UNIKASSEL VERSITÄT

Kasseler Schriftenreihe Qualitätsmanagement Band 9

Methode des Anforderungsmanagements zur Variantenvermeidung in der frühen Phase der Produktentwicklung

Lisa Ritter

UNIKASSEL VERSITÄT
Fachgebiet Qualitäts- & Prozessmanagement

Prof. Dr.-Ing. Robert Refflinghaus (Hrsg.)
Prof. Dr.-Ing. Roland Jochem (Hrsg.)

Kasseler Schriftenreihe Qualitätsmanagement

Herausgegeben vom Fachgebiet Qualitäts- und Prozessmanagement
an der Universität Kassel

Band 9

Herausgeber:
Prof. Dr.-Ing. Robert Refflinghaus
Prof. Dr.-Ing. Roland Jochem

Fachgebiet Qualitäts- & Prozessmanagement

Lisa Ritter

Methode des Anforderungsmanagements zur Variantenvermeidung in der frühen Phase der Produktentwicklung

kassel university press

Die vorliegende Arbeit wurde Fachbereich Maschinenbau der Universität Kassel als Dissertation zur Erlangung des akademischen Grades einer Doktorin der Ingenieurwissenschaften (Dr.-Ing.) angenommen.

Erster Gutachter: Prof. Dr.-Ing. Robert Refflinghaus
Zweiter Gutachter: Prof. Dr.-Ing. Roland Jochem

Tag der mündlichen Prüfung: 27. April 2023

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

Zugl.: Kassel, Univ., Diss. 2023
ISBN: 978-3-7376-1140-4
DOI: https://doi.org/10.17170/kobra-202308288695

Druck und Verarbeitung: Print Management Logistik Service, Kassel
Printed in Germany

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit als Doktorandin bei der BMW AG in der Entwicklung Fahrdynamik in München. An dieser Stelle möchte ich mich bei einigen Personen bedanken, die mich auf dem Weg des Promotionsverfahrens begleitet haben.

Mein besonderer Dank für die wissenschaftliche Betreuung der Arbeit und das mir entgegengebrachte Vertrauen gilt Herrn Prof. Dr.-Ing. Robert Refflinghaus, Leiter des Fachgebiets für Qualitäts- und Prozessmanagement des Instituts für Arbeitswissenschaft und Prozessmanagement (IfA) der Universität Kassel. Seine stetige Unterstützung und wertvollen Anregungen haben in besonderem Maße zum Gelingen meiner Arbeit beigetragen.

Prof. Dr.-Ing. Roland Jochem, Leiter des Fachgebiets Qualitätswissenschaft des Instituts für Werkzeugmaschinen und Fabrikbetrieb der Technischen Universität Berlin, danke ich für sein Interesse an meiner Arbeit und die Übernahme des Korreferats.

Im ProMotion Programm der BMW AG wurde ich durch Herrn Dr.-Ing. Tobias Diekmann betreut. Sein detailliertes Wissen zu den Anforderungsmanagementprozessen der BMW AG sowie die inhaltlichen Diskussionen und pragmatischen Empfehlungen halfen mir bei der Auseinandersetzung mit dem Forschungsthema. Hierfür gilt ihm mein besonderer Dank.

Nicht zuletzt danke ich meinen ehemaligen Kollegen und Kolleginnen der Entwicklung Fahrdynamik und der IT Data Science, Natural Language Processing, and Artificial Intelligence für die konstruktiven Anmerkungen und die kollegiale Arbeitsatmosphäre. An dieser Stelle möchte ich mich insbesondere bei Herrn Dr.-Ing. Gerrit Kiesgen† und Herrn Steffen Schmitz bedanken, die mir die Teilnahme am ProMotion Programm der BMW AG ermöglicht haben.

Abschließend bedanke ich mich herzlich bei meinen Eltern und meinem Partner für das mir in dieser Zeit entgegengebrachte Verständnis, den uneingeschränkten Rückhalt und den motivierenden Zuspruch.

München, November 2022 Lisa Ritter

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

1.1 Motivation

Die Produktentwicklung trägt durch ihre Entscheidungen zur Produktgestaltung die Verantwortung für die Kosten, die in den späteren Phasen der Produktherstellung in der Fertigung, Logistik, Service etc. entstehen (Ehrlenspiel et al. 2007, S. 5ff.). Die Verantwortung in der Entwicklung ist somit für die Wirtschaftlichkeit eines Unternehmens hoch. Gleichzeitig wird die Erfüllung aller Ansprüche an die Produktentwicklung im Spannungsdreieck aus Kosten, Qualität und Zeit (Atkinson 1999, S. 337f.) herausfordernder: Ein großer Kostentreiber sind die Komplexitätskosten, auch varianteninduzierte Mehrkosten eines Unternehmens, die durch die Variantenvielfalt seiner Produkte und Prozesse entstehen (Schuh 2005, S. 45ff.). Insbesondere Varianten, die von Kunden nicht wahrgenommen und daher nicht bestellt werden, sind wirtschaftlich nicht sinnvoll und machen dennoch 30% der angebotenen Varianten aus (Schlott 2005, S. 40). Nach Kerstens Studie (2002, S. 2f.) werden 27,3% aller Produktvarianten und 36,5% aller Komponentenvarianten von den befragten Unternehmen als nicht erforderlich bezeichnet. Diese Zahlen zeigen, dass eine kundenorientierte Produktentwicklung basierend auf den Anforderungen der Kunden und Märkte wichtig ist und unwirtschaftliche Varianten für Unternehmen einen großen Kostentreiber darstellen. Bereits seit den 1980-er Jahren wird die wachsende Variantenvielfalt in der Literatur thematisiert (vgl. Hichert 1985; Zimmermann 1988) und soll in der Entwicklung durch Methoden der Standardisierung, Plattformen, Baukästen und Baureihen gestaltet werden (vgl. Schuh 1989), um im Management der Zielkonflikte zwischen Kosten, Qualität und Zeit das wirtschaftlich optimalste Ergebnis zu erzielen. Gerade im Hinblick auf die Variantenvielfalt und dem Trend zur Individualisierung stehen Unternehmen vor der Herausforderung, einerseits den Kundenanforderungen nach einer größtmöglichen externen Variantenvielfalt Rechnung zu tragen und andererseits die damit einhergehende interne Variantenvielfalt der Produkte, Komponenten und Prozess und ihrer Komplexität zu beherrschen bei gleichzeitiger Zunahme der Anforderungsmenge (Boutkova 2014, S. 5; Kubica 2007, S. 3). Dabei ist der Grad, in dem die Anforderungen durch das Produkt erfüllt werden, der Maßstab der Qualität (DIN EN ISO 9000, S. 39).

Aufgrund der immer kürzer werdenden Produktlebenszyklen müssen Unternehmen ihre Produkte schneller in den Markt bringen, sodass auch die Entwicklungszeit kürzer

wird. Durch den hohen Zeit- und Kostendruck werden Anforderungen von Entwicklern aller Industrien bei Anpassungs- und Variantenentwicklung aus Vorgängerprojekten kopiert (Palomares et al. 2014, S. 303). Dieses Vorgehen der Wiederverwendung von Anforderungen spart viel Zeit für die Erfassung, Dokumentation und Analyse von Anforderungen und kann zu einer höheren Qualität führen, da bereits validierte Anforderungen verwendet werden. Gleichzeitig besteht bei einer unstrukturierten Wiederverwendung, wie sie in der Praxis laut Studien (Palomares et al. 2014, S. 303) vorherrscht, das Risiko, nicht zutreffende, veraltete und/oder fehlerhafte Anforderungen über viele Entwicklungsprojekte hinweg zu kopieren. Dies kann zu einer schlechteren Qualität, höheren Entwicklungskosten und somit zu höheren Produktpreisen führen. Durch die unstrukturierte Wiederverwendung von Anforderungen in der Praxis mittels Copy-Paste-Vorgehen wächst die Anforderungsmenge immer weiter an und die Übersicht sowie Transparenz über die Unterschiede der Anforderungen sinkt. Aufgrund dieser mangelnden Transparenz über die Unterschiedlichkeit und Ähnlichkeit der Anforderungen und den daraus abgeleiteten Merkmalen einer Produktkomponente können potenziell wirtschaftlich nicht sinnvolle Varianten entstehen.

Im Bereich des Anforderungsmanagements gibt es in der Literatur keine Ansätze, die die Produktentwicklung in der Wiederverwendung von technischen Anforderungen an mechanische Komponenten unterstützt, um eine historisch gewachsene, unstrukturierte Anforderungsmenge automatisiert für eine Wiederverwendung zu klassifizieren und hinsichtlich potenzieller Variantentreiber zu analysieren. An dieser Stelle der Wiederverwendung von Anforderungen im Produktenwicklungsprozess setzt diese Arbeit an.

1.2 Zielsetzung und Einordnung der Arbeit

Die Zielsetzung der vorliegenden Arbeit ist die Entwicklung einer Methode des Anforderungsmanagements zur Variantenvermeidung und Unterstützung der Wiederverwendung von Anforderungen in der frühen Phase der Produktentwicklung. Das Anforderungsmanagement ist ein Teilgebiet des Qualitätsmanagements und definiert Anforderungen als die Aufgabenstellung an die Entwicklung. Am Grad der Anforderungserfüllung durch ein Produkt lässt sich die Kundenzufriedenheit und die Qualität bemessen. Im Verständnis des Qualitätsmanagements ist die kundenorientierte Produktentwicklung essenziell, um mit einem Produkt im Markt erfolgreich zu sein (Elser et al. 2021, S. 280; Gussen et al. 2021, S. 514; Ponn et al. 2011, S. 60; vgl. Kamiske 2008, S. 12).

Um unwirtschaftliche Varianten bereits in der frühen Phase der Produktentwicklung in der Definition der Anforderungen zu verhindern, die aufgrund von Defiziten in der Erhebung, Dokumentation und Strukturierung der Anforderungen entstehen können, bedarf es einer Transparenz über die variantenverursachenden Anforderungen und die Unterschiede in den Anforderungen zwischen den Varianten. Aufgrund des Zeit- und Kostendrucks in der Praxis bedarf es automatisierter Verfahren, um diese Anforderungsmenge für die Wiederverwendung zu strukturieren und die Unterschiede transparent zu machen. Die in der Literatur beschriebenen Ansätze zur Wiederverwendung insbesondere von deutschsprachigen Anforderungen bieten keine automatisierte Lösung für einen semantischen Vergleich einer bereits bestehenden Anforderungsmenge und sind daher aufgrund ihres Aufwand-Nutzen-Verhältnisses nicht praktikabel. Hieraus lassen sich die beiden Lösungshypothesen für diese Arbeit ableiten, die die Zielsetzung detaillieren.

Die erste Lösungshypothese dieser Arbeit ist, dass mithilfe eines automatisierten Anforderungsvergleichs basierend auf der semantischen Ähnlichkeit unter Anwendung von Verfahren aus dem Bereich Natural Language Processing inhaltliche Unterschiede in den Anforderungen zwischen verschiedenen Entwicklungsprojekten transparent werden. Automatisiert bedeutet hier gemäß der DIN IEC 60050-351, dass ein Prozess *„[…] unter festgelegten Bedingungen ohne menschliches Eingreifen abläuft oder arbeitet“ (DIN IEC 60050-351, S. 30)*. Durch diese automatisierten Verfahren kann eine historisch gewachsene, unstrukturierte Anforderungsmenge, die in verschiedenen Dokumenten und Dateien vorliegt, in relativ kurzer Zeit automatisch analysiert werden. Das Ergebnis dieses Vergleichs ist die Häufigkeit des Vorkommens einer Anforderung in den verglichenen Dokumenten als auch die Identifizierung des Ausprägungsmerkmals und somit auch die Unterschiedlichkeit dieser Ausprägung in den verglichenen Dokumenten.

Die zweite Lösungshypothese baut auf dieser Analyse des Anforderungsvergleichs auf. Sie besagt, dass Unterschiede in den Anforderungen und somit die variablen Anforderungen zu Varianten des Produktes bzw. der Komponente führen. Daher werden die Anforderungen basierend auf den Ergebnissen des Anforderungsvergleichs hinsichtlich ihrer variantenverursachenden Wirkung klassifiziert.

Durch diese Lösungen soll der Aufbau einer Anforderungsbibliothek basierend auf den bestehenden Anforderungsdokumenten zur Unterstützung der Entwickler in der Wiederverwendung von Anforderungen möglich werden. Anhand der klassifizierten

Anforderungen können Entwickler effizient hinterfragen und entscheiden, welche Anforderungen sie für ihr jeweiliges Entwicklungsprojekt übernehmen wollen und in welchen Wertebereichen sie die Schwankungen ihrer ausgeprägten Anforderungen am besten harmonisieren. Aufgrund dieses Praxiszusammenhangs und der Orientierung an der Praktikabilität und Nutzen für die Praxis fällt diese Arbeit in die Fragestellungen der angewandten Wissenschaft im Gegensatz zur Grundlagenwissenschaft, die sich mit der Schließung von Theorielücken beschäftigt.

Die Limitationen der entwickelten Methode werden durch den Objektbereich verdeutlicht. Der Objektbereich dieser Arbeit sind deutschsprachige, technische Anforderungen an mechanische Komponenten. Die Methode wird für die Anwendung auf deutschsprachige Komponentenlastenhefte entwickelt. Dabei müssen die Anforderungen in Textformat vorliegen. Anforderungen auf technischen Zeichnungen, Abbildungen oder Diagrammen werden nicht interpretiert oder verarbeitet. Des Weiteren betrachten viele in der Literatur behandelten Methoden zur Wiederverwendung von Anforderungen Softwarekomponenten. Diese Arbeit soll sich von diesem Fokus abgrenzen, in dem sie ausschließlich mechanische Komponenten betrachtet.

1.3 Aufbau der Arbeit

Die vorliegende Arbeit ist in fünf Kapitel strukturiert. Zunächst wird im einleitenden Kapitel 1 die Problemstellung im Praxiszusammenhang sowie die Zielsetzung der Arbeit erläutert. Hierauf folgt in Kapitel 2 eine Erläuterung der wichtigsten Begriffe und des Stands der Technik. Da die Arbeit drei Forschungsbereiche verknüpft, untergliedert sich Kapitel 2 in die Bereiche Anforderungsmanagement und Produktentwicklung (2.1), Variantenmanagement (2.2) und Natural Language Processing (2.3) sowie einer Abgrenzung der Arbeit zu bestehenden Ansätzen und Aufzeigen der Forschungslücken in Unterkapitel 2.4. Anhand dieser Forschungslücken werden die Forschungsfragen dieser Arbeit detailliert. In Kapitel 3 wird das Konzept der entwickelten Methode erläutert. Dabei wird zunächst auf die Grundidee der Unterscheidung in projektübergreifende, spezifische und ausgeprägte Anforderungen eingegangen, um anschießend das Vorgehen zur Umsetzung eines automatisierten Vergleichs zu beschreiben. Des Weiteren wird der Aufbau einer Anforderungsbibliothek basierend auf der Auswertung des automatisierten Anforderungsvergleichs dargelegt. Eine Zusammenfassung des Konzepts sowie das Aufzeigen der Limitationen der entwickelten Methode schließen Kapitel 3 ab.

Die Anwendung und Validierung der entwickelten Methode wird in Kapitel 4 beschrieben. Dieses Kapitel unterteilt sich in vier Unterkapitel. Zunächst werden in Kapitel 4.1 die Rahmenbedingungen des Anwendungsfalls sowie der anwendungsfallspezifische Aufbau des Sprachmodells für den automatisierten Vergleich und die Auswahl der Validierungsbeispiele vorgestellt. Die anschließende Auswertung der Ergebnisse der Anwendung wird in drei Schritte basierend auf den Forschungsfragen unterteilt: Kapitel 4.2 zeigt die Validierung des semantischen Vergleichs auf, Kapitel 4.3 die Überprüfung der automatisierten Klassifikation der Anforderungen und Kapitel 4.4 untersucht, ob die identifizierten Anforderungen in den Anwendungsfällen tatsächlich zu Varianten der Komponenten geführt haben. Zum Abschluss der Arbeit erfolgt in Kapitel 5 eine Zusammenfassung der gewonnenen Erkenntnisse sowie ein Ausblick über weiterführende Forschungsfragen.

