturanalyse
Testendekriterium	Projektindividuell

Reviews

Tabelle A.28: Testkategorie: Reviews

ID_019	**Reviews**
Testebene & Testobjekte	Anforderungen (SWC & SWS) Code (SWC) Modell (SWC) Testspezifikation (SWC & SWS)
Testziel	Statische Prüfung der Testobjekte

Tabelle A.29: Technisches Review

ID_019_1	**Technisches Review**
Testziel	Prüfung, ob Anforderungen an Testobjekt erfüllt und für den geplanten Einsatzzweck nutzbar sind
Testmethoden	Technisches Review
Testendekriterium	Testobjekt wurde vollständig untersucht

Tabelle A.30: Informelles Review

ID_019_2	**Informelles Review**
Testziel	Bestätigung der Korrektheit, Vollständigkeit und Qualität ohne formale Vorgaben nach dem Vier-Augen-Prinzip.
Testmethoden	Informelles Review
Testendekriterium	Testobjekt wurde vollständig untersucht

Tabelle A.31: Walk-Through

ID_019_3	**Walk-Through**
Testziel	Bestätigung der Korrektheit, Vollständigkeit und Qualität durch ein strukturiertes Vorgehen bei dem der Autor die Reviewaktivitäten leitet
Testmethoden	Walk-Through
Testendekriterium	Testobjekt wurde vollständig untersucht

Tabelle A.32: Inspection

ID_019_4	**Inspection**
Testziel	Bestätigung der Korrektheit, Vollständigkeit und Qualität durch ein stark strukturiertes Vorgehen, bei dem ein unabhängiger Reviewleiter die Reviewaktivitäten koordiniert und leitet
Testmethoden	Informelles Review
Testendekriterium	Testobjekt wurde vollständig untersucht

B Relevante Regularien

Diese Übersicht zeigt alle für die Fahrwerkentwicklung relevanten Gesetze und Normen für den Testbaukasten der Fahrwerkentwicklung entsprechend der BsM aus Abschnitt 4.2.

Tabelle A.33: Übersicht relevanter Regularien für den modularen Testbaukasten

Nummer bzw. Titel	Inhalt und Umfang	Besonderes Merkmal
Automotive SPICE	Entwicklungsprozesse in der Automobilbranche	Alle, insb. Qualitätssicherung
AUTOSAR CP R20-11	SG und SW Entwicklung in der Automobilbranche	Alle
FINAL REGULATION ORDER § 1968.2 of title 13 CCR	Anforderungen an OBD Systeme	On Board Diagnosis
GB 201-5	Software Updates für Fahrzeuge	Remote Updates
IEEE Std 1028-2008	Software Reviews	Alle
ISO/DIS 24089:2022	Software Updates für Fahrzeuge	Remote Updates
ISO/IEC 20246:2017	Reviews von Arbeitsergebnissen	Alle
ISO/PAS 21448:2019	Sicherheit der Sollfunktion von Fahrzeugen	SOTIF
ISO/SAE 21434:2021	Informationssicherheit von Fahrzeugen	Cybersecurity, Connected Functions
NHTSA	Informationssicherheit von Fahrzeugen	Cybersecurity, Remote Updates, Connected Functions
Normreihe ISO 15031-x:2015	Anforderungen an OBD Systeme	On Board Diagnosis
Normreihe ISO 26262-x:201x	Funktionale Sicherheit von elektrischen Systemen in Fahrzeugen	Funktionale Sicherheit ASIL A – ASIL D
ISO/IEC 15408-3:2008; Normreihe ISO/IEC 15408-x:200x	Informationssicherheit von IT Systemen	Cybersecurity, Connected Functions
Normreihe ISO/IEC/IEEE 29119-x:201x	Software Testing	Alle, insb. Qualitätssicherung
UN Regelung Nr. 0	Zulassung von Fahrzeugen	Alle