2 Grundlagen und Stand der Technik

Dieses Kapitel soll einerseits durch grundlegende Begriffsdefinitionen ein gemeinsames Verständnis der behandelten Themen Produktentwicklung, Anforderungsmanagement, Variantenmanagement und Natural Language Processing schaffen. Andererseits sollen durch ein Aufzeigen bestehender Ansätze in den genannten Themenbereichen der Stand der Technik dargestellt und Forschungslücken aufgezeigt werden. Durch die Erläuterung bestehender Ansätze in der Literatur erfolgt eine Abgrenzung der in Rahmen dieser Arbeit entwickelten Methode.

2.1 Anforderungsmanagement und Produktentwicklung

Die Produktentwicklung und das Anforderungsmanagement sind zwei wichtige Säulen für den Erfolg eines Unternehmens. Das Ziel eines Unternehmens ist es, mit einem Produkt Kundenbedürfnisse zu erfüllen und dadurch Gewinn zu erzielen (Bender et al. 2016, S. 401). Daher sind die Identifizierung, Untersuchung und Umsetzung der Kundenwünsche und -anforderungen in der Produktentwicklung ein wichtiger Baustein für ein erfolgreiches Produkt (vgl. Ponn et al. 2011, S. 60). Der Grad, in dem ein Produkt die an sich gestellten Anforderungen erfüllt, wird als **Qualität** bezeichnet (DIN EN ISO 9000, S. 39). In diesem Kapitel sollen die Begriffe der Produktentwicklung als auch des Anforderungsmanagements erläutert werden. Dazu wird zunächst auf die Definition des Produktes, den Produktlebenszyklus als auch die verschiedenen Formen eines Produktentwicklungsprojektes eingegangen. Anschließend erfolgt die Definition des Begriffs Anforderung, eine Erläuterung der Anforderungsarten sowie des Aufbaus einer Anforderung, der Dokumentationsformen von Anforderungen und den Aufgaben des Anforderungsmanagements. Im letzten Unterkapitel werden verschiedene Ansätze der Wiederverwendung von Anforderungen in Entwicklungsprojekten aufgezeigt.

2.1.1 Produkt, Produktlebenszyklus und Entwicklungsprojekte

Produkte sind das hergestellte Ergebnis eines Entwicklungsprozesses, die sowohl materiell als auch immateriell sein können (VDI 2221, S. 41). Zu Beginn eines jeden **Produktlebenszyklus** steht eine Produktidee als Entwicklungsauftrag (Gebhardt 2020, S. 7). In einer ersten Phase werden die Anforderungen und Bedürfnisse der Stakeholder an die Produktidee ermittelt (Baumgart 2016, S. 425), um anschließend in der Entwicklungs- und Konstruktionsphase in Produktmerkmale übersetzt zu werden. Dieser Transfer von Kundenanforderungen in Produktmerkmale kann durch den Einsatz des

Quality Function Deployment (QFD) als Methode des Qualitätsmanagements unterstützt werden (vgl. ISO 16355-1; Refflinghaus 2000, S. 26). Eine Weiterentwicklung des QFD ist das **House of Quality (HoQ)**, dass in einem mehrstufigen Verfahren Kundenanforderungen in Produktmerkmale und weiter in technische Anforderungen an einzelne Komponenten übersetzt als auch deren Bewertung ermöglicht (Refflinghaus 2000, S. 26). Das Ergebnis der Entwicklungsphase, auch als **Produktentwicklungsprozess** bezeichnet (vgl. Lindemann 2016, S. 402), ist das Produkt, das in Form von Dokumenten, Spezifikationen, digitalen Modellen, Simulationen, Berechnungen sowie Ausführungs- und Nutzungsangaben definiert ist (Eigner et al. 2013, S. 10 und VDI 2221, S. 7). Liegt nach Abschluss dieses Produktentwicklungsprozesses das produktionsbereite Produkt vor, beginnt die Phase der Produktherstellung. Nach erfolgter Produktion wird durch den Vertrieb das Produkt an den Kunden übergeben, der es für eine begrenzte Nutzungsdauer gebraucht. Durch Instandhaltung des Produktes kann die Nutzungsdauer verlängert werden. Nach Beendigung der Nutzungsdauer wird das Produkt dem Recycling zugeführt, mit dem der Produktlebenszyklus endet. (VDI 2221, S. 7f.) Abbildung 2-1 visualisiert die Phasen des Produktlebenszyklus von der Produktidee bis hin zum Recycling einschließlich des Verlaufs des Gewinns und des Umsatzes, den ein Unternehmen mit einem Produkt erwirtschaftet.

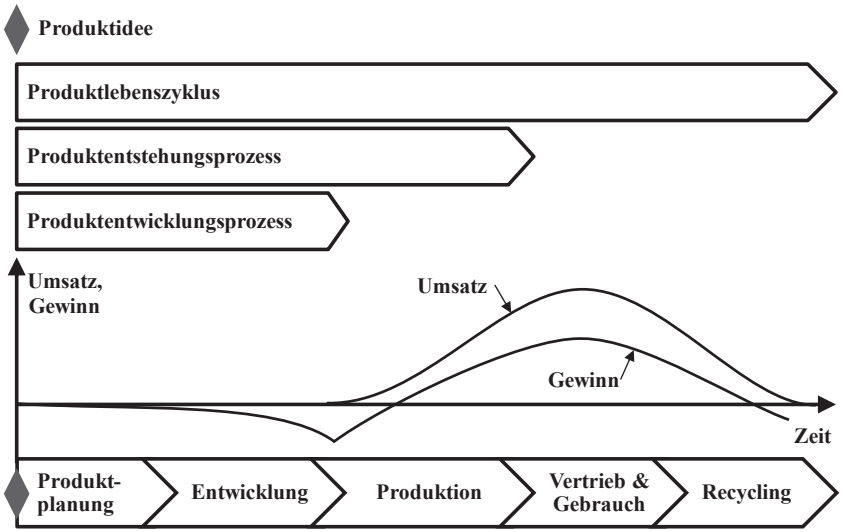


Abbildung 2-1: Produktlebenszyklus - Einordnung des Produktentwicklungsprozesses (eigene Darstellung in Anlehnung an Matys 2018, S. 139, Eigner et al. 2013, S. 27, Pahl et al. 2007, S. 114 und VDI 2221, S. 9)

Des Weiteren zeigt die Abbildung, dass der Produktentstehungsprozess einen Abschnitt im Produktlebenszyklus von der Idee bis zum hergestellten Produkt ist, der den Produktentwicklungsprozess enthält. Der Produktentwicklungsprozess endet bereits mit dem Ergebnis der Entwicklung und Konstruktion eines herstellbaren Produktes (vgl. Lindemann 2016, S. 402).

Der Produktentwicklungsprozess wird in der Industrie in Form von **Produktentwicklungsprojekten** organisiert. Dabei ist laut DIN 69901-5, S. 11 ein Projekt ein „*Vorhaben, das im Wesentlichen durch Einmaligkeit der Bedingungen in ihrer Gesamtheit gekennzeichnet ist*". Im spezifischen Kontext der Produktentwicklung wird der Begriff Projekt zu Produktentwicklungsprojekt erweitert, der eine zeitlich und inhaltlich begrenzte Entwicklungsaufgabe mit einer definierten Zielvorgabe, das neue Produkt, bezeichnet (vgl. Dölle 2018, S. 24; Paulukuhn 2005, S. 20).

Bei Produktentwicklungsprojekten können aufgrund der Neuartigkeit des Produktes, aber auch aufgrund der Einbindung von Zulieferern und Entwicklungsdienstleistern verschiedene Arten der Entwicklungsprojekte unterscheiden werden, auf die im Folgenden eingegangen wird.

Komplexe Produkte werden in verschiedene Ebenen strukturiert. So gibt es die Gesamtproduktebene, die Systemebene, die Teilsystemebene und die Komponentenebene. Als **Komponente** bezeichnet man je nach Betrachtungsebene eine Baugruppe z.B. ein Steuergerät bestehend aus der physischen Einheit einschließlich der Anschlüsse und Schrauben sowie der darauf installierten Software oder man versteht darunter ein einzelnes Bauteil. *„Somit ist eine Komponente ein Teil, welches in einem übergeordneten anderen Teil verbaut wird."* (Hüttenrauch et al. 2008, S. 130) Insbesondere bei komplexen Produkten ist es wichtig, Anforderungen über die verschiedenen Ebenen von Gesamtprodukt, System, Teilsystem bis hin zur einzelnen Komponente eindeutig zu dokumentieren und an den jeweiligen Anforderungsnehmer zu kommunizieren, denn in die Entwicklungsarbeit werden häufig **Entwicklungsdienstleister (EDL)** und Zulieferer der einzelnen Komponenten eingebunden. Der VDA bestätigt das Wachstum der EDL seit den 1970er Jahren und gibt den Umsatz der weltweit 25 größten EDL mit 5,7 Mrd. Euro für das Jahr 2013 an (VDA 2015, S. 60). EDL bzw. -lieferanten erhalten von ihrem Auftraggeber, den OEMs, Vorgaben für Bauteile oder Baugruppen, um diese im Auftrag zu konstruieren, allerdings nicht selbst herzustellen (Reindl 2013, S. 49).

Clark et al. (1992, S. 142) unterscheiden Zuliefereigenentwicklungen, Black-Box-Teile und Herstellereigenentwicklungen, die sich in der Einbindung der Zulieferer in

den Entwicklungsprozess und Entwicklungsleistung unterscheiden. **Zuliefereigenentwicklungen** werden durch den Zulieferer selbst entwickelt und hergestellt. Hingegen wird die Entwicklungsarbeit bei s.g. **Black-Box-Teilen** zwischen Hersteller/OEM und Zulieferer aufgeteilt. Hierbei werden vom Hersteller die Anforderungen an Leistung, Kosten, Außenformen, Schnittstellen und weitere Grundentwurfsanforderungen in Form von Lastenheften an mehrere potentielle Zulieferer weitergegeben, die sich um den Entwicklungsauftrag bewerben (Clark et al. 1992, S. 144). Diese Vergabeform gibt es nicht nur auf Komponentenebene, sondern auch als **Systemlieferanten** mit größeren Umfängen. Systemlieferanten übernehmen eine größere Verantwortung als Zulieferer von einzelnen Komponenten, da sie in der Entwicklungsarbeit die Integration in das Endprodukt als auch die Problemlösung verantworten und dabei ggf. mit anderen Lieferanten innerhalb ihres Moduls kooperieren müssen. Durch den Hersteller werden lediglich die Grundeigenschaften wie Funktionen und Abmessungen in einem Systemlastenheft festgelegt. Die Umsetzung erfolgt durch den Systemlieferanten. (Reindl 2013, S. 52) Herstellereigenentwicklungen werden vollständig durch den OEM entwickelt und zur Produktion an Lieferanten vergeben. Diese **Produktionslieferanten** fertigen den Umfang gemäß der konstruktiven und fertigungstechnischen Anforderung des OEM (Wolters 1995, S. 7). Die Fertigung vorentwickelter Umfänge wird auch als **Build-to-Print** bezeichnet. (Reindl 2013, S. 50)

Basierend auf der Neuartigkeit des Produktes werden Neu-, Anpassungs- und Variantenentwicklung unterschieden. **Neuentwicklung** bezeichnet Entwicklungsprojekte, bei denen für eine veränderte oder neue Aufgabenstellung neue Lösungsprinzipien entwickelt werden. Dabei werden sowohl neue als auch bekannte Technologien als Baugruppen und Bauteile eingesetzt. Sie zeichnen sich durch eine hohe Neuartigkeit mit experimentellem Charakter aus. Das technische und wirtschaftliche Risiko von Neuentwicklungen ist vergleichsweise hoch. Neuentwicklungen, auch Neukonstruktionen, können sowohl das Gesamtprodukt als auch einzelne Baugruppen oder Bauteile beschreiben. (Schmelzer 1992, S. 18–19; Pahl et al. 2007, S. 4) **Anpassungsentwicklungen** basieren auf bereits bekannten Lösungsprinzipien und Technologien. Das Produkt wird dabei an veränderte Anforderungen über die Anordnung, Dimensionierung und Gestaltung von Komponenten angepasst, einzelne Bauteile oder Baugruppen werden neu entwickelt. (Schmelzer 1992, S. 18f.; Pahl et al. 2007, S. 4) Im Gegensatz zu Neuentwicklungen liegen hier bereits mehr Informationen über mögliche Lösungsmethoden vor, die Funktions- und Baustruktur bleiben gleich. (Schmelzer 1992, S. 20) Die **Variantenentwicklung** ist die

„Variation von Baugruppen oder Bauteilen bei gleichbleibenden Lösungsprinzipien“ (Schmelzer 1992, S. 18). Der Aufwand einer Neuentwicklung fällt hierbei einmalig bei der Konstruktion der Basis an. Anschließend werden Größe oder Anordnung von Bauteilen variiert. Beispiele sind hierfür Baureihen und Baukästen. (Pahl et al. 2007, S. 4) Hierbei ist es analog zur Anpassungsentwicklung möglich, auf vorhandene Erfahrung zurückzugreifen sowie Möglichkeiten zur Standardisierung zu nutzen (Schmelzer 1992, S. 20).

Die unterschiedlichen Formen der Produktentwicklungsprojekte zeigen einerseits aufgrund der vielen Entwicklungspartner und Schnittstellen die Notwendigkeit eines guten Anforderungsmanagements, andererseits die Anpassungs- und Variantenentwicklung Chancen zur Komplexitätsreduzierung. Auf diese beiden Themenkomplexe Anforderungsmanagement und Variantenmanagement wird in den folgenden Kapiteln weiter eingegangen.

2.1.2 Anforderungen, Anforderungsmanagement und Dokumentation

Eine **Anforderung** ist laut DIN EN ISO 9000 ein *„Erfordernis oder Erwartung, das oder die festgelegt, üblicherweise vorausgesetzt oder verpflichtend ist“.* Der IEEE Std 610.12 erweitert diese Definition: Hier wird die Anforderung als von einem Anwender benötigte Bedingung oder Eigenschaft zur Problemlösung oder Zielerreichung definiert. Des Weiteren wird die Anforderung als Bedingung oder Eigenschaft beschrieben, die durch ein System oder eine Systemkomponente eingehalten werden muss, um einen Vertrag, Standard oder Spezifikation zu erfüllen. In beiden Fällen stellt die Anforderung die Dokumentation dieser Bedingung oder Eigenschaft dar. (IEEE Std 610.12) Diese Definition einer Anforderung aus Sicht eines Anwenders oder Auftraggebers als eine zu erfüllende Eigenschaft oder Leistung eines Produktes wird in der Literatur geteilt (vgl. Versteegen et al. 2004, S. 3f.; Ilie 2013, S. 11f.).

Rupp (2014, S. 219ff.) unterscheidet je nach Anforderungsart verschiedene Anforderungsschablonen (Funktions-, Eigenschafts-, Umgebungs-, Prozess- oder Bedingungsmaster) mit jeweils verschiedenen Anforderungselementen. Nach Kickermann (1995) setzt sich eine Anforderung aus folgenden Elementen zusammen: Anforderungsgegenstand, Anforderungsattribut, Forderungsgrad, Ausprägung und ggf. einer Referenz und einer Anforderungsbedingung. Der **Anforderungsgegenstand** beschreibt hierbei das Bezugsobjekt, auf das sich eine Anforderung bezieht und dessen Ausprägung durch die Anforderung definiert wird (vgl. Kickermann 1995, S. 25; Ahrens 2000, S. 213). In einigen Arbeiten wird dieses Bezugsobjekt auch als **Anforderungsmerkmal** (vgl. Ponn et al. 2011, S. 39; Ilie 2013, S. 11) oder als Betrachtungsgegenstand (Rupp 2014,

S. 235ff.) bezeichnet. Die **Ausprägung** einer Anforderung definiert den Soll-Wert des Anforderungsgegenstandes. Dabei wird anhand der Ausprägung in qualitative und quantitative Anforderungen unterschieden. **Quantitative** Anforderungen zeichnen sich durch einen Größenwert und i.d.R. einer Einheit aus. **Qualitative** Anforderungen hingegen weisen an dieser Stelle einen verbalen Ausdruck auf. (Ahrens 2000, S. 213; Kickermann 1995, S. 70; Ponn et al. 2011, S. 39). Rupp (2014, S. 235ff.) bezeichnet diesen Bestandteil in der Anforderungsschablone Eigenschaftsmaster als Eigenschaft und Wert.

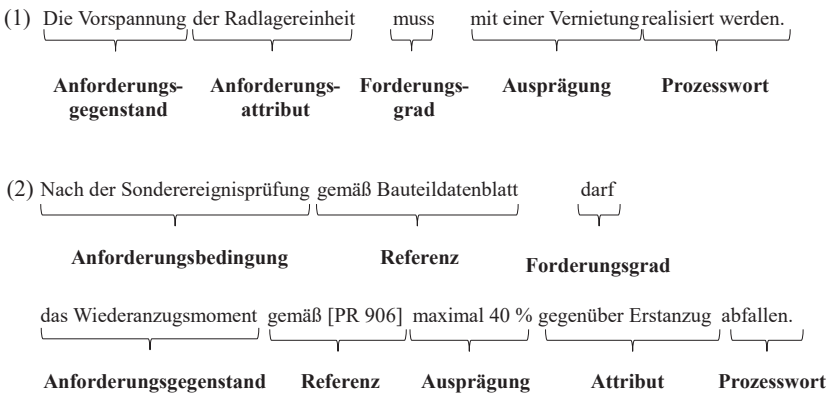


Abbildung 2-2: Zusammensetzung einer Anforderung
(Eigenes Beispiel in Anlehnung an Kickermann 1995, S. 25ff. und Rupp 2014, S. 227)

In Beispiel (1) in der Abbildung 2-2 ist der Gegenstand der Anforderung die Vorspannung. Die Ausprägung definiert, wie der Anforderungsgegenstand umgesetzt werden soll, hier als Vernietung. In diesem Fall handelt es sich um eine qualitative Anforderung, da die Ausprägung „Vernietung" keinen Zahlenwert enthält. Hingegen wird in Beispiel (2) der Anforderungsgegenstand „Wiederanzugsmoment" durch die Ausprägung „maximal 40%" als quantitative Anforderung charakterisiert. Beispiel (1) weist einen weiteren Bestandteil auf: das **Anforderungsattribut**. Das Attribut hat die Funktion, zusätzliche Informationen zum Anforderungsgegenstand zu geben. Im Beispiel wird der Anforderungsgegenstand „Vorspannung" durch das Attribut „Radlagereinheit" präzisiert. Der **Forderungsgrad** der Anforderung wird in beiden Beispielen durch die Schlüsselworte „muss" und „darf" definiert. Diese Begriffe zeigen als Forderungsgrad ein Maß für die Wichtigkeit und rechtlichen Verbindlichkeit einer Anforderung (Rupp 2014, S. 226).

Das Beispiel (2) weist zwei weitere Bestandteile auf: die Referenz und die Anforderungsbedingung. Mit Hilfe der **Referenz** wird eine zusätzliche Information zu einer in Verbindung mit der Anforderung zu berücksichtigendem Begleitdokument gegeben z.B. Verweise auf zu erfüllende Standards, Richtlinien oder Datenblätter. Die **Anforderungsbedingung** wiederum stellt die Abhängigkeit einer Anforderung von einem bestimmten Szenario dar. Signalwörter für solche konditionale Bedingungen sind „Wenn…, dann…" Formulierungen oder wie in Beispiel (2) nach einem bestimmten Vorereignis als zeitliche Bedingung. (Rupp 2014, S. 240ff.; Kickermann 1995, S. 25ff.)

Anforderungsarten

In der Literatur werden verschiedene Arten von Anforderungen unterschieden. Die am weitesten verbreitete Definition unterscheidet funktionale Anforderungen und nicht-funktionale Anforderungen. **Funktionale Anforderungen** beschreiben, was ein Produkt leisten kann, das Verhalten des Produktes und seine Funktionalität (vgl. Rupp 2013, S. 19f.; Pohl 2007, S. 15; Boehm o.J., S. 16; Grande 2014, S. 37). Demgegenüber stehen **nicht-funktionale Anforderungen**, die konkrete Eigenschaften des Produktes definieren und sich auf ein bestimmtes Konzept der funktionalen Anforderungen beziehen (Boehm o.J., S. 16). Nicht-funktionale Anforderungen werden in Qualitätsanforderungen und Randbedingungen unterteilt.

Qualitätsanforderungen beziehen sich auf funktionale Anforderungen, indem sie beschreiben, wie und in welcher Qualität das Produkt die Funktionen erfüllt. Rupp (2013, S. 19f.) und Hood et al. (2007, S. 17) zählen Anforderungen an die Effizienz, Zuverlässigkeit, Benutzbarkeit, Änderbarkeit und Übertragbarkeit des Produktes zur Gruppe der Qualitätsanforderungen. **Randbedingungen** sind laut Ebert (2010, S. 28) Anforderungen bezüglich Kosten, Marketing, Durchlaufzeit, Vertrieb, Organisation und Dokumentation. Sie begrenzen die Lösungsmöglichkeiten in der Produktgestaltung, geben Schnittstellen und Hardware vor und legen Benutzerschnittstellen und Entwicklungsvorgehensweisen und -werkzeuge fest (Rupp 2013, S. 19ff.). Abweichend hiervon werden speziell Vorgaben zu Vorgehensweisen, anzuwendende Werkzeuge und Standards von Hood et al. (2007, S. 17) als **Projektanforderungen** bezeichnet. Solche rechtlich-vertragliche Vorgaben zur Zusammenarbeit im Entwicklungsprojekt sowie Lieferbestandteile fallen dennoch in die Gruppe der Randbedingungen (Rupp 2013, S. 19ff.). Darüber hinaus werden detaillierte Anforderungen wie Stücklisten, technische Zeichnungen, Schaltpläne und Montageanweisungen als **Designanforderungen** bezeichnet (Hood et al. 2007, S. 17), die als Randbedingungen den Lösungsraum einschränken.

Randbedingungen, die die Lösung einschränken oder sogar die Realisierung beschreiben, als Anforderungen zu bezeichnen, widerspricht Ebert (2010, S. 21). Die Unterscheidung in Problemraum und Lösungsraum soll verdeutlichen, dass im Problemraum durch Anforderungen ein zu erfüllendes Bedürfnis oder Nutzen in Form von Marktanforderungen und Lastenheften beschrieben wird. Im Lösungsraum hingegen wird die Umsetzung, die Implementierung durch ein Konzept oder Design in einem Pflichtenheft dokumentiert. Anforderungen sollten laut Ebert daher nicht die Realisierung einer Funktion oder eines Nutzens beschreiben. (Ebert 2010, S. 21) Allerdings ist diese Formulierung von lösungsneutralen bzw. lösungsoffenen Anforderungen nicht in allen Fällen möglich. Tavakoli Kolagari et al. (2007, S. 135) stellen fest, dass, entgegen der Forderung in der Literatur nach lösungsneutralen Anforderungen, in der Praxis die Anforderungen bereits eine Lösung vorgeben, da es nicht möglich ist, zwischen Problem- und Lösungsraum eindeutig zu unterscheiden. So müssen z.B. in den zuvor in Kapitel 2.1.1 beschriebenen Kooperationen mit Entwicklungsdienstleistern oder bei der Vergabe von Black-Box-Teilen die Komponenten zur Integration in das Gesamtprodukt bestimmte Eigenschaften und Schnittstellen sowie Geometrien aufweisen. Diese sind als Anforderungen in einem Lastenheft an einen Komponentenlieferanten festgelegt und an dieser Stelle nicht lösungsneutral.

Mayer-Bachmann (2008, S. 15) hingegen schlägt zur Strukturierung einer großen Anzahl an Anforderungen eine Klassifizierung von Anforderung nach den Dimensionen: Ebene, Funktion und Projekt vor. In der Dimension Projekt unterschiedet Mayer-Bachmann **projektspezifische** und **projektübergreifende** Anforderungen. Basierend auf der Zugehörigkeit einer Anforderung zu einer bestimmten Produktfunktion werden die Anforderungen nach **Funktionen**, wie z.B. Federung oder Antrieb bei einem PKW, klassifiziert. Abschließend können Anforderungen auch nach ihrer Detaillierungsebene in Architektur, Domäne oder Komponente klassifiziert werden. (Mayer-Bachmann 2008, S. 15)

Eine an Detaillierungsebenen angelehnte Klassifizierung schlägt auch Ebert (2010, S. 24ff.) vor. Hier werden Anforderungen in Markt-, Produkt- und Komponentenanforderungen aus der Perspektive der verschiedenen am Produktentwicklungsprozess beteiligten Akteure klassifiziert. So stellen die **Marktanforderungen** die Kundensicht dar und sind in Kundensprache gehalten. Häufig synonym verwendete Begriffe sind Kundenanforderungen, Nutzer- oder Geschäftsanforderungen. Die aus den Marktanforderungen abgeleiteten Eigenschaften und Funktionen eines Produktes aus der Sicht der Realisierung werden hier als **Produktanforderungen** bezeichnet. Dazu gehören auch s.g. Systemanforderungen, Funktionen und Eigenschaften, die den Lösungsraum

und die Priorisierung festlegen. Bricht man diese Produktanforderungen weiter herunter und definiert die Anforderung an die einzelnen Bestandteile des Produktes, spricht Ebert (2010, S. 24ff.) von **Komponentenanforderungen**. Bei der Definition dieser Ebenen ist die Perspektive ausschlaggebend. So kann eine Komponentenanforderung eines OEMs aus Sicht eines Komponentenlieferanten dieses OEMs wiederum eine Marktanforderung darstellen, die der Kunde, der OEM, an das Produkt des Lieferanten stellt.

Anforderungsdokumentation

Für die Zusammenarbeit mit einer Vielzahl an Schnittstellenpartnern in der Entwicklungsarbeit eines Produktes (Kapitel 2.1.1) spielt die Anforderungsdokumentation eine wichtige Rolle. Innerhalb eines Unternehmens gibt es zahlreiche Schnittstellen, bei denen ein Anforderungsgeber, z.B. der After-Sales oder die Produktion, Anforderungen an die Entwicklung, den Anforderungsnehmer, weitergibt. In der Entwicklung übergibt z.B. ein Systemingenieur die auf ein Teilsystem oder Komponente spezifizierten Anforderungen an einen Komponentenentwickler. Um ein Bauteil durch einen Lieferanten entwickeln zu lassen, muss der OEM die Anforderungen an das Bauteil bezüglich Schnittstellen, Außenformen, Funktionen etc. genau definieren, um sie in das Gesamtprodukt integrieren zu können. Diese Anforderungen werden in der Vergabe des Entwicklungsauftrags in Form eines Komponentenlastenheftes an die Lieferanten übergeben, die basierend auf diesem Anforderungsdokument die Komponente entwickeln. Das Anforderungsdokument ist somit ein wichtiger Bestandteil und Vertragsdokument in einem hochkomplexen Entwicklungsprojekt eines Produktes mit einer großen Anzahl an Schnittstellen und Lieferanten.

Die in der VDI 2221 vorgeschlagene Dokumentationsform ist die **Anforderungsliste**, in der alle unternehmensexternen und -internen Anforderungen sowie eine formulierte Aufgabenstellung der Konstruktion enthalten sind. Sie ist das Ergebnis des ersten Arbeitsabschnitts des in der VDI 2221 vorgestellten Vorgehens beim Entwickeln und Konstruieren. In diesem ersten Abschnitt soll die Aufgabenstellung geklärt und detailliert werden. Dabei dient die Anforderungsliste als Basis für alle weiteren Arbeitsabschnitte im Entwicklungsvorgehen und muss auf den Stand der neuesten Erkenntnisse im Laufe des Entwicklungsprozesses fortlaufend aktualisiert werden. (VDI 2221, S. 9f.)

Das **Lastenheft** dient zur Anforderungsdokumentation und vertraglichen Regelung des Leistungs- und Lieferumfangs zwischen Auftraggeber und Auftragnehmer (DIN 69901-5, S. 9). Dabei wird das Lastenheft durch den Auftraggeber erstellt und enthält alle Anforderungen aus Anwendersicht sowie die Rahmenbedingungen. Dadurch dient es als Ausschreibungsunterlage und Vertragsdokument. (VDI/VDE 3694, S. 3; VDA 2007, S. 18)

*„Im Lastenheft wird definiert, **was** und **wofür** zu lösen ist und unter welchen Randbedingungen." (VDI/VDE 3694, S. 3)*

Der VDA (2007, S. 18) definiert zusätzlich das **Komponentenlastenheft**. Dabei kann ein Komponentenlastenheft ein System-, Modul- oder auch ein Bauteillastenheft sein. Basierend auf dem Komponentenlastenheft erfolgt die Anfrage bei einem Komponentenlieferanten. Durch die englische Übersetzung des Wortes Lastenheft in „specification" wird **Spezifikation** häufig als Synonym für das Lastenheft verwendet (vgl. Boutkova 2014, S. 13). Allerdings definiert die DIN 69901-5 die Übersetzung des Begriffs Lastenheft als „user specification" und des Wortes Pflichtenheft als „functional specification". Bei der Verwendung des Begriffs Spezifikation als Synonym besteht die Gefahr, die Abgrenzung zwischen Lastenheft und Pflichtenheft zu verlieren. Die Unterscheidung in Lasten- und Pflichtenheft ist allerdings im Englischen nicht üblich (Pohl 2007, S. 232).

Basierend auf dem Lastenheft wird durch den Auftragnehmer, z.B. dem Komponentenlieferanten, das **Pflichtenheft** erstellt. Hier beschreibt der Auftragnehmer die Realisierung der vom Auftraggeber im Lastenheft definierten Anforderungen. (DIN 69901-5, S. 10) Das Pflichtenheft referenziert die Anforderungen des Lastenheftes, *wie und womit* diese umgesetzt werden. Es enthält die Lösungsbeschreibung. Dabei werden vom Auftragnehmer die Anforderungen des Lastenheftes auf Widersprüche und Realisierbarkeit geprüft. (VDI/VDE 3694, S. 3; Ilie 2013, S. 52)

Sowohl Lastenheft als auch Pflichtenheft werden i.d.R. in den Dateiformaten .pdf oder .docx als Fließtext (vgl. Goldin et al. 2015, S. 30) geschrieben. Der Nachteil dieser Dokumentationsform ist, dass zur Anforderungsverwaltung keine zusätzlichen Informationen in Form von Attributen zu jeder einzelnen Anforderung gespeichert werden können. Attribute zur Gültigkeit, Historie, Quelle oder Priorität unterstützen das Änderungsmanagement und auch die Wiederverwendung von Anforderungen. (vgl. Hood et al. 2005, S. 103) Um eine größere Menge Anforderungen sinnvoll strukturieren zu können, ist eine Attribuierung der Anforderungen erforderlich. Daher bieten

sich insbesondere zur Verwaltung von einer großen Anforderungsmenge **Anforderungsdatenbanken**, auch Anforderungsmanagementsystem genannt, an. Hier werden die Anforderungen einzeln durch eine ID gekennzeichnet und als Einträge in einer umfassenden Datenbank erfasst. Es gibt zahlreiche Software zum Anforderungsmanagement, die Anforderungsdatenbanken enthalten wie z.B. Rational DOORS bzw. DOORS Next der Firma IBM, codeBeamer ALM von Intland, Dimensions RM von MicroFocus oder Polarion Requirements von Siemens (beispielhafte Aufzählung ohne Priorisierung).