Nummer bzw. Titel	Inhalt und Umfang	Besonderes Merkmal
UN Regelung Nr. 13	Zulassung der Bremsanlage von Fahrzeugen	Alle, insb. SOTIF
UN Regelung Nr. 13 H	Zulassung der Bremsanlage von Fahrzeugen	Alle, insb. SOTIF
UN Regelung Nr. 49	Maßnahmen gegen die Emission von Schadstoffen	Tailpipe, Leistungsdaten, Fahrwiderstände
UN Regelung Nr. 79	Zulassung der Lenkanlage von Fahrzeugen	Alle, insb. SOTIF
UN Regelung Nr. 83	Zulassung von Fahrzeugen hinsichtlich Schadstoffemission des Motors	Tailpipe, Leistungsdaten, Fahrwiderstände
UN Regelung Nr. 140	Zulassung von elektronischen Fahrdynamik-Regelsystemen	Alle, insb. Funktionale Sicherheit & SOTIF
UN Regelung Nr. 141	Zulassung von Reifendruckkontrollsystemen	Alle, insb. Funktionale Sicherheit, SOTIF & Fahrwiderstände
UN Regelung Nr. 154	Worldwide harmonized Light vehicles Test Procedure zur Bestimmung der Abgasemissionen	Tailpipe, Leistungsdaten, Fahrwiderstände
UN Regelung Nr. 155	Informationssicherheit von Fahrzeugen	Cybersecurity, Remote Updates, Connected Functions
UN Regelung Nr. 156	Software Updates für Fahrzeuge	Remote Updates
VERORDNUNG (EC) Nr. 715/2007	Zulassung von Fahrzeugen hinsichtlich Emissionen	Tailpipe, Leistungsdaten, Fahrwiderstände
VERORDNUNG (EU) 2017/1151	Zulassung von Fahrzeugen hinsichtlich Emissionen	Tailpipe, Leistungsdaten, Fahrwiderstände
VERORDNUNG (EU) 2017/1154	Zulassung von Fahrzeugen hinsichtlich Emissionen	Tailpipe, Leistungsdaten, Fahrwiderstände
VERORDNUNG (EU) 2017/1347	Zulassung von Fahrzeugen hinsichtlich Emissionen	Tailpipe, Leistungsdaten, Fahrwiderstände
VERORDNUNG (EU) 2018/1832	Optimierung der Typgenehmigungsprüfung hinsichtlich Emissionen	Tailpipe, Leistungsdaten, Fahrwiderstände
VERORDNUNG (EU) 2018/858	Zulassung von Fahrzeugen und Fahrzeugsystemen	Alle

Nummer bzw. Titel	Inhalt und Umfang	Besonderes Merkmal
VERORDNUNG (EU) 2019/2144	Zulassung von Fahrzeugen hinsichtlich allgemeiner Sicherheit	Alle

Literaturverzeichnis

VDI Richtlinie VDI 2206, 06.2004: 2206 Entwicklungsmethodik für mechatronische Systeme.

70/220/EWG, 20.03.1970: Angleichung der Rechtsvorschriften der Mitgliedstaaten über Maßnahmen gegen die Verunreinigung der Luft durch Abgase von Kraftfahrzeugmotoren mit Fremdzündung.

ATB; GTB; STB (2011): Certified Tester Foundation Level Syllabus. Deutschsprachige Ausgabe. Hg. v. Austrian Testing Board (ATB), German Testing Board (GTB) e.V. & Swiss Testing Board (STB). International Software Testing Qualifications Board, zuletzt aktualisiert am 2017.

ATB; GTB; STB (2020): Lehrplan Certified Tester Foundation Level. Deutschsprachige Ausgabe. Hg. v. Austrian Testing Board (ATB), German Testing Board (GTB) e.V. & Swiss Testing Board (STB). International Software Testing Qualifications Board. Online verfügbar unter https://www.german-testing-board.info/wp-content/uploads/2022/01/GTB-CTFL_Lehrplan_v3.1_DE.pdf, zuletzt aktualisiert am 2020.

AutoBild (2022a): Alle Infos zur Generation: Renault ZOE. Hg. v. Axel Springer SE. Online verfügbar unter https://www.autobild.de/marken-modelle/renault/zoe/1/, zuletzt geprüft am 23.08.2022.

AutoBild (2022b): Alle Infos zur Generation: Tesla Model S. Hg. v. Axel Springer SE. Online verfügbar unter https://www.autobild.de/marken-modelle/tesla/model-s/1/, zuletzt geprüft am 23.08.2022.

AUTOSAR (2022a): ADAPTIVE PLATFORM. AUTOSAR. Online verfügbar unter https://www.autosar.org/standards/adaptive-platform/, zuletzt aktualisiert am 2022, zuletzt geprüft am 01.08.2022.

AUTOSAR (2022b): CLASSIC PLATFORM. AUTOSAR. Online verfügbar unter https://www.autosar.org/standards/classic-platform/, zuletzt aktualisiert am 2022, zuletzt geprüft am 01.08.2022.