Die Feststellung des VDA (2007, S. 12), dass früher eine Dokumentation der Anforderungen in Form einer Zeichnung und Stückliste ausgereicht hätte, heute allerdings zur Definition und Beschreibung der Anforderungen an eine einzelne elektronische, mechatronische oder mechanische Komponente ein ganzes Komponentenlastenheft erforderlich ist, zeigt die Entwicklung der Anforderungsmenge und Komplexität der erforderlichen Anforderungen.

Am Beispiel der Automobilindustrie sollen an dieser Stelle die Dimensionen einer großen Anforderungsmenge, in der Literatur als „very large scale RE“ bezeichnet (vgl. Regnell et al. 2008, S. 124; Leuser et al. 2010, S. 204), verdeutlicht werden, um die Herausforderungen für die Verwaltung und Strukturierung einer solchen Datenmenge zu veranschaulichen. Für das Produkt PKW werden Anforderungen auf drei verschiedenen Ebenen definiert: Fahrzeug, System und Komponente. Ein PKW besteht aus 100 verschiedenen Systemen und mehreren Hundert Komponentenlastenheften. Ein einzelnes Komponentenlastenheft kann dabei je nach Komplexität der jeweiligen Komponente (mechanisch, mechatronisch, elektronisch, Software) bis zu 50.000 Anforderungen enthalten. (Leuser et al. 2010, S. 204) Ein Lastenheft für ein Software-System umfasst 20.000 Seiten, das einer einzelnen Komponente bereits mehrere hundert Seiten. (Heumesser et al. 2004, S. 302)

Dabei lässt sich eine stetige Zunahme der Anforderungsmenge in der Automobilindustrie beobachten. So hat Kubica (2007, S. 3) die Zunahme der Anforderungen eines Steuergerätes untersucht und eine Steigerung von ursprünglich 50 Seiten (1990) auf ca. 3.100 Seiten in 2000 festgestellt. Eine aktuelle Untersuchung am Beispiel konventioneller Stoßdämpfer eines PKW zeigt, dass nicht nur bei mechatronischen oder elektronischen Komponenten wie dem Steuergerät die Anforderungsmenge gestiegen ist. Abbildung 2-3 zeigt, dass sich der Lastenheftumfang des Stoßdämpfers innerhalb von 10 Jahren verdreifacht hat: von 16 Seiten im Jahr 2004 auf 50 Seiten im Jahr 2014.

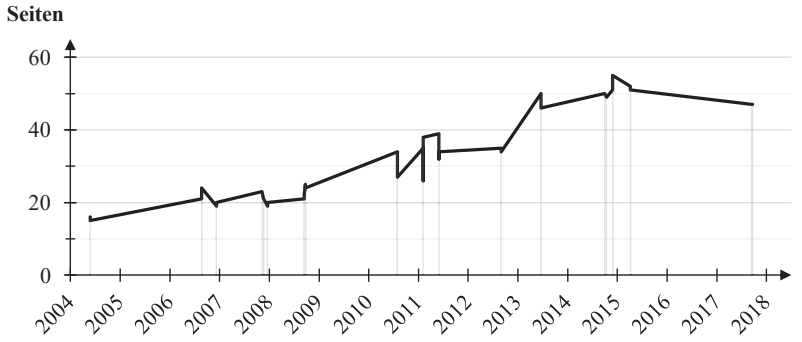


Abbildung 2-3: Entwicklung des Lastenheftumfangs eines PKW-Stoßdämpfers (Ritter et al. 2020, S. 116)

Anforderungen werden in den oben beschriebenen Anforderungsdokumenten und -datenbanken in **natürlicher Sprache** dokumentiert. Natürliche Sprache beschreibt dabei alle von Menschen gesprochenen Sprachen einschließlich Gebärdensprache. Sie grenzt sich gegenüber formalen oder künstlichen Sprachen, wie z.B. einer Programmiersprache, ab. Ein wesentlicher Unterschied ist die Mehrdeutigkeit von einzelnen Worten in der natürlichen Sprache, deren jeweilige Bedeutung erst im Kontext des Wortes oder des Satzes entsteht. Des Weiteren kann derselbe Inhalt eines Satzes in natürlicher Sprache auf unterschiedlichste Art und Weise ausgedrückt werden. In der natürlichen Sprache wird durch die Verwendung von Sprichwörtern oder Metaphern der eigentliche Inhalt der Sprache verdeckt. Obwohl diese Dokumentationsform eine Reihe von Nachteilen wie z.B. die Mehrdeutigkeit, mangelnde Struktur und Formalität mit sich bringt (Cybulski et al. 2000, S. 190), ist die Anforderungsdokumentation in natürlicher Sprache die am weitesten verbreitete Dokumentationsform von der Automobilindustrie bis hin zur Softwareentwicklung (vgl. Vogelsang et al. 2016, S. 162; Arora et al. 2015, S. 6; Körner 2014, S. 36; Sommerville 2012, S. 128; Pohl 2007, S. 229; Mich et al. 2004, S. 48; Gleich et al. 2010, S. 218; Ott 2012, S. 291; Cybulski et al. 2000, S. 190). Diese Dokumentationsform als natürlichsprachliche Anforderungen erschwert die Verarbeitung und die Automatisierung dieser.

Weitere Dokumentationsformen von Anforderungen sind UML-Sequenzdiagramme, Zustandsdiagramme oder mathematische Spezifikationen. **UML** steht für Unified Modelling Language und ist eine Form der grafischen Notation. Sie dient zur Beschreibung von Prozessen in Softwaresystemen. Dabei existieren verschiedene Diagrammtypen mit unterschiedlichen Inhalt. **Aktivitätsdiagramme** oder **Zustandsdiagramme** können den Informationsfluss in einem System abbilden, die Schnittstellen und das

Verhalten eines Systems mit einem Anwender wird als **Use-Case-Diagramm** dargestellt. (vgl. Rupp 2014, S. 189ff.). Zusätzlich gibt es auch noch **Klassen-** und **Sequenzdiagramme**. (vgl. Sommerville 2012, S. 128; Rupp 2014, S. 197ff.). Weitere Dokumentationsformen für das geforderte Verhalten von Systemen als Ablaufdiagramme können Ereignisgesteuerte Prozessketten (EPK) oder die Business Process Model und Notation (BPMN) sein (Rupp 2014, S. 172ff.).

Anforderungsmanagement

Für den Umgang mit Anforderungen existieren in der Literatur unterschiedliche Bezeichnungen und Definitionen: Anforderungsmanagement (AM), Anforderungsentwicklung, Requirements Engineering (RE), Requirements Management (RM) und die kombinierte Abkürzung Requirements Management and Engineering (RM&E). Sie unterscheiden sich je nach Autor hinsichtlich ihrer Hierarchie zueinander als auch in den Anforderungsaktivitäten und Aufgaben, die der jeweilige Begriff umfasst (Abbildung 2-4).

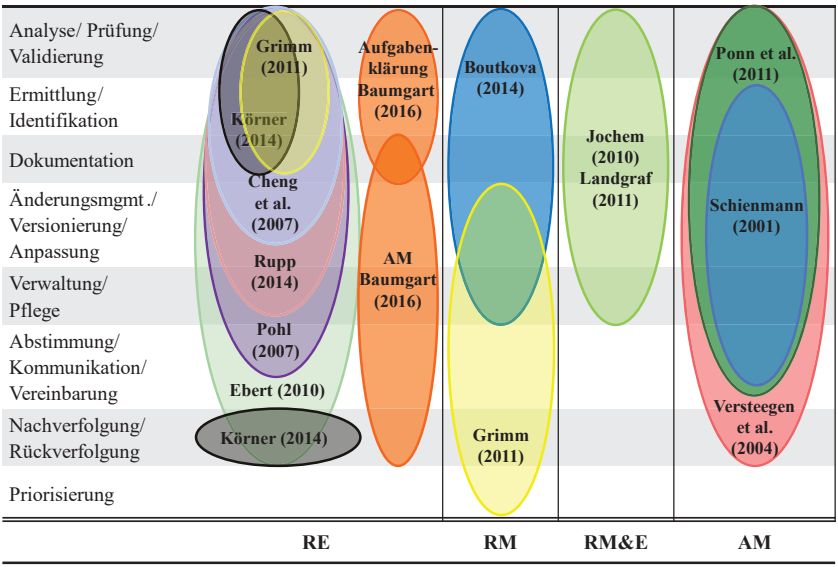


Abbildung 2-4: Unterschiedliches Begriffsverständnis im Bereich Anforderungen

Der Begriff **Requirements Engineering (RE)** stammt aus der Disziplin des Software-Engineering aus dem englischsprachigen Raum. Dementsprechend wird RE in der Literatur des Software Engineerings als Oberbegriff für alle mit Anforderungen in Zusammenhang stehenden Aktivitäten betrachtet (Cheng et al. 2007, S. 286; Rupp 2013,

S. 13f.; Ebert 2010, S. 32-35; Pohl 2007, S. 44ff.; Grimm 2011, S. 38). Baumgart (2016, S. 426) verdeutlicht das Verständnis des RE als einen Oberbegriff durch die Unterscheidung der beiden Unterdisziplinen Anforderungsmanagement und Aufgabenklärung. Die **Aufgabenklärung** beinhaltet die Erhebung, Analyse und Dokumentation von Anforderungen, wohingegen das **Anforderungsmanagement** die Freigabe, Versionierung, Änderung und Rückverfolgung von Anforderungen einschließt.

Im deutschsprachigen Raum hat sich der Begriff des **Anforderungsmanagements (AM)** verbreitet. Er wird teils als gesamtheitlicher Oberbegriff für alle Aufgaben in Zusammenhang mit Anforderungen analog zum Begriff RE verwendet, teils schließt er die Aufgaben der Ermittlung, Analyse und Dokumentation aus. (Schienmann 2001, S. 32; Ponn et al. 2011, S. 35; Versteegen et al. 2004, S. 4f.). Die englische Übersetzung als **Requirements Management (RM)** ist in der Literatur eher selten zu finden (Boutkova 2014, S. 3; Grimm 2011, S. 38).

Unter der kombinierte Abkürzung **RM&E** werden beide Disziplinen von Jochem (2010, S. 130) und Landgraf (2011, S. 18f.) definiert. Sie verwenden die Abkürzung **RM&E** als Klammer für alle Aufgaben und Aktivitäten in Zusammenhang mit Anforderungen. Dabei präzisiert Landgraf durch die Unterteilung und Unterscheidung in Requirements Management (analog Anforderungsmanagement) und Requirements Engineering (analog Anforderungstechnik). Das Requirements Management wird hier mit den Aufgaben der *„[…] Anforderungskontrolle, -verfolgung und -verwaltung sowie die Steuerung und Kontrolle des Prozesses und der Zielsetzungen definiert.“* (Landgraf 2011, S. 18), wohingegen das Requirements Engineering die Umsetzung des Prozesses durch die Erhebung, Analyse, Spezifikation, Dokumentation, Prüfung und Änderung von Anforderungen umfasst.

In Abbildung 2-4 ist dargestellt, wie unterschiedlich das Begriffsverständnis in der Literatur ausgeprägt ist. Der Begriff RE beschreibt je nach Autor nur die Aufgaben der Ermittlung, Analyse und Dokumentation bis hin zum vollen Umfang aller Aktivitäten eines Entwicklungsprojektes in Zusammenhang mit Anforderungen. Ähnlich unterschiedlich ist das Begriffsverständnis des AM. In dieser Arbeit wird der Begriff des Anforderungsmanagements zur Beschreibung aller Prozesse und Aktivitäten zur Unterstützung der Anforderungsentwicklung verwendet. Dabei umfasst das Anforderungsmanagement die Verwaltung, Rückverfolgung, Strukturierung und Versionierung von Anforderungen. Das Anforderungsmanagement gestaltet den Prozess und gibt der Anforderungsentwicklung darüber Rahmenbedingungen vor, innerhalb denen

die Anforderungsentwicklung Anforderungen identifiziert, analysiert, spezifiziert, dokumentiert, validiert und vereinbart. Die Identifizierung der Anforderungen der Stakeholder ist die Grundlage zur Aufgabenklärung in einem Entwicklungsprojekt. Danach folgt die Spezifikation als Detaillierung der Kundenanforderungen auf den betrachteten Entwicklungsumfang bis hin zur Detaillierung auf Komponentenebene. Die Kontrolle und Validierung der Umsetzbarkeit und Realisierung der Anforderungen sowie die Definition von Prüffällen ist ebenfalls Bestandteil der Anforderungsentwicklung.

2.1.3 Wiederverwendung von Anforderungen

Wie im vorherigen Abschnitt beschrieben, hat sich die Anforderungsmenge in den Entwicklungsprojekten vervielfacht. Gleichzeitig steigt der Druck in der Industrie, Produkte immer schneller in den Markt zu bringen und gleichzeitig die eigenen Kosten zu senken. Um in diesem Spannungsdreieck (Atkinson 1999, S. 337f.) die steigenden Anforderungsmenge zu bewältigen, werden in Entwicklungsprojekten Anforderungen aus Vorgängerprojekten, anderen Baureihen oder ähnlichen Produkten wiederverwendet. 80% der Anforderungen sind für ähnliche Systeme gleich (Sommerville et al. 2006, S. 106). Insbesondere für die Automobilindustrie mit ihrer großen Anforderungsmenge bietet sich die Wiederverwendung von Anforderungen an, da im Entwicklungsprozess eines neuen Fahrzeugs überwiegend Anpassungsentwicklungen durchgeführt werden (Jörg 2005, S. 106; Mayer-Bachmann 2008, S. 2). Nur in wenigen Fällen werden Komponenten vollständig neu entwickelt, wie z.B. bei innovativen Systemen des autonomen Fahrens oder neuen Bauteilen für elektrische Antriebe.

Die Wiederverwendung von Anforderungen bietet eine Reihe von Vorteilen: die Anforderungsentwicklung spart sich sehr viel Zeit und Aufwand zur Ermittlung der Anforderungen der unterschiedlichen Stakeholder, wenn sie auf bereits erfasste Anforderungen zurückgreift. Laut Rupp (2014, S. 460) können bis 80% der Anforderungen für ein neues Projekt wiederverwendet werden und langfristig im Unternehmen die Zeit zum Markteintritt reduziert werden. Gleichzeitig sind die vorhandenen Anforderungen bereits geprüft und validiert, sodass man sich auch für diesen Schritt Zeit und Aufwand spart und durch ausgereifte Lösungen auch die Produktqualität steigert. (Darimont et al. 2017, S. 456; Rupp 2014, S. 461; Ilie 2013, S. 72; Chernak 2012, S. 48; Cybulski et al. 2000, S. 207) Auf der anderen Seite birgt die Wiederverwendung von Anforderungen auch Risiken. Chernak (2012, S. 48) hat in einer weltweiten Studie IT Experten zu ihren Hürden in der Wiederverwendung von Anforderungen befragt. Die drei am

häufigsten genannten Hürden sind, dass

1. die Anforderungen aus Vorgängerprojekten unvollständig sind oder gar nicht existieren,
2. die vorhandenen Anforderungen schlecht strukturiert sind, sodass nur schwer geeignete Anforderungen zur Wiederverwendung identifiziert werden können,
3. die vorhandenen Anforderungen nicht aktualisiert werden.

Auch Rupp (2014, S. 112) zählt die unzureichende Qualität alter Anforderungen sowie die Übernahme von fehlerhaften Anforderungen durch unkontrolliertes Wiederverwenden zu den größten Risiken in der Wiederverwendung von Anforderungen. Solche fehlerhaften, unvollständigen, unstrukturierten und veraltete Anforderungen können durch Wiederverwendung über viele Entwicklungsprojekte hinweg weiterverwendet werden und so zu schlechter Qualität und zusätzlichen Entwicklungskosten führen.

Die genannten Risiken zeigen, dass nicht jede Anforderung für eine Wiederverwendung geeignet ist. Am geeignetsten sind nach Rupp (2014, S. 462) Anforderungen mit hohem Spezifikationslevel, die unabhängig von einem Ablauf oder Technologie sind sowie bei einem begrenzter Produktportfolio des Unternehmens. Dabei sind nicht alle der genannten Merkmale durch eine Anforderung zu erfüllen, sondern im Einzelfall zu prüfen. Die Identifizierung geeigneter Wiederverwendungskandidaten ist je nach angewandter Methode gar nicht bis stark ausgeprägt. Dennoch verspricht die Wiederverwendung von Anforderungen ein hohes Potential zur Kosten- und Zeitersparnis, sodass in der Literatur zahlreiche unterschiedliche Methoden existieren, von denen einige an dieser Stelle vorgestellt werden.

Copy-and-Paste, Templates und Pattern

Das in der Praxis am häufigsten eingesetzte Verfahren zur Wiederverwendung ist das **Copy-and-Paste** Verfahren (>50%), das reine Kopieren einzelner Anforderungen bis hin zu ganzen Anforderungsdokumenten aus einem Vorgängerprojekt oder einem ähnlichen Produkt (Palomares et al. 2014, S. 303). Bei diesem Verfahren ist das Identifizieren geeigneter Anforderungen gar nicht ausgeprägt, insbesondere beim Kopieren ganzer Dokumente. Die Risiken bei diesem Vorgehen liegen nach Monzon (2008, S. 223) im Versionsmanagement, da nicht sichergestellt werden kann, dass die aktuellste Version der Anforderung kopiert wird, und in der Aktualität in Bezug auf veränderte Kundenbedürfnissen und technischen Einschränkungen. Zusätzlich läuft man Gefahr, fehlerhafte oder qualitativ schlechte Anforderungen ungeprüft zu übernehmen und über viele Entwicklungsprojekte hinweg wiederzuverwenden. Das in der Praxis so häufig angewandte Copy-and-Paste Verfahren könnte auch erklären, warum in der

Studie von Chernak 69% der Teilnehmer die Frage, wie in der Praxis Anforderungen zur Wiederverwendung identifiziert werden, gar nicht beantwortet haben und 15% eine informelle Vorgehensweise angaben (Chernak 2012, S. 50).

Das nach dem Copy-and-Paste zweithäufigste eingesetzte Verfahren ist die Verwendung von **Templates** (~40%) (Palomares et al. 2014, S. 303). Ein Template ist eine Schablone, die eine konkrete Struktur oder Bauplan der Anforderung vorgibt. Rupp (2014, S. 219) zeigt in ihrer Sammlung MASTER Satzbauschablonen für die unterschiedlichen Anforderungsarten auf (Abbildung 2-5). Durch solche Templates werden die Anforderungen einheitlich formuliert und die Qualität der Anforderungen verbessert sich. Allerdings können auch unter Verwendung solcher Templates keine zusätzlichen Informationen zur Wiederverwendung dokumentiert werden.

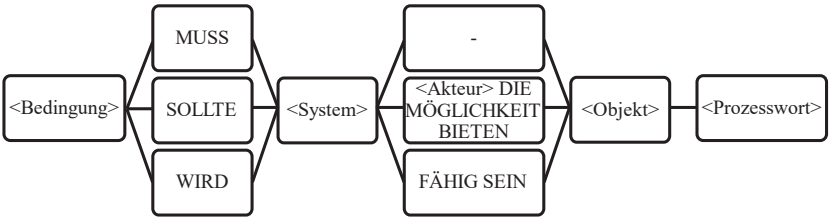


Abbildung 2-5: Template - Satzbauschablone für eine funktionale Anforderung (Rupp 2014, S. 219)

Weiter gehen s.g. **Pattern** (Abbildung 2-6). Solche Mustervorlagen geben in der Anforderungsdokumentation eine spezifische Struktur der Anforderung mit definierten Attributen wie Ziel, Rahmenbedingungen, Abhängigkeiten, Verknüpfungen etc. vor.

Beschreibung	Diese Mustervorlage beschreibt die Notwendigkeit einer Softwarelösung zur Information des Anwenders über Fehler.
Kommentar	Die Warnung soll in dem Moment ausgegeben werden, in dem der Fehler auftritt.
Ziel	Den Anwender vor Fehlern warnen
Autor	Max Mustermann
Quellen	Anforderungsbuch CIT Fachliteratur
Schlagwörter	Warnung, Fehler, Absturz
Abhängigkeiten	Voraussetzung: Fehlerberichte

Abbildung 2-6: Beispiel für ein Anforderungspattern für Software (Renault et al. 2009, S. 84)

Des Weiteren stellen solche Vorlagen sicher, dass die wichtigsten Attribute bzw. Zusatzinformationen zu einer Anforderung dokumentiert werden. Ein Beispiel für eine solche Mustervorlage geben Renault et al. (2009, S. 84), in der auch Abhängigkeiten der Anforderung von anderen Anforderungen oder Parametern aufgezeigt werden sowie Versionsstand und den Autor. Diese Form der Wiederverwendung wird allerdings von den wenigsten Anwendern (13%) in der Praxis verwendet (Palomares et al. 2014, S. 303).

Anforderungsbibliothek – gemeinsame und variable Anforderungen

In der Literatur wird häufig zur Wiederverwendung von Anforderungen die lösungsneutrale Dokumentation der Anforderungen in **Anforderungsbibliotheken** beschrieben (Mahmoud et al. 2010, S. 330; Jörg 2005, S. 106f.) und ist laut Tavakoli Kolagari et al. (2007, S. 129) ein auch in der Praxis bei der Mercedes-Benz Group AG angewendetes Verfahren. Das Prinzip einer Anforderungsbibliothek ist die projektübergreifende Verwaltung der Anforderungen in einer gemeinsamen Datenbank, aus der für die einzelnen Projekte die relevanten Anforderungen konfiguriert werden können.

Grundlegend dafür ist die Unterscheidung in **gemeinsame** und **variable** Anforderungen (Ilie 2013, S. 74). Jörg (2005, S. 71) unterscheidet in seiner Anforderungsbibliothek in ausgeprägte und nicht ausgeprägte Anforderungen, Karatas et al. (2014, S. 832) gruppiert die Anforderungen in **gemeinsame, variable und produktspezifische Anforderungen**. Die Unterteilung von Karatas et al. (2014) wird sinngemäß für diese Arbeit übernommen, wobei die variablen als ausgeprägte Anforderungen und die gemeinsamen als projektübergreifende Anforderungen bezeichnet werden (Kapitel 3). Eine variable Anforderung unterscheidet sich von Produkt zu Produkt. Die Differenzierung erfolgt in der Regel durch Parameter, innerhalb dessen ein projektspezifischer Wert definiert wird (Mannion et al. 2008, S. 66-68). Gemeinsame Anforderungen sind projektübergreifend gleich. Dehlinger et al. (2008, S. 208) beschreibt eine Commonality and Variability Analysis (CVA) zur Dokumentation von gemeinsamen und variablen Anforderungen in Software-Produktlinien. Allerdings stellt Karatas et al. (2014, S. 833) fest, dass die größte Hürde bei der Wiederverwendung von Anforderung in dem Mangel an effizienten Methoden zur Kommunalitäts- /Variabilitätsanalyse liegt als auch diese effizient zu dokumentieren.

Im Folgenden wird der Begriff Anforderungsbibliothek zur Beschreibung einer umfassende Anforderungsdatenbank aller Produktentwicklungsprojekte mit dem Ziel der Wiederverwendung von Anforderungen verstanden. Ein **Anforderungskatalog** hingegen kann synonym zu einer Anforderungsliste auch nur die Anforderungen eines

Projektes darstellen (vgl. Burghardt 2013, S. 36f.). Weitere Ansätze wurden zur Darstellung des Stands der Technik anhand der Kriterien Wiederverwendung von Anforderungen, Technische Anforderungen (nicht Kundenanforderungen) und dem Objektbereich mechanischer Komponenten bewertet. Die ausführliche Auflistung der betrachteten Ansätze und deren Bewertung ist in Anhang A dargestellt. Die drei Methoden mit der höchsten Punktzahl werden an dieser Stelle noch einmal ausführlicher erläutert.

Tavakoli Kolagari et al. (2007)

Der Ansatz, den Tavakoli Kolagari et al. (2007) zur Wiederverwendung beschreiben, fokussiert speziell die Verwendung einer Anforderungsbibliothek für Software-Produktlinien von ECU (Steuergeräten) in der Automobilindustrie. Die beschriebene Anforderungsbibliothek soll dabei architekturunabhängige Module enthalten, die später zur Konfiguration einer baureihenspezfischen Systemspezifikation dienen. Dies wird durch die Unterscheidung einer allgemeinen und einer technischen Ebene realisiert. In beiden Ebenen werden die Anforderungen unabhängig einer Systemarchitektur beschrieben, indem weder ECU Zuordnungen noch Signaladressierungen beschrieben werden. Die allgemeine Ebene der Anforderungsbibliothek beschreibt nur die Funktionalität und die erforderlichen Parameter und Signale. Die meisten Entscheidungen auf dieser Ebene betreffen die Auswahl von Alternativen und Optionen. Erst auf der technischen Ebene werden Details wie Fehlerverhalten, Wertebereiche etc. hinzugefügt. Hier werden durch die Verwendung von Parametern die Anforderungen an die unterschiedlichen Baureihen angepasst.

Die Variabilität der Anforderungen wird in einem Variabilitätsmodell durch Definition von Entitäten und Kardinalitäten umgesetzt. Die Strukturierung des Variabilitätsmodells erfolgt anhand von Variationspunkten. Der Prozess der Wiederverwendung zur baureihenspezifischen Konfiguration eines Lastenheftes wird dabei durch die Auswahl von gewünschten Funktionen der Software realisiert, die mit einer parallelen, automatischen Vorauswahl der in der Funktion enthaltenen Anforderungen erfolgt. Durch weitere nicht weiter erläuterte Entscheidungen wird die Vorauswahl durch den Requirements Engineer vervollständigt.

Der hier beschriebene Ansatz wurde speziell für die Anforderungen an Steuergerätesoftware entwickelt. Es wird vor allem die Struktur der Anforderungsbibliothek und der Prozess zur Auswahl der produktspezifischen Anforderungen beschrieben. Der Ansatz enthält keine Beschreibung zum Vorgehen für den Aufbau der Bibliothek und lässt offen, ob die Bibliothek von Grund auf mit neu geschriebenen Anforderungen

aufgebaut wurde oder ob dafür manuell oder automatisiert bereits bestehende Lastenhefte analysiert und die Anforderungen in die Bibliothek übernommen wurden.

Mannion et al (2008), (1999) - MRAM

Die von Mannion et al. vorgestellte Method for Requirements Authoring Management (MRAM) verwaltet Anforderungen für Produktlinien in einer Baumstruktur. In dieser Baumstruktur wird die Eltern-Kind-Abhängigkeiten zwischen den Anforderungen dargestellt. Das Produktlinienmodell für Anforderungen besteht zum einem aus einem Wörterbuch der jeweiligen Domäne, zum anderen aus einer Menge an nummerierten, natürlichsprachigen, wiederverwendbaren Anforderungen aller existierenden Produkte der Produktlinie. In einem Metamodell in der SysML Modellierungssprache werden zusätzliche Informationen zu einer Anforderung abgebildet, dass die anwendungsspezifische Auswahl der Anforderungen unterstützen soll. Hier werden ebenfalls die Beziehungen zwischen den Anforderungen als *„[…] derive, satisfy, verify, refine, trace, containment, and copy." (Mannion et al. 2008, S. 64)* abgebildet.

Grundsätzlich unterscheiden Mannion et al. die Gruppen *„[..] common, variable, unavailable." (2008, S. 66).* Als allgemeine (common) Anforderungen beschreiben Mannion et al. Anforderungen, die in allen oder einigen Produkten der Produktlinie vorkommen und für neue Produkte dieser Produktlinie wiederverwendet werden können. Währenddessen unterscheiden sich variable Anforderungen von Produkt zu Produkt. Hier definieren Mannion et al. Diskriminanten und Parameter in den Anforderungen. Während Parameter die quantitative Variabilität von Anforderungen abbilden, stellen Diskriminanten qualitative Unterschiede dar. Gleichzeitig schaffen Mannion et al. durch die Definition von nicht verfügbaren (unavailable) Anforderungen die Möglichkeit, spezifische Anforderungen aufgrund von Marktnischen oder Alter für die Wiederverwendung auszuschließen.

Der Aufbau eines solchen Produktlinienmodells für Anforderungen erfordert nach Mannion et al. insbesondere die Fähigkeit, über ein Produkt hinaus Kommunalitäten und Variabilität in den Anforderungen zu identifizieren als auch das Expertenwissen zur Abstraktion und ein breites Fachwissen in der jeweiligen Domäne.

Hauksdottir et al. (2012) - Adjustable requirement reuse

Hauksdottir et al. (2012) stellen in ihrem Ansatz die Wiederverwendung von Anforderungen bei der Danfoss GmbH vor. Hier wird ein s.g. company repository als unternehmensweites Anforderungsverzeichnis aufgebaut. Für die Wiederverwendung der Anforderungen in einzelnen Entwicklungsprojekten werden zunächst die Anforderun-

gen von dem company repository in ein Projekt repository abgebildet und anschließend Teile der Anforderungen auf das jeweilige Projekt angepasst. Diese Anpassungen können sowohl Verbesserungen in der Formulierung der Anforderungen als auch die Anpassung spezifischer Werte und Parameter sowie zusätzliche Kommentare sein.

Um diese projektspezifische Anpassung zu unterstützen, werden ein allgemeiner und ein veränderlicher Teil der Anforderungen definiert. Die veränderlichen Teile werden als Variationsstellen farblich hervorgehoben, um den Anwender darauf hinzuweisen, welcher Teil verändert werden kann. Der allgemeine Teil der Anforderung darf durch den Anwender nicht verändert werden. Nachteil dieses Vorgehens der Wiederverwendung besteht darin, dass die Anwender frei darin sind, wie der veränderliche Teil angepasst wird. Hauksdottir et al. (2012) beschreiben den Aufbau des company repository als manuellen Prozess, in dem initial das Verzeichnis mit den Anforderungen eines ersten Entwicklungsprojektes befüllt wurde. Diese initialen Anforderungen wurden mit dem Fokus auf das spezifische Entwicklungsprojekt dokumentiert und nach der Akzeptanz des Lastenheftes ohne Änderungen in das company repository übertragen. Das zweite Entwicklungsprojekt wurde mit den initialen Anforderungen aus dem company repository begonnen. Diese Anforderungen wurden durch die Anwender an das spezifische Entwicklungsprojekt angepasst. Nachdem auch das Lastenheft des zweiten Projektes akzeptiert wurde, wurden die Anforderungen durch einen Anforderungsspezialisten und einem Entwickler des zweiten Projektes hinsichtlich Konsolidierung und Generalisierung als Unternehmensanforderungen überprüft. Basierend auf dieser Überprüfung wurde das company repository wieder aktualisiert.