Beck, Kent; Beedle, Mike; van Bennekum, Arie; Cockburn, Alistair; Cunningham, Ward; Fowler, Martin et al. (2001): Manifesto for Agile Software Development. Online verfügbar unter http://agilemanifesto.org/iso/de/manifesto.html, zuletzt aktualisiert am 13.02.2001, zuletzt geprüft am 02.08.2022.

Böhm, Janko (2019): Erfolgsfaktor Agilität. Warum Scrum und Kanban zu zufriedenen Warum Scrum und Kanban zu zufriedenen Mitarbeitern und erfolgreichen Kunden führen. Wiesbaden: Springer Vieweg.

Bohnet, Johannes (2015): Wie Alltags-Software Menschenleben gefährden kann. Komplexe Programme überfordern Autobauer. Hg. v. manager magazin new media GmbH & Co. KG. manager-magazin.de. Online verfügbar unter https://www.manager-magazin.de/digitales/it/automobilbranche-leidet-unter-software-komplexitaet-a-1049852.html, zuletzt aktualisiert am 31.08.2015, zuletzt geprüft am 02.08.2022.

Buiga, Andrei (2012): Investigating the Role of MQB Platform in Volkswagen Group's Strategy and Automobile Industry. In: *International Journal of Academic Research in Business and Social Sciences* (Vol. 2, No. 9), S. 391–399.

Burkacky, Ondrej; Deichmann, Johannes; Klein, Benjamin; Pototzky, Klaus; Scherf, Gundbert (2020): Cybersecurity in automotive. Mastering the challenge. Hg. v. McKinsey & Company. Online verfügbar unter https://www.mckinsey.com/~/media/mckinsey/industries/automotive%20and%20assembly/our%20insights/cybersecurity%20in%20automotive%20mastering%20the%20challenge/cybersecurity-in-automotive-mastering-the-challenge.pdf, zuletzt aktualisiert am März 2020, zuletzt geprüft am 08.08.2022.

Center of Automotive Management (CAM) (2021): Rückruf-Trends der globalen Automobilhersteller im Langfristvergleich (2011-2021) Referenzmarkt USA. AutomotivePerformace Studie. Hg. v. Stefan Bratzel. Center of Automotive Managment. Bergisch Gladbach. Online verfügbar unter https://auto-institut.de/automotiveperformance/rueckruf-trends-der-globalen-automobilhersteller-im-langfristvergleich-2011-2021-referenzmarkt-usa/, zuletzt aktualisiert am 19.10.2021, zuletzt geprüft am 02.08.2022.

AUTOSAR CP R20-11, 30.11.2020: Classic Platfrom: Requirements on Firmware Over-The-Air.

Daigl, Matthias; Glunz, Rolf (2016): ISO 29119 - Die Softwaretest-Normen verstehen und anwenden. Heidelberg: dpunkt.verlag GmbH.

Delos Santos, Jose Maria (2022): Best ALM Tools and Software. Hg. v. Project-Management.com. TechnologyAdvice. Online verfügbar unter https://project-management.com/alm-tools/, zuletzt aktualisiert am 01.03.2022, zuletzt geprüft am 08.08.2022.

DIN 6789-5 : 1995-10, 10.1995: Dokumentationssystematik - Teil 5: Freigabe in der Technischen Produktdokumentation.

DIN 6789:2013-10, 10.2013: Dokumentationssystematik - Verfälschungssicherheit und Qualitätskriterien für die Freigabe digitaler Produktdaten.

DIN EN 82045-1:2001, 01.11.2002: Dokumentenmanagement - Prinzipien und Methoden.

Doll, Nikolaus (2018): Das Märchen von der Überlegenheit deutscher Autos. Hg. v. Axel Springer SE. WELT. Online verfügbar unter https://www.welt.de/wirtschaft/article174717353/Rueckruf-Statistik-Die-Ueberlegenheit-deutscher-Autos-erweist-sich-als-Maerchen.html, zuletzt aktualisiert am 20.03.2018, zuletzt geprüft am 02.08.2022.

Douglass, Bruce Powel (2016): Agile Systems Engineering. Waltham: Morgan Kaufmann.

Dralle, Jens (2020): Fast planmäßig mit fast allen Features. Produktionsstart VW ID.3. Hg. v. Motor Presse Stuttgart GmbH & Co.KG. Auto Motor und Sport. Stuttgart. Online verfügbar unter https://www.auto-motor-und-sport.de/elektroauto/vw-id3-auslieferung-september-2020/, zuletzt aktualisiert am 10.06.2020, zuletzt geprüft am 02.08.2022.

Dyba, Tore; Dingsoyr, Torgeir (2009): What Do We Know about Agile Software Development? In: *IEEE Software* (September/October 2009), S. 6–9.

UN Regelung Nr. 79, 16.10.2018: Einheitliche Bedingungen für die Genehmigung der Fahrzeuge hinsichtlich der Lenkanlage.

UN Regelung Nr. 155, 22.01.2021: Einheitliche Bedingungen für die Genehmigung von Fahrzeugen hinsichtlich der Cybersicherheit und des Cybersicherheitsmanagementsystems.

UN Regelung Nr. 156, 22.01.2021: Einheitliche Bedingungen für die Genehmigung von Fahrzeugen hinsichtlich der Softwareaktualisierung und des Softwareaktualisierungsmanagementsystems.

UN Regelung Nr. 141, 22.01.2017: Einheitliche Bedingungen für die Genehmigung von Kraftfahrzeugen hinsichtlich ihrer Reifendruckkontrollsysteme (RDKS).

UN Regelung Nr. 140, 29.12.2018: Einheitliche Bedingungen für die Genehmigung von Personenkraftwagen hinsichtlich der elektronischen Fahrdynamik-Regelsysteme (ESC-Systeme).

UN Regelung Nr. 13, 08.10.2015: Einheitliche Vorschriften für die Typgenehmigung von Fahrzeugen der Klassen M, N, und O hinsichtlich der Bremsen.

UN Regelung Nr. 13 H, 15.06.2015: Einheitliche Vorschriften für die Typgenehmigung von Personenkraftwagen hinsichtlich der Bremsen.

Konzernnorm VW 80303: 2014-06, 06.2014: Elektrische Eigenschaften und elektrische Sicherheit von Hochvolt-Komponenten in Kraftfahrzeugen.

Ersoy, Metin; Elbers, Christoph; Schick, Bernhard (2017): Fahrwerkentwicklung. In: Metin Ersoy und Stefan Gies (Hg.): Fahrwerkhandbuch. Wiesbaden: Springer Fachmedien Wiesbaden, S. 245–300.

Ersoy, Metin; Gies, Stefan (Hg.) (2017): Fahrwerkhandbuch. 5. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden.

Ersoy, Metin; Vogel, Volker (2017): Bestandteile des Fahrwerks. In: Metin Ersoy und Stefan Gies (Hg.): Fahrwerkhandbuch. 5. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden, S. 301–317.

Ettelbrück, Bernd (2021): Agilität im Management. In: Mario A. Pfannstiel, Werner Siedl und Peter F.-J. Steinhoff (Hg.): Agilität in Unternehmen. Wiesbaden: Springer Gabler, S. 51–81.

GeeksforGeeks (Hg.) (2020): Test Case Prioritization in Software Testing. Online verfügbar unter https://www.geeksforgeeks.org/test-case-prioritization-in-software-testing/, zuletzt aktualisiert am 11.08.2020, zuletzt geprüft am 08.08.2022.

GB 201-5, Entwurf vom 17.06.2021: General Technical Requirements for Software Updates of Vehicle.

Gerhardt, Thomas; Of-Allinger, Andreas; Harloff, Thomas (2022): Überraschende Fakten zum Urvater der Plattformen. Alles zum MQB von VW. Hg. v. Motor Presse Stuttgart GmbH & Co.KG. Auto Motor und Sport. Stuttgart. Online verfügbar unter https://www.auto-motor-und-sport.de/tech-zukunft/modularer-querbaukasten-von-vw-zehn-antworten-zum-mqb/, zuletzt aktualisiert am 24.06.2022, zuletzt geprüft am 07.08.2022.

Bundesgesetzblatt Teil II 1977 Nr.39, 11.10.1977: Gesetz zu den Übereinkommen vom 8. Novmeber 1968 über den Straßenverkehr und über Straßenverkehrszeichen, zu den Europäischen Zusatzübereinkommen vom 1. Mai 1971 zu diesen Übereinkommen sowie zum Protokoll vom 1. März 1973 über Straßenmarkierungen.

Grande, Marcus (2011): 100 Minuten für Anforderungsmanagement. Kompaktes Wissen nicht nur für Projektleiter und Entwickler. 1. Aufl. Wiesbaden: Vieweg + Teubner Verlag (Praxis).

Hatton, Les; Spinellis, Diomidis; van Genuchten, Michiel (2017): The long-term growth rate of evolving software: Empirical results and implications. Software Growth Rate. In: *Journal of Software: Evolution and Process* 2017 (Volume 29, Issue 5).