Im Fazit von Hauksdottir et al. wird deutlich, wie viel Zeit in der Entwicklung durch eine Wiederverwendung von Anforderungen gespart werden kann. Während das Lastenheft im ersten Projekt nach 21 Wochen fertig war und ungefähr 6.000 Arbeitsstunden benötigt hat, war das Lastenheft im zweiten Projekt bereits nach 15 Wochen und 4.000 Arbeitsstunden abgeschlossen. Allerdings wird im Fazit auch deutlich, dass dieser manuelle Ansatz vor allem kleinere Anforderungsmengen betrachtet. Für das erste Projekt wurden 530 Anforderungen, im zweiten Projekt 480 Anforderungen dokumentiert. Der Prozess zur Identifizierung der Variationsstellen ist manuell. Müssen mehrere tausend Anforderungen aus 15 Projekten oder mehr in eine Datenbank übertragen und die Variationsstellen identifiziert werden, bietet dieser Ansatz zu wenig Automatisierung.

Wie Monzon (2008, S. 223) bemerkt, sind die Wiederverwendung von Anforderungen und das Variantenmanagement von Anforderungen eng verwandte Themen, sodass im

folgenden Kapitel zunächst auf das Variantenmanagement und die Definition einer Variante eingegangen werden soll. Innerhalb der Erläuterung der Methoden des Variantenmanagements im Anforderungsmanagement werden die Methoden von Monzon (2008) und Boutkova (2014) erläutert und an dieser Stelle darauf verwiesen, dass es sich hierbei auch um Methoden zur Wiederverwendung von Anforderungen handelt.

2.2 Variantenmanagement und Variantentreiber

Der Fokus der vorliegenden Arbeit sind variantenverursachende Anforderungen in Komponentenlastenheften. Daher sollen in diesem Kapitel die Begriffe Variante, Variantenvielfalt sowie Variantenmanagement definiert und von dem Begriff der Komplexität abgegrenzt werden. Darüber hinaus werden die Strategien des Variantenmanagements, Variantenreduzierung und Variantenbeherrschung vorgestellt. Abschließend wird auf die in der Literatur beschriebenen Variantentreiber eingegangen sowie mögliche Methoden zu deren Identifizierung.

2.2.1 Varianten, Komplexität und Variantenmanagement

Variante

In der Literatur wird die Herausforderung der steigenden Variantenvielfalt für die Industrie seit Jahrzehnten thematisiert (Schmelzer 1992, S. 141f.; Jeschke 1997; Gembrys 1998; Firchau 2003; Schuh et al. 2016). Um genau zu verstehen, wo die Ursachen der Variantenvielfalt liegen und wie diese in Unternehmen beherrscht werden können, muss ein gemeinsames Verständnis geschaffen werden, was unter einer **Variante** verstanden wird. Eine häufig zitierte Definition des Begriffs Variante ist die der DIN 199-2 als *"Gegenstände ähnlicher Form und/oder Funktion mit in der Regel hohem Anteil identischer Gruppen oder Teile"*. Diese Definition bezieht sich somit auf ein zusammengesetztes Produkt. Ebenfalls definiert Lingnau (1994, S. 20) die Variante eines Produktes über die Vergleichbarkeit hinsichtlich Eigenschaften, Funktionen oder Arbeitsprinzipien. Dabei führt er weiter aus, dass Varianten Ähnlichkeit in mindestens einem der Merkmale Geometrie, Material oder Technologie aufweisen (Lingnau 1994, S. 24). Auch Ehrlenspiel et al. (2014, S. 294) definieren die Variante hinsichtlich ihrer Ähnlichkeit in der Gestalt oder Funktion. Zur Unterscheidung der Varianten eines Erzeugnisses weisen Varianten eine unterschiedliche Identifikation über eine Artikelnummer, Sachnummer, Teilenummer oder Materialnummer auf (Zimmermann 1988, S. 1f.). Im Gegensatz hierzu definiert Heina (1999, S. 5) eine Variante über den Unterschied als ein Produkt, dessen Ausprägung von mindestens einem Merkmal sich von der Grundversion des Produktes unterscheidet. Franke et al. (2002, S. 12) teilen die Definition hinsichtlich des Unterschieds in mindestens einer Eigenschaft, beziehen den Begriff Variante allerdings auf Technische Systeme, die den gleichen Zweck erfüllen.

Die Variante unterscheidet sich von der **Version** des technischen Systems, da die Version den Stand des Systems in seinem Lebenszyklus mit einer zeitlichen und örtlichen

Gültigkeit beschreibt (vgl. Gebhardt et al. 2016, S. 112). Somit kann eine Variante über ihren Lebenszyklus verschiedene Versionen aufweisen, eine Version aber nie mehrere Varianten.

Die bisher vorgestellten Definitionen des Begriffs Variante beziehen sich auf Produkte oder auf technische Systeme, die aus mehreren Baugruppen und Bauteilen bestehen. In der vorliegenden Arbeit geht es insbesondere um Varianten auf Komponentenebene und somit um Varianten von Werkstücken bzw. Bauteilen. Diese Bauteile erfüllen die gleiche Grundfunktion, weisen aber unterschiedliche Maße, Formen, Materialien und zusätzliche Funktionen auf. Zimmermann (1988, S. 1f.) unterscheidet bei Varianten auf Bauteilebene geometrische sowie technologische Varianten und ergänzt diese um Lösungsvarianten, die ähnliche Funktionen erfüllen oder diese auf verschiedene Weise realisieren. Hinsichtlich technologischer Kriterien unterscheiden Franke et al. (2002, S. 14) Geometrie-, Material-, Technologie- und Prozessvarianten sowie bei der Frage nach der Ursache in Herstellerspezifische oder Kundenspezifische Variante. In dieser Arbeit sollen **Varianten** sowohl als Produkt zusammengesetzt aus mehreren Bauteilen als auch als einzelne Bauteile verstanden werden, die dieselbe Grundfunktion erfüllen, sich in mindestens einem Merkmal voneinander unterscheiden und durch eine Sachnummer identifiziert werden können.

Weisen Bauteile mit gleicher Grundfunktion identische Merkmale auf, gelten sie als Gleichteile und stellen keine Variante dar. Bauteile oder Komponente werden in der Literatur als **Gleichteile**, engl. Carry Over Parts (COP), bezeichnet, wenn sie standardisiert ohne Veränderungen oder Anpassungen mehrmals in einem Produkt oder in mehreren Produkten und Produktfamilien vorkommen (vgl. Gebhardt et al. 2016, S. 126; Böttrich 2015, S. 32). Sie bieten durch ihre Mehrfachverwendung das Potential, die Variantenvielfalt und somit die Komplexität einzuschränken. Jeschke (1997, S. 28) und Böttrich (2015, S. 32) unterscheiden hierbei noch den Fall, dass ein bereits früher entwickeltes Bauteil aus anderen Produkten Verwendung in einem anderen Produkt gleicher oder verschiedener Produktart wiederholt verwendet wird und bezeichnen diese als **Wiederholteil**, welches nicht mit der Wiederverwendung eines gebrauchten Bauteils verwechselt werden darf.

Variantenvielfalt

Die Herausforderung im Umgang mit Varianten entsteht durch die Vielfalt, in denen Varianten auftreten. Der Begriff der **Variantenvielfalt** beschreibt dabei nicht nur die Anzahl, sondern auch die Unterschiedlichkeit der Varianten zueinander. Da aber die Unterschiedlichkeit der Varianten häufig nicht quantitativ bestimmt werden könne,

werde die Bestimmung der Variantenvielfalt in der Praxis daher auf die Anzahl der Varianten reduziert (Gembrys 1998, S. 6). Allerdings existieren in der Literatur auch einige Ansätze, um die Unterschiedlichkeit der Varianten mithilfe von Formeln als Standardisierungsgrad und Differenzierungsgrad zu berechnen (Bauer 2016, S. 122). Weitere Ansätze sind die VMEA-Variant Modes and Effects Analysis (Caesar 1991), der Grad der Kommunalität (Collier 1981, S. 86f.) und der prozentualer Kommunalitätsindex (Siddique et al. 1998, S. 7ff.). Dabei bezeichnet die **Kommunalität** die gemeinsame Nutzung von Bauteilen in verschiedenen Produkten (Dellanoi 2006, S. 30). Die Unterscheidung in externe und interne Variantenvielfalt bezieht sich entweder auf die vom Kunden wahrgenommenen, am Markt angebotenen Produktvarianten (**externe Variantenvielfalt**) oder die innerhalb eines Unternehmens benötigten Bauteil- und Prozessvarianten, die zur Realisierung der Produktvarianten gehandhabt werden müssen (**interne Variantenvielfalt**). (Kesper 2012, S. 26f.; Gebhardt et al. 2016, S. 114)

Komplexität

Von der Variantenvielfalt ist der Begriff der **Komplexität** zu differenzieren. Komplexität umfasst dabei nicht allein die Anzahl der Bauteile bzw. Elemente eines Systems (Varietät), sondern auch deren Beziehungen und Wechselwirkungen untereinander (Konnektivität) sowie die zeitliche Veränderlichkeit und somit die unterschiedlichen Zustände, die die Elemente annehmen können (Dynamik). (Patzak 1982, S. 23; Ulrich et al. 1991, S. 58; vgl. Schuh 2005, S. 5; Dellanoi 2006, S. 26)

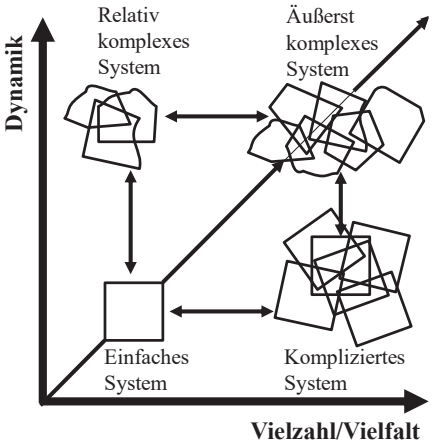


Abbildung 2-7: Darstellung von Komplexität in den Dimensionen Vielfalt und Dynamik (Ulrich et al. 1991, S. 61)

Zur Darstellung der Komplexität schlagen Ulrich et al. (1991, S. 61) eine Matrix (Abbildung 2-7) mit den Dimensionen Dynamik und Vielfalt vor, die den Grad der Komplexität bemessen sollen. Dabei stellt ein System mit geringer Dynamik und geringer Vielfalt ein einfaches System dar. Die Variantenvielfalt ist somit ein Bestandteil der Komplexität in der Dimension Vielzahl /Vielfalt. In Bezug auf ein Unternehmen verursachen eine wachsende Produktvielfalt und Variantenvielfalt eine zunehmende Komplexität der notwendigen Unternehmensprozesse. Betrachtet man das Produkt an sich, steigt die Komplexität durch die Vielfalt einzelner Bauteile zur Individualisierung, aber vor allem durch Innovation und Vernetzung einzelner Komponenten innerhalb eines Produktes (vgl. Clark et al. 1992, S. 132–133). Durch die Vernetzung der Komponenten innerhalb eines Produktes, was sich in einer steigenden Anzahl von mechanischen, elektrischen, elektronischen und Softwarekomponenten zeigt (vgl. Schernikau 2001, S. 60), nehmen die Beziehungen und Wechselwirkungen zwischen den einzelnen Produktkomponenten zu. Proportional zur Komplexität eines Produktes verhält sich auch die Anzahl der Anforderungen, um ein solches Produkt zu definieren und zu beschreiben (vgl. Klute et al. 2011, S. 537).

Variantensystematik

Zur Variantenplanung eines Produktes verwenden Unternehmen eine Variantensystematik (vgl. Harms 2009, S. 41). An oberster Stelle der Variantensystematik steht das **Produktprogramm** oder synonym das **Produktportfolio** des Unternehmens. Das Produktprogramm umfasst alle am Markt durch das Unternehmen angebotene Produkte und wird in den Dimensionen Programmbreite als Anzahl der verschiedene Produktarten und Programmtiefe als Anzahl unterschiedlicher Varianten eines Produktes gemessen. (Ponn et al. 2011, S. 249; Lingnau 1994, S. 105ff.; Renner 2007, S. 12; Blees 2011, S. 8; Bauer 2016, S. 22)

Das Produktprogramm besteht aus mehreren Produktfamilien. Eine **Produktfamilie** besteht aus verschiedenen Produkten, die gemeinsame Technologien und Komponenten haben und für ähnliche Anwendungsbereiche (Funktionen) am Markt entwickelt wurden. (Blees 2011, S. 8; Grotkamp, S. 7; Gebhardt et al. 2016, S. 121; Meyer et al. 1997, S. 16) In der Literatur werden die Begriffe **Produktfamilie** und **Produktlinie** sowie Plattform häufig synonym verwendet (Schuh et al. 2008, S. 23; Tesch 2010, S. 23; Dellanoi 2006, S. 46). Allerdings setzt die Bezeichnung als Plattform voraus, dass im Rahmen einer Standardisierung von Bauteilen und Schnittstellen produktfamilienübergreifend Bauteile wiederverwendet werden (Schuh 2005, S. 132). Kotler et al. (2001, S. 719) und Dellanoi (2006, S. 47) unterscheiden die Begriffe Produktfa-

milie und Produktlinie: Während die Produkte einer Produktfamilie zwar dasselbe Bedürfnis erfüllen, weisen Produkte derselben **Produktlinie** die gleiche Zielgruppe und Preisklasse sowie Distributionssystem auf. Produkte einer Produktfamilie verfügen über eine ähnliche Produktarchitektur (Kirner 2014, S. 91; Dellanoi 2006, S. 47) und können basierend auf einer gemeinsamen Plattform durch Ableitung von Produktvarianten/Derivaten unterschiedliche Marktsegmente bedienen (Bauer 2016, S. 22).

Innerhalb einer Produktlinie können eine oder mehrere **Baureihen** bzw. **Modellreihen** existieren (Abbildung 2-8). Der Begriff Baureihe bezeichnet dabei sowohl ein Produkt als auch Baugruppen oder Einzelteile, die die gleiche Funktion mit derselben Lösung erfüllen und in unterschiedlichen Größen angeboten werden. Diese unterschiedlichen Größenstufen werden dabei nach definierten Regeln aus einem Grundentwurf abgeleitet. (Pahl et al. 2007, S. 640; Kirner 2014, S. 106)

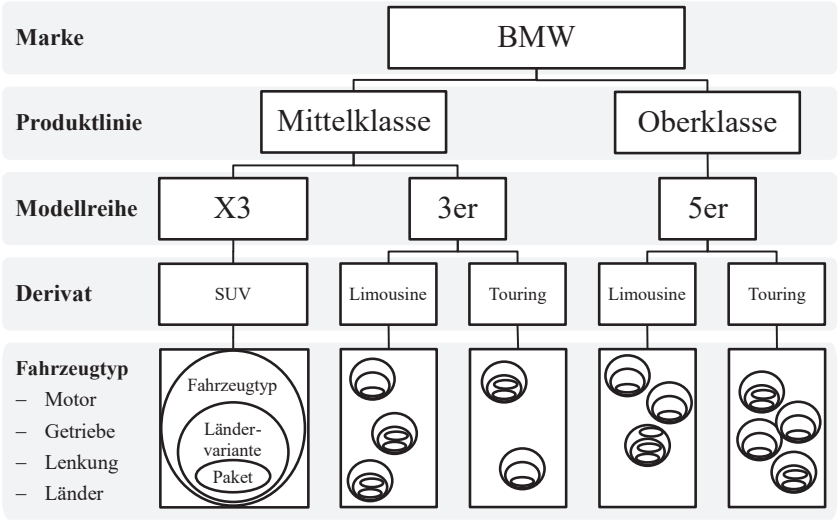


Abbildung 2-8: Variantensystematik am Beispiel der BMW AG (Harms 2009, S. 41)

Bei der Ableitung handelt es sich um Anpassungskonstruktionen, bei denen die Funktion, konstruktive Lösung, Werkstoffe und Fertigung gleich bleiben und nur die Leistungsdaten sowie Abmessungen, Gewicht und Kosten variieren (Ehrlenspiel et al. 2013, S. 725f.).