Hoffmann, Dirk W. (2013): Software-Qualität. 2. Aufl. Berlin, Heidelberg: Springer Vieweg.

IEEE Std 1028-2008, 15.08.2008: IEEE Standard for Software Reviews and Audits.

Normreihe ISO/IEC 330xx:2015, 01.03.2015: Information technology - Process assessment.

Normreihe ISO/IEC 15408-x:200x, 15.01.2014: Information technology - Security techniques - Evaluation criteria for IT security.

ISO/IEC 15408-3:2008, 06.01.2011: Information technology - Security techniques - Evaluation criteria for IT security - Part 3: Security assurance components.

Isermann, Rolf (2008): Mechatronische Systeme. Grundlagen. 2. Aufl. Berlin, Heidelberg: Springer Berlin Heidelberg.

ISTQB (2021): ISTQB Glossary. Hg. v. Matthias Hamburg und Armin Born. International Software Testing Qualifications Board. Online verfügbar unter https://glossary.istqb.org/de/search/, zuletzt aktualisiert am 30.06.2021, zuletzt geprüft am 02.08.2022.

ISTQB (2022): ISTQB - Who We Are. Hg. v. International Software Testing Qualifications Board. Online verfügbar unter https://www.istqb.org/about-us/who-we-are, zuletzt aktualisiert am 2022, zuletzt geprüft am 02.08.2022.

Jooß, Benedikt; Schramm, Dieter (2022): Komplexität von Softwaretests in der Fahrwerksentwicklung. In: *Automobiltechnische Zeitschrift* 2022 (04), S. 44–47.

Jooß, Benedikt; Schuld, Julian; Enderle, Matthias; Schramm, Dieter (2022): Testing of OTA-enabled functions in electronic control unit development. Hg. v. Hans-Christian Reuss. Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart. Stuttgart (9. AutoTest Fachkonferenz). Online verfügbar unter https://www.fkfs-veranstaltungen.de/veranstaltungen/autotest/bilder-der-veranstaltung/manuskripte, zuletzt geprüft am 22.10.2022.

Kiefner, Dominique Xavier (2014): Dynamisches und risikobasiertes Fahrwerksverbund-Testverfahren. Dissertation. Universität Stuttgart, Stuttgart. Institut für Verbrennungsmotoren und Kraftfahrwesen.

VDI Richtlinie VDI2225 Blatt 3, 11.1998: Konstruktrionsmethodik Technisch-wirtschaftliches Konstruieren - Technisch-wirtschaftliche Bewertung.

Kords, Martin (2022a): Anteil der Elektroautos am Bestand der Personenkraftwagen in Deutschland von 2012 bis 2022. Marktanteil der Elektroautos am Pkw-Bestand in Deutschland bis 2022. Hg. v. KBA und Statista. Online verfügbar unter https://de.statista.com/statistik/daten/studie/784986/umfrage/marktanteil-von-elektrofahrzeugen-in-deutschland/, zuletzt aktualisiert am 03.2022, zuletzt geprüft am 16.08.2022.

Kords, Martin (2022b): Anzahl der Neuzulassungen von Elektroautos in Deutschland von 2003 bis 2022. Neuzulassungen von Elektroautos in Deutschland bis 2022. Hg. v. KBA und Statista. Online verfügbar unter https://de.statista.com/statistik/daten/studie/244000/umfrage/neuzulassungen-von-elektroautos-in-deutschland/, zuletzt aktualisiert am 08.2022, zuletzt geprüft am 16.08.2022.

Krause, Dieter; Gebhardt, Nicolas (2018): Methodische Entwicklung modularer Produktfamilien. Hohe Produktvielfalt beherrschbar entwickeln. Unter Mitarbeit von Tammo Bahns, Erik Greve, Jennifer Hackl, Henry Jonas, Sebastian Ripperda, Olga Sankowski et al. Berlin, Heidelberg: Springer Vieweg.

Krause, Frank-Lothar; Franke, Hans-Joachim; Gausemeier, Jürgen (2007): Innovationspotenziale in der Produktentwicklung. 1. Aufl. München: Hanser.

Final Regulation Order §1968.2. of title 13, 25.06.2016: Malfunction and Diagnostic System Requirements--2004 and Subsequent Model-Year Passenger Cars, Light-Duty Trucks, and Medium-Duty Vehicles and Engines.

VDI Richtlinie VDI 2221, 05.1993: Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte.

VDI Richtlinie VDI 2223, 2004: Methodisches Entwerfen technischer Produkte.

Meyer, Bertrand (2014): Agile! The Good, the Hype and the Ugly. Cham: Springer International Publishing.

NHTSA (2020): Cybersecurity Best Practices for the Safety of Modern Vehicles. Hg. v. U.S. Department of Transportation - National Highway Traffic Safety Administration. Online verfügbar unter https://www.nhtsa.gov/sites/nhtsa.gov/files/documents/vehicle_cybersecurity_best_practices_01072021.pdf, zuletzt aktualisiert am 2020, zuletzt geprüft am 10.08.2022.

Ovesen, Nis (2012): The Challenges of becoming Agile. Implementig and conducting scrum in integrated product development. PhD Thesis. Aalborg Universitet, Aalborg. Department of Architecture, Design & Media Technology.

Pauly, Axel; Gruber, Steffen; Sendler, Jan; Volk, Heiner; Seethaler, Ludwig; Sattler, Michael et al. (2021): Fahrwerk. In: Stefan Pischinger und Ulrich Seiffert (Hg.): Vieweg Handbuch Kraftfahrzeugtechnik. 9. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden, S. 261–459.

Pischinger, Stefan; Seiffert, Ulrich (Hg.) (2021): Vieweg Handbuch Kraftfahrzeugtechnik. 9. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden.

Prefi, Thomas (2014): Qualitätsmanagement in der Produktentwicklung. In: Tilo Pfeifer und Robert Schmitt (Hg.): Masing Handbuch Qualitätsmanagement. 6. Aufl. München, Wien: Carl Hanser Verlag, S. 401–440.

DIN 69901-2:2009-01, 01.2009: Projektmanagement - Projektmanagementsysteme - Teil 2: Prozesse: Prozessmodell.

DIN 69901-5:2009-01, 01.2009: Projektmanagement - projektmanagementsysteme - Teil 5: Begriffe.

Quoc Tuan Le, Hoang; Friedl, Natalie; Ramsbrock, Jens; Martinus; Marcus (2009): Fokussierung auf den Testprozess. Methoden zur Testfall-Priorisierung. In: *ATZelektronik* 2009 (4), S. 42–48. Online verfügbar unter https://doi.org/10.1007/BF03223962, zuletzt geprüft am 08.08.2022.

Renner, Ingo (2007): Methodische Unterstützung funktionsorientierter Baukastenentwicklung am Beispiel Automobil. Dissertation. Technische Universität München, München. Lehrstuhl für Produktentwicklung.

Ring, Martin (2019): Systematische Security-Tests von Kraftfahrzeugen. Dissertation. Universität Ulm, Ulm. Fakultät für Ingenieurwissenschaften.

ISO/SAE 21434:2021, 08.2021: Road vehicles - Cybersecurity engineering.

Normreihe ISO 26262-x:201x, 12.2018: Road vehicles - Functional safety.

ISO 26262-3:2018, 12.2018: Road vehicles - Functional safety - Part 3: Concept phase.

ISO 26262-6:2018, 12.2018: Road vehicles - Functional safety - Part 6: Product development at the software level.

ISO 26262-8:2018, 12.2018: Road vehicles - Functional safety - Part 8: Supporting processes.

ISO/PAS 21448:2019, 01.2019: Road vehicles - Safety of the intended functionality.

ISO/DIS 24089:2022, Entwurf vom 11.01.2022: Road vehicles — Software update engineering.

Rothermel, Gregg; Untch, Roland H.; Chu, Chengyun; Harrold, Mary Jean (1999): Test case prioritization: an empirical study. In: Proceedings IEEE International Conference on Software Maintenance. Software Maintenance for Business Change. Proceedings IEEE International Conference on Software Maintenance. Oxford, UK, 03.09.1999 - 03.09.1999: IEEE, S. 179–188.

Sattler, Kathrin (2015): Methodik für den Systemtest in der integralen Fahrzeugsicherheit. Dissertation. Otto-von-Guericke-Universität Magdeburg, Magdeburg. Fakultät für Elektrotechnik und Informationstechnik.

Scaled Agile, Inc. (Hg.) (2021): Agile Architecture in SAFe. Online verfügbar unter https://www.scaledagileframework.com/agile-architecture/, zuletzt aktualisiert am 10.02.2021, zuletzt geprüft am 08.08.2022.

Schäppi, Bernd; Andreasen, Mogens M.; Kirchgeorg, Manfred; Radermacher, Franz Josef (2005): Handbuch Produktentwicklung. München: Carl Hanser Verlag GmbH & Co. KG.

Schäuffele, Jörg; Zurawka, Thomas (2013): Automotive Software Engineering. 5. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden.