Die aus einem Grundentwurf oder Grundmodell abgeleiteten Varianten werden in der Literatur auch als **Derivate** bezeichnet (Dellanoi 2006, S. 47). Insbesondere in der Au-

tomobilindustrie bezeichnen Derivate die aus einem Basismodell abgeleiteten Karosserievarianten (Hasenpusch 2018, S. XXVII; Harms 2009, S. 41). Nach DIN 70010 sind dies unter Personenkraftwagen gängigen Karosserievarianten Limousine, Coupé, Kabriolett und Kombi sowie die in der Norm nicht explizit genannten, aber unter Mehrzweck-Personenkraftwagen einzuordnenden SUV (Sports Utility Vehicle) (Tesch 2010, S. 25). Innerhalb eines Derivats kann es noch weitere Variantenunterscheidungen je nach Produkt geben. So kann es z.B. erforderlich sein, länderspezifische Varianten und Sonderausstattungen anzubieten. Diese Variantensystematik zeigt das Potential der Entwicklung einer Plattform als einheitliche Produktarchitektur zur Reduzierung der Variantenvielfalt und Erhöhung der Kommunalität (Dellanoi 2006, S. 47).

Als **Produktarchitektur** bezeichnet man dabei die funktionale als auch physikalische Beschreibung (Baustruktur) eines Produktes, die Zuordnung der funktionalen Elemente zu einer physikalischen Komponente und die Beschreibung der Schnittstelle zwischen den wechselwirkenden Komponenten (Ulrich et al. 2020, S. 190f.). Wird eine Produktarchitektur auf Ebene einer Produktfamilie entwickelt und dadurch Komponenten und Schnittstellen einer Produktfamilie standardisiert, können Skaleneffekte optimal ausgenutzt und gleichzeitig auf der Kundenseite die gewünschte Varianz angeboten werden (Schuh 2005, S. 133f.). Die Entwicklung von Plattformen ist eine von mehreren Strategien des Variantenmanagements, auf das im Folgenden eingegangen wird.

Variantenmanagement, -beherrschung und -reduzierung

Eine steigende Variantenvielfalt ist einerseits für die Unternehmen erforderlich, um die Diversifizierung der Märkte und wachsenden Anforderungen der Kunden Rechnung zu tragen und dabei aufgrund des großen Konkurrenzdrucks in der Globalisierung auch Marktnischen zu erschließen. (Ponn et al. 2011, S. 250) Andererseits hat eine große Variantenvielfalt auch negative Folgen für das Unternehmen. Durch die starke Diversifikation der Produkte entfällt auf ein Produkt ein immer geringeres Absatzvolumen bei gleichbleibendem oder sogar steigendem Aufwand in der Entwicklung und Herstellung des Produktes (Abbildung 2-9), den sogenannten Komplexitätskosten. Daher ist es für die Wettbewerbsfähigkeit der Unternehmen erforderlich, die Varianten zu hinterfragen und ein Optimum der Variantenvielfalt im Kosten-Nutzen-Verhältnis zu finden.

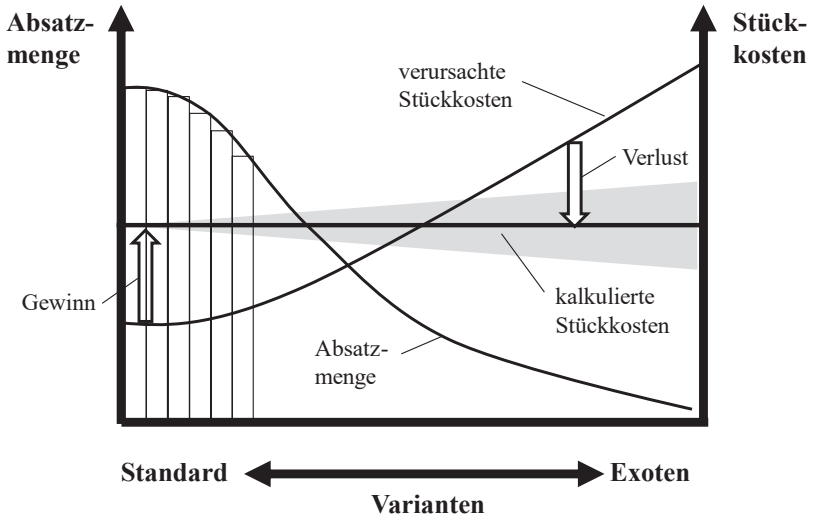


Abbildung 2-9: Wirtschaftliche Folgen der Variantenvielfalt (Firchau 2003, S. 11)

Das **Variantenmanagement** beschreibt als Oberbegriff alle Strategien und Maßnahmen, durch die die Anzahl der Varianten und deren Vielfalt sowohl produkt- als auch prozessseitig in einem Unternehmen beeinflusst wird. Das Ziel des Variantenmanagements ist es, die durch die Variantenvielfalt entstandenen Komplexitätskosten zu reduzieren bei gleichzeitiger Erfüllung der vom Markt geforderten Angebotsvielfalt. (vgl. Ehrlenspiel et al. 2014, S. 294f.) In der Literatur werden dabei die Strategien der Variantenvermeidung, Variantenbeherrschung und Variantenreduzierung unterschieden, die in verschiedenen Phasen des Produktlebenszyklus eingesetzt werden (Abbildung 2-10). (vgl. Heina 1999, S. 40-42; Kersten 2002, S. 7)

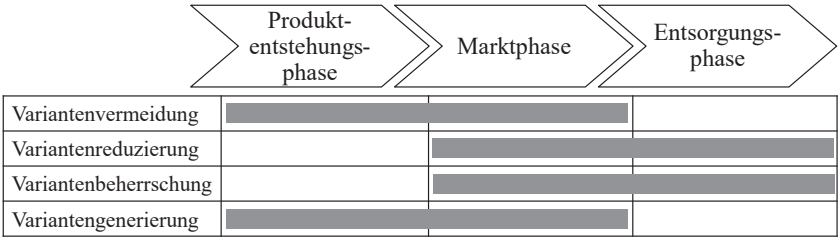


Abbildung 2-10: Strategien des Variantenmanagements im Produktlebenszyklus (Heina 1999, S. 42)

Werden in der frühen Phase der Produktentwicklung durch eine Analyse und Prognose der Nachfrage gezielt Produktvarianten angeboten, kann dies zur **Variantenvermeidung** dienen. (Kesper 2012, S. 42) Darüber hinaus können durch eine variantengerechte Konstruktion Varianten von Produkten, Bauteilen und Prozessen vor ihrer Entstehung verhindert werden. (vgl. Heina 1999, S. 48ff.)

Die **Variantenreduzierung** bezieht sich auf die Eliminierung von bereits existierenden Produktvarianten, sowie Teile- und Prozessvarianten (vgl. Gembrys 1998, S. 8ff.; Kesper 2012, S. 42) und setzt somit in einer späteren Phase des Produktlebenszyklus ein (vgl. Heina 1999, S. 42). Allerdings sollten durch die Reduzierung der Varianten die Erfüllung der Kundenanforderungen nicht eingeschränkt werden. (vgl. Kesper 2012, S. 42)

Die Strategie der **Variantenbeherrschung** setzt bei bereits existierenden Varianten an, in dem sie Methoden für eine effiziente Gestaltung der Prozesse zur Handhabung und Bewältigung der Varianten vorgibt (vgl. Kesper 2012, S. 41f.). Dabei bleibt die vorhandene Variantenvielfalt allerdings bestehen und soll durch die Methoden nicht beeinflusst werden. Stattdessen sollen Organisation als auch Fertigungstechnik effizient zur Bewältigung der Variantenvielfalt eingesetzt werden (vgl. Heina 1999, S. 53).

2.2.2 Variantentreiber

Nach der Definition der grundlegenden Begriffe Variante, Variantenvielfalt, Komplexität und die Erläuterung des Variantenmanagements sollen in diesem Kapitel die Ursachen der Variantenvielfalt beschrieben werden. In der Literatur werden diese Ursachen häufig als **Variantentreiber** bezeichnet. Die Variantentreiber werden in externe und interne Treiber unterteilt: **Externe** Treiber sind Ursachen außerhalb des Unternehmens wie z.B. Markt- und Gesetzesanforderungen. Sie können i.d.R. nicht oder nur schwer durch das Unternehmen beeinflusst werden. **Interne** Treiber sind unternehmensinterne Prozesse und Rahmenbedingungen, durch die Varianten entstehen. (Ehrlenspiel et al. 2014, S. 299; Jeschke 1997, S. 13f.)

Externe Variantentreiber

Die externen Ursachen der Variantenvielfalt lassen sich in die Gruppen Vorschriften, technologische Entwicklung, gesellschaftlicher und politischer Wandel sowie Marktkomplexität untergliedern (Renner 2007, S. 24). Eine Vielzahl der Varianten entsteht aus den **Vorschriften** der Länder, in denen ein Produkt angeboten wird. So können Gesetze, Normen und Richtlinien international stark differenzieren und dadurch Vari-

anz verursachen (Ehrlenspiel et al. 2014, S. 299ff.). Ein Beispiel für variantenverursachende gesetzliche Vorschriften ist die Vorgabe eines Local Contents, also der Vorgabe eines bestimmten Anteils der Wertschöpfung im Zielland zu erzeugen. Die Einhaltung der jeweiligen Gesetze, Normen und Richtlinie ist allerdings für die Unternehmen zwingend erforderlich, sodass die daraus resultierenden Varianten für Unternehmen nicht vermeidbar sind.

Die nächste Gruppe der externen Variantentreiber stellen die **technologische Entwicklung** dar. So kommt es durch neue Technologien und Innovationen zu immer kürzer werdenden Produktlebenszyklen (Ponn et al. 2011, S. 250; Ehrlenspiel et al. 2014, S. 299ff.; Franke et al. 2002, S. 5). Dies lässt sich insbesondere am Beispiel der Computertechnologie und dem Mooreschen Gesetz verdeutlichen, nachdem sich die Anzahl der Schaltkreiskomponenten auf einem Chip/ Schaltkreis pro Jahr verdoppelt (Moore 1965, S. 115). Dies hat dazu geführt, dass Produkte, die Computerchips enthalten, bereits nach wenigen Monaten veraltet sind. Durch diesen Innovationsdruck und schnelle technologische Entwicklungen kommt es auch zu kurzfristigen Überarbeitungen und Neuentwicklungen in Unternehmen (Ponn et al. 2011, S. 250; Ehrlenspiel et al. 2014, S. 299ff.), um kein veraltetes Produkt auf den Markt zu bringen. Darüber hinaus hat die technologische Entwicklung dazu geführt, dass sich Kunden weltweit über Produkte und Unternehmen informieren können (Renner 2007, S. 24f.).

Dies führt zur nächsten Gruppe der externen Variantentreiber: dem **gesellschaftlichen und politischen Wandel**. Die Werte und Normen in der Gesellschaft verändern sich und können sich auch teils regional stark unterscheiden (Renner 2007, S. 24f.; Franke et al. 2002, S. 5). Einer der Entwicklungen in diesem Bereich ist die Priorisierung von nachhaltigen und umweltfreundlichen Produkten, auf die Kunden einen steigenden Wert legen. Um am Markt diese Kundengruppen zu erreichen, müssen von Unternehmen entsprechende Produkte angeboten werden. Gleichzeitig verursachen aber auch gesellschaftliche Veränderungen in der Demografie Varianten, da Unternehmen einerseits auf die geänderten Anforderungen der Altersgruppen eingehen, andererseits sich das Kaufkraftaufkommen durch den demographischen Wandel in den Altersgruppen verschiebt (Renner 2007, S. 24f.; Franke et al. 2002, S. 5). Zusätzliche müssen Unternehmen auf politische Veränderungen weltweit (z.B. Brexit, Handelskriege, eingeschränkte Handelswege durch nationale Erlasse aufgrund der Corona-Krise etc.) reagieren, die ebenfalls zu Variantenvielfalt führen können. Einer der größten Variantentreiber in dieser Gruppe ist der Trend der Individualisierung (Renner 2007, S. 24f.). In der Gesellschaft bekommt die Selbstverwirklichung einen wachsenden Stellenwert und wird getrieben durch steigende individuelle Wahlfreiheit. Dies spiegelt sich auch

in den vom Kunden geforderten Individualisierungsmöglichkeiten in Produkten wider, die abermals zu Variantenvielfalt führen.

Die letzte Gruppe der externen Variantentreiber stellt die **Marktkomplexität** dar (Ehrlenspiel et al. 2014, S. 299ff.). In dieser Gruppe befindet sich die Globalisierung und ihre Auswirkungen auf die Variantenvielfalt. Durch die Vernetzung internationaler Wirtschaftsbeziehungen und der Möglichkeit der Kunden weltweit ihre Produkte zu beziehen, steigt sowohl die Wettbewerbsintensität als auch die -dynamik (Ehrlenspiel et al. 2014, S. 299ff.). Die Internationalisierung bringt zwar den Vorteil, neue wachsende Märkte zu erschließen sowie der Risikostreuung um Marktschwankungen auszugleichen (Renner 2007, S. 24f.), allerdings müssen länderspezifische Anforderungen erfüllt werden. Diese beinhalten nicht die unter der Gruppe Vorschriften genannten gesetzlichen Vorgaben, sondern vielmehr die verschiedenen klimatischen Umgebungsbedingungen im internationalen Kontext als auch die unterschiedlichen ergonomischen Anforderungen der Bevölkerungsgruppen (Ehrlenspiel et al. 2014, S. 299ff.). Des Weiteren führt die Globalisierung auch zu einer größeren Lieferantenvielfalt, von denen Komponenten bezogen werden können (Böttrich 2015, S. 28). Dies führt wiederum zu einer steigenden Variantenvielfalt, da dasselbe Bauteil an verschiedene Lieferanten vergeben sein kann. Sind die traditionellen Märkte gesättigt, weichen viele Unternehmen in Nischen aus und Segmentieren dadurch den Markt. Dies kann auch aus der Notwendigkeit resultieren, sich von der Konkurrenz differenzieren zu müssen (Ehrlenspiel et al. 2014, S. 299ff.). Sowohl diese Marktsegmentierung als auch die Diversifikation bekannter und neuer Bedarfe führen zu immer neuen Varianten (Ehrlenspiel et al. 2014, S. 299ff.; Franke et al. 2002, S. 5). Abschließend soll in der Gruppe der Marktkomplexität auch die Problematik der Plagiate genannt sein. Durch das Risiko eines Unternehmens, durch Plagiate der Produkte wirtschaftlichen Schaden zu erleiden, sehen sich Unternehmen gezwungen, schnell neue Produkte oder Varianten mit neuen Merkmalen zu veröffentlichen, um den Plagiatoren zuvorzukommen (Ehrlenspiel et al. 2014, S. 299ff.). Dies kann ebenfalls in bestimmten Branchen als externer Variantentreiber wirken.