Schmidt, Christoph (2016): Agile Software Development Teams. The Impact of Agile Development on Team Performance. Cham, Heidelberg, New York, Dordrecht, London: Springer International Publishing.

Scholz, Gerd (2006): Im Fokus: Volkswagen. Kernkompetenz: Sparen. In: *Automobil-Produktion* 2006 (März 2006), S. 14–15. Online verfügbar unter https://web.archive.org/web/20110718200927/http://www.automobil-produktion.de/imperia/md/content/ap/heftausgaben/ap2006/ap03-2006/14_15_ap.pdf, zuletzt geprüft am 08.08.2022.

Schrof, Julian Immanuel; Paetzold, Kristin (2019): Product modularization requirements in agile automotive product development. In: Dieter Krause, Kristin Paetzold und Sandro Wartzack (Hg.): DFX 2019: Proceedings of the 30th Symposium Design for X, 18-19 September 2019, Jesteburg, Germany. 30th Symposium Design for X. Jesteburg, 18. -19. September 2019: The Design Society.

Schwaber, Ken; Sutherland, Jeff (2020): The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game. Hg. v. Ken Schwaber und Jeff Sutherland. Online verfügbar unter https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf#zoom=100, zuletzt aktualisiert am November 2020, zuletzt geprüft am 02.08.2022.

Sekitoleko, Nelson; Evbota, Felix; Knauss, Eric; Sandberg, Anna; Chaudron, Michel; Olsson, Helena Holmström (2014): Technical Dependency Challenges in Large-Scale Agile Software Development. In: Wil van der Aalst, John Mylopoulos, Michael Rosemann, Michael J. Shaw, Clemens Szyperski, Giovanni Cantone und Michele Marchesi (Hg.): Agile Processes in Software Engineering and Extreme Programming, Bd. 179. 15th International Conference, XP 2014. Rom, 26. - 30. Mai 2014. Cham: Springer International Publishing (Lecture Notes in Business Information Processing), S. 46–61.

Simon, Yvonne (2019): 100 Millionen Codezeilen: Das kann der Golf 8. kfz-betrieb. Online verfügbar unter https://www.kfz-betrieb.vogel.de/100-millionen-codezeilen-das-kann-der-golf-8-a-824084/, zuletzt aktualisiert am 26.04.2019, zuletzt geprüft am 01.08.2022.

Normreihe ISO/IEC/IEEE 29119-x:201x, 01.09.2013: Software and systems engineering - Software testing.

ISO/IEC/IEEE 29119-1:2013, 01.09.2013: Software and systems engineering - Software testing - Part 1: Concepts and definitions.

ISO/IEC/IEEE 29119-2:2013, 01.09.2013: Software and systems engineering - Software testing - Part 2: Test processes.

ISO/IEC/IEEE 29119-3:2013, 01.09.2013: Software and systems engineering - Software testing - Part 3: Test documentation.

ISO/IEC/IEEE 29119-4:2015, 01.12.2015: Software and systems engineering - Software testing - Part 4: Test techniques.

ISO/IEC/IEEE 29119-5:2016, 15.11.2016: Software and systems engineering - Software testing - Part 5: Keyword-Driven Testing.

ISO/IEC 20246:2017, 02.2017: Software and systems engineering - Work product reviews.

Specht, Thorsten (2021): Testen im agilen Kontext: habe ich genug getestet – wann höre ich auf? Hg. v. Jakob Robert Jaworski. Testautomatisierung Gewusst Wie. Armsheim. Online verfügbar unter https://testautomatisierung-gewusst-wie.de/testen-im-agilen-kontext-wann-habe-ich-genug-getestet-wann-hoere-ich-auf/, zuletzt aktualisiert am 16.04.2021, zuletzt geprüft am 08.08.2022.

Spillner, Andreas; Linz, Tilo (2019): Basiswissen Softwaretest. Aus- und Weiterbildung zum Certified Tester : Foundation Level nach ISTQB-Standard. 6., überarbeitete und aktualisierte Auflage. Heidelberg: dpunkt.verlag.

ISO/IEC/IEEE 12207:2017, 11.2017: Systems and software engineering - Softwarelife cycle processes.

Takeuchi, Hirotaka; Nonaka, Ikujiro (1986): The New New Product Development Game. In: *Harvard Buisness Review* 1986 (January 1986), S. 137–146.

TechTarget, Inc. (Hg.) (2015): Application Lifecycle Management (ALM). Definition. ComputerWeekly.de. Online verfügbar unter https://www.computerweekly.com/de/definition/Application-Lifecycle-Management-ALM, zuletzt aktualisiert am 12.2015, zuletzt geprüft am 08.08.2022.

Thiel, Sebastian (2012): Petri-Netz basierte Verifikation von funktionalen Testfällen. Dissertation. Universität Koblenz-Landau, Koblenz. Fachbereichs 4: Informatik.

Tischer, Mirko (2018): Das Rechenzentrum im Fahrzeug. AUTOSAR Adaptive. In: *Elektronik automotive* 2018 (Sonderausgabe Bordnetz 2018), S. 30–33.

VERORDNUNG (EG) Nr. 715/2007, 20.06.2007: Typgenehmigung von Kraftfahrzeugen hinsichtlich der Emissionen von leichten Personenkraftwagen und Nutzfahrzeugen (Euro 5 und Euro 6) und über den Zugang zu Reparatur- und Wartungsinformationen für Fahrzeuge.

VDA (2022): Automotive SPICE. Hg. v. Verband der Automobilindustrie e. V. (VDA). Qualitäts Management Center (QMC). Online verfügbar unter https://vdaqmc.de/software-prozesse/automotive-spice/, zuletzt geprüft am 03.08.2022.

VDA QMC Working Group 13 / Automotive SIG (2017): Automotive SPICE. Process Reference Model Process Assessment Model. Hg. v. Verband der Automobilindustrie e. V. (VDA). Online verfügbar unter https://www.automotivespice.com/fileadmin/software-download/AutomotiveSPICE_PAM_31.pdf, zuletzt aktualisiert am 01.11.2017, zuletzt geprüft am 03.08.2022.

Verband der Automobilindustrie e. V. (VDA) (2020): Prozessbeschreibung Besondere Merkmale (BM). 2. Aufl. Frankfurt am Main: Henrich Druck + Medien GmbH.

Widmann, Ulrich; Weissinger, Jürgen; Breitling, Thomas; Hackenberg, Ulrich; Wundram, Kai; Goß, Stefan (2021): Produktentstehungsprozess. In: Stefan Pischinger und Ulrich Seiffert (Hg.): Vieweg Handbuch Kraftfahrzeugtechnik. Wiesbaden: Springer Fachmedien Wiesbaden, S. 1271–1382.

Witte, Frank (2020): Strategie, Planung und Organisation von Testprozessen. Basis für erfolgreiche Projektabwicklung im Softwaretest. Wiesbaden: Springer Vieweg.

Wolff, Klaus; Futschik, Hans Dieter; Achleitner, August; Burgers, Christiaan; Döllner, Gernot (2021): Gesamtfahrzeug. In: Stefan Pischinger und Ulrich Seiffert (Hg.): Vieweg Handbuch Kraftfahrzeugtechnik. Wiesbaden: Springer Fachmedien Wiesbaden, S. 125–167.

Zwintzscher, Olaf (2005): Software-Komponenten im Überblick. Einführung, Klassifizierung & vergleich von JavaBeans, EJB, COM+,.Net, CORBA, UML 2. Herdecke, Bochum: W3L GmbH.

Publikationen

1. Jooß, Benedikt; Rotter, Thomas, Schramm, Dieter (2021): The efficient use of HiL simulation and vehicle tests for in-house software development at Porsche. In Peter E. Pfeffer (Hg.): Proceedings of the 11th International Munich Chassis Symposium 2020: chassis tech plus. Berlin, Heidelberg: Springer-Verlag GmbH

2. Jooß, Benedikt; Schramm, Dieter (2022): Komplexität von Softwaretests in der Fahrwerksentwicklung. In: Automobiltechnische Zeitschrift 04/2022, S. 44-47
3. Jooß, Benedikt; Schuld, Julian; Enderle Matthias; Schramm Dieter (2022): Testing of OTA-enabled functions in electronic control unit development. In: Manuscripte of the 9th AutoTest Fachkonferenz: Test von Hardware und Software in der Automobilentwicklung 2022. Verfügbar: https://www.fkfs-veranstaltungen.de/veranstaltungen/autotest/bilder-der-veranstaltung/manuskripte, zuletzt geprüft am 21.10.2022
4. Jooß, Benedikt; Schramm, Dieter (o. J.): Modular test kit – A modular approach for efficient and function-oriented testing. (Publikation eingereicht, Status: “Accept” vom 05.04.2023)

DOI: 10.17185/duepublico/78447
URN: urn:nbn:de:hbz:465-20230609-102509-2