Interne Variantentreiber

Innerhalb eines Unternehmens werden Varianten durch Defizite in den Bereichen Methoden, Organisation, Struktur, Information und Kommunikation sowie individuelle Defizite verursacht. Die internen Variantentreiber werden daher in diesen fünf Bereichen im Folgenden erläutert: Im Bereich der **methodischen Defizite** führt ein fehlendes Bewusstsein der Mitarbeiter (Wildemann 2012, S. 11) sowie mangelnde Transpa-

renz über die bereits existierende Variantenvielfalt (Jeschke 1997, S. 13) zu einer Vernachlässigung dieser und somit zu steigender Vielfalt. Oft mangelt es aber auch an Werkzeugen für ein markt- und kostengerechtes Produktprogramm und einer Produktstrategie. Nichtexistierende Regelungen und Richtlinien zur Variantenvermeidung und -reduzierung führen zu neuen Varianten (Jeschke 1997, S. 13) ebenso wie eine zu späte Standardisierung der Bauteile. Eine unzureichende Beschreibung der Produktstruktur (Ehrlenspiel et al. 2014, S. 301) als auch die fehlende Übersicht über das Produktprogramm, den einzelnen Komponenten, deren Schnittstellen und einem Wiederholteilsuchsystem (Ehrlenspiel et al. 2014, S. 301; Jeschke 1997, S. 13) führen letztlich dazu, dass bestehende Bauteile nicht in neue Produkte übertragen werden. Statt einer effektiven und schnellen Anpassungskonstruktion steigt der Anteil der Neukonstruktionen. Wird in einer so historisch gewachsenen Produkt- und Teilevielfalt keine kontinuierliche Reduzierung des Teilestamms durchgeführt, steigt die Variantenvielfalt weiter (Jeschke 1997, S. 12). Zuletzt sind auch methodische Defizite in der Kostenrechnung ein Grund, dass die Variantenvielfalt steigt, da die durch sie entstehenden Kosten nicht transparent werden. Konventionelle Kalkulationsverfahren sind nicht verursachungsgerecht (Ehrlenspiel et al. 2014, S. 301) und führen zu einer Quersubventionierung der vielfaltinduzierten Kosten (Kesper 2012, S. 31). Die Kostenverursachung wäre aber ein wichtiges Bewertungskriterien bei der Beurteilung unnötiger Varianten. Zusätzlich wäre eine Transparenz über die varianteninduzierten Kosten wichtig, da zwischen der Verursachung der Kosten in der Entwicklungsphase und der Kostenentstehung in der Fertigungsphase sowie in der Ersatzteilebereitstellung eine erhebliche zeitliche Differenz liegt.

Die nächste Gruppe der internen Variantentreiber sind die **organisatorischen Defizite**. Greift die Unternehmensleitung durch aktionistisches Verhalten und unabgestimmten Entscheidungen kurzfristig in die Produktentwicklung ein, entstehen häufig neue Varianten (Ehrlenspiel et al. 2014, S. 301), da die Auswirkungen in den Unternehmensbereichen aufgrund der ad hoc Entscheidung keine Berücksichtigung finden können. Insgesamt lässt sich an dieser Stelle festhalten, dass ein unstrukturiertes Änderungsmanagement in allen Unternehmensbereichen zu Variantenvielfalt führen kann (Ehrlenspiel et al. 2014, S. 301). Eine Konstruktion kann zwar schnell in CAD-Systemen geändert werden, aber die Herstellung dieser Änderung ist meist kostenintensiv und schlägt zeitlich später auf, als die Änderung der Konstruktion. Ein kontinuierlicher Verbesserungsprozess (KVP) und die Nutzung vorhandener Erfahrung kann helfen, Variantenvielfalt zu reduzieren, allerdings kann ein KVP ohne übergeordnete Strategie neue Varianten verursachen (Ehrlenspiel et al. 2014, S. 301). Ein weiterer, nicht zu

unterschätzender interner Variantentreiber besteht in den konkurrierenden Bereichsziele in einem Unternehmen. Jeder Unternehmensbereich verfolgt eigene Ziele, sodass die unterschiedlichen Bereichssichten koordiniert werden müssen. Findet dies nicht statt und tritt ein Unternehmensbereich besonders eigennützig auf, kann dies zu unnötigem Variantenwachstum führen. Ist der Vertrieb zum Beispiel gegenüber der Konstruktion und Produktion zu stark ausgeprägt, werden individuelle Kundenwünsche ohne Berücksichtigung der Komplexitätskosten angeboten und sich hauptsächlich am Umsatz anstelle des Gewinns orientiert. (vgl. Kesper 2012, S. 31) Die reine Umsatzorientierung wirkt dabei dem Ziel der Variantenreduzierung entgegen (Jeschke 1997, S. 12). Oft werden neue Varianten auch als Türöffner für Folgeaufträge oder der Erschließung neuer Marktsegmente auf Druck des Vertriebs angeboten, obwohl sie sich für das Unternehmen nicht rentieren (Ehrlenspiel et al. 2014, S. 301). Des Weiteren kann die Zusicherung des Vertriebs und insbesondere des Aftersales-Bereichs an die Kunden, auch nach Jahren alle Ersatzteile von einem OEM beziehen zu können, eine große Variantenvielfalt verursachen, da auch Jahre nach Ende des Produktlebenszyklus Bauteile produziert und in Ersatzteillagern eingelagert werden müssen (Jeschke 1997, S. 12). Insbesondere Produkte mit langer Lebensdauer verursachen bei Verfügbarkeit von Ersatzteilen und Serviceleistungen hohe Kosten. Andererseits können auch die Ziele der Fertigung selbst Varianten verursachen. Eines der Ziele der Fertigung ist es, die Fertigungskapazitäten bestmöglich auszulasten, sodass teils Varianten nur zur Auslastung der Produktion angeboten werden (Kesper 2012, S. 31). Zugleich kann auch das Poka Yoke Prinzip zur Fehlervermeidung in der Fertigung ein konkurrierendes Ziel zur Reduzierung der Komponentenvarianten darstellen (Kesper 2012, S. 31). Diese Beispiele der Zielkonflikte zwischen den Unternehmensbereichen zeigen, dass je mehr Stellen in der Wertschöpfungskette eingebunden sind, es eine Koordination dieser Bereichssichten geben muss und geeignete Entscheidungsstrukturen geschaffen werden müssen. Des Weiteren muss bei der Schaffung einer Normungsstelle zur Kontrolle der Variantenvielfalt auch sichergestellt werden, dass sie die entsprechenden Befugnisse und Unterstützung durch die Unternehmensführung hat.

Dies führt zum nächsten Bereich der internen Variantentreiber: den **strukturellen Defiziten**. Die zuvor genannten Zielkonflikte der Unternehmensbereiche können auch auf Mängel in der Struktur zurückzuführen sein. Eine Funktionsorientierung, eine Vielzahl von Hierarchieebenen, eine hohe Schnittstellendichte und damit einhergehende lange Entscheidungsprozesse können dazu führen, dass aufgrund der Volatilität der Märkte kurzfristige Änderungen durchgeführt werden müssen, die dann aufgrund der langen Entscheidungswege nicht mit allen Bereichen abgestimmt werden (vgl.

Wildemann 1995, S. 22f.). Außerdem können diese strukturellen Defizite wiederum zu einer Trennung von Aufgabe, Verantwortung und Kompetenz führen. Eine Vielzahl von Hierarchieebenen kann ebenfalls zu einem Übermaß an Kontrollinstanzen führen, die in der Literatur ebenfalls als Variantentreiber genannt werden.

Neben den strukturellen Defiziten können auch **Mängel in der Kommunikation und Informationswesen** für eine große Variantenvielfalt verantwortlich sein (Ehrlenspiel et al. 2014, S. 301). Informationsasymmetrie zwischen den Unternehmensbereichen und Entscheidungsträgern, resultierend aus Medienbrüchen und fehlendem Zugang bzw. Zugriff auf die erforderlichen Informationen, führt zu steigender Komplexität. Gleichzeitig wird auch das Bring-Prinzip bei der Weitergabe von Informationen und Kommunikationsdefizite zwischen Unternehmensbereichen als Ursachen für die Informationsasymmetrie genannt (Wildemann 1995, S. 23). Allerdings können auch die Maßnahmen zur Standardisierung der Informationen und Verbesserung der Kommunikation in einem umfangreichen Berichts- und Formularwesen übertrieben werden, dass eine gegenteilige Wirkung erzielt wird (Wildemann 1995, S. 23).

Abschließend soll noch auf die **individuellen Defizite** als mögliche interne Variantentreiber eingegangen werden. So beschreibt Wildemann (1995, S. 23), dass Machtstreben von Führungskräften und Bereichsegoismen zu einem Verschleiern und Verkomplizieren von Informationen führen würde und dadurch zu einer quantitativ nicht bestimmbaren Auswirkung auf die Komplexität der Organisation haben. In diesem Zusammenhang nennt Wildemann auch mangelnde Motivation oder Identifizierung mit den Unternehmenszielen ebenso wie fehlende Sozial- und Fachkompetenz als Komplexitätstreiber einer Organisation. Abschließend ist unter den individuellen Defiziten als Variantentreiber noch die Technikverliebtheit einiger Mitarbeiter zu nennen, deren Tendenz zur technologischen Überausstattung, engl. Overengineering, zu hoher Variantenvielfalt und vom Kunden nicht wahrgenommenen oder bestellten Produkten führt (Kesper 2012, S. 32).

Nach der ausführlichen Erläuterung der in der Literatur untersuchten internen und externen Variantentreiber soll im Folgenden insbesondere auf Anforderungen als Variantentreiber zur Identifizierung variantentreibender Anforderungen eingegangen werden.

2.2.3 Methoden zur Identifizierung variantenverursachender Anforderungen

Vendég (2020, S. 31) stellt fest, dass das Variantenmanagement mit der Identifizierung der Variantentreiber in der Produktstruktur beginnt und dass das Erkennen von konstruktiven Merkmalen, die die Varianten verursachen, großes Potential zur Standardisierung ergibt. Konstruktive Merkmale leiten sich aus Anforderungen an das Produkt und dessen Bauteile ab. Gemäß der Definition einer Anforderung in Kapitel 2.1.2 als zu erfüllende Bedingung oder Eigenschaft, ist es daher von Interesse, Anforderungen hinsichtlich ihrer variantenverursachenden Wirkung zu untersuchen. Zusätzlich kann vor dem Hintergrund der in Kapitel 2.2 erläuterten methodischen Defizite ein weiteres Potential zur Reduzierung der Variantenvielfalt in der Schaffung einer Transparenz und Übersicht über die vorhandenen Anforderungen und ihrer Spreizung liegen, um daraus ein Wiederholteilsuchsystem oder ein System zur Wiederverwendung von Anforderungen zu schaffen. In der Literatur gibt es bereits einige Arbeiten, die sich mit variantenerzeugenden Anforderungen beschäftigen. Franke et al. (2002, S. 55) und Firchau (2003, S. 44) beziehen sich dabei auf die zuvor genannten externen Variantentreiber:

- Unterschiedliche Gesetze, Normen, Richtlinien → gesetzliche Anforderungen zur Produktzulassung/Produkthaftung
- Klimatische Bedingungen → Anforderungen an Temperaturbeständigkeit und Korrosionsbeständigkeit (Luftfeuchtigkeit)
- Verschiedene Sprachen → Anforderungen an Bedienung und Handbuch
- Unterschiedliche Anatomie der Kunden → ergonomische Anforderungen
- Unterschiedliche Netzspannungen und Frequenzen → funktionale Anforderungen.

Häufig sprechen die Autoren auch unspezifisch von Kundenanforderungen, die die Varianten verursachen, ohne dabei genauer einzugehen, welche Kundenanforderungen dies sind (Ponn et al. 2011, S. 247) oder von der Dynamik, in der sich Anforderungen ändern, wodurch neue Varianten entstehen (Guodong et al. 2015). Schuh et al. (2016) untersuchen in ihrer vorgestellten Studie speziell variantenverursachende Faktoren für Dienstleistungsprodukte. Ihre Literaturrecherche hat dabei 28 Faktoren ergeben, die Varianten in Bezug auf Dienstleistungen verursachen (Schuh et al. 2016, S. 584). Da die vorliegende Arbeit physische Produkte und deren Komponenten behandelt, wird an dieser Stelle nicht weiter auf die Variantentreiber für Dienstleistungen eingegangen.

In der Literaturrecherche wurden daher die veröffentlichten Ansätze hinsichtlich der folgenden vier Kriterien bewertet: Kriterium 1 „Variantentreiber“ beurteilt, ob der beschriebene Ansatz die Ursachen von Varianten untersucht und diese identifiziert. Unter dem Kriterium 2 „Variabilität von Anforderungen“ wird betrachtet, ob der vorhandene Ansatz sich mit der Spreizung von Anforderungen zwischen verschiedenen Produktvarianten beschäftigt. Als nächstes Kriterium 3 „technische Anforderungen“ wird der Objektbereich der vorhandenen Ansätze bewertet. Hierbei wird zwischen Kundenanforderungen einerseits und technische Anforderungen andererseits unterschieden. Fokus der vorliegenden Arbeit liegt auf technischen Anforderungen. Das letzte Kriterium 4 „mechanische Komponenten“ schließt den Objektbereich der betrachteten Ansätze ab und untersucht, ob der beschriebene Ansatz sich auf Softwarekomponenten oder immaterielle Produkte wie Dienstleistungen bezieht oder ob er auf Anforderungen an physische Produkte wie mechanische Komponenten angewendet wird. Eine vollständige Auflistung der bewerteten Veröffentlichungen ist in Anhang B zu finden. Die vier zutreffendsten Ansätze werden an dieser Stelle weiter erläutert.

Renner (2007) - Anforderungsharmonisierung und –optimierung

Im Hinblick auf das Anforderungsmanagement gibt es nur wenige Ansätze, um bereits bei der Anforderungsdefinition Variantentreiber zu identifizieren. Eine Methode zur Betrachtung von Anforderungen als Variantentreiber in Bezug auf eine Baukastenentwicklung stellt Renner vor. Durch eine Anforderungsharmonisierung und -optimierung sollen die Anforderungen für Baukastenentwicklungen angeglichen und dadurch eine höhere Kommunalität der Komponenten erzielt werden. In der **Anforderungsoptimierung** werden Anforderung hinsichtlich einer Übererfüllung hinterfragt. Dies kann z.B. durch eine Anforderung vorliegen, die für den Anwendungsfall nicht erforderlich wäre. Eine solche Übererfüllung ist mit höheren Produktkosten verbunden, sodass durch ein Hinterfragen und Plausibilitätscheck der Anforderungen Kosten eingespart werden können. Die Herausforderung besteht bei großem Kostendruck darin, in der Optimierung der Anforderungen nicht zu einer Untererfüllung der Anforderungen zu kommen und dadurch Qualitätsproblemen zu erzeugen.

Die **Anforderungsharmonisierung,** Ehrlenspiel et al. (2013, S. 728) spricht auch von einer Anforderungskanalisierung, ist für die Baukastenentwicklung essenziell. Um Wiederholbauteile in verschiedenen Produkten einsetzen zu können, müssen die Anforderungen angeglichen werden. Diese Methode betrachtet ausschließlich variable Anforderungen mit unterschiedlichen quantitativen Ausprägungen (Abbildung 2-11). Ziel dieses Ansatzes ist es, die unterschiedlichen Ausprägungen einer variablen An-

forderung auf möglichst wenige, am besten einen einzigen Zielwert zu vereinheitlichen. Durch diese Angleichung kommt es in manchen Anwendungsfällen zu einer Über- oder Untererfüllung. Daher ist bei jeder dieser Harmonisierungen ein Abwägen zwischen den Vorteilen der Vereinheitlichung für die Kommunalität von Bauteilen und den Auswirkungen auf die Kundenwertigkeit erforderlich. Wird eine Über- oder Untererfüllung durch die Kunden toleriert oder eine Übererfüllung mit einem höheren Preis akzeptiert, kann auch eine spezifische Ausprägung aus wirtschaftlicher Sicht für ein Unternehmen sinnvoll sein.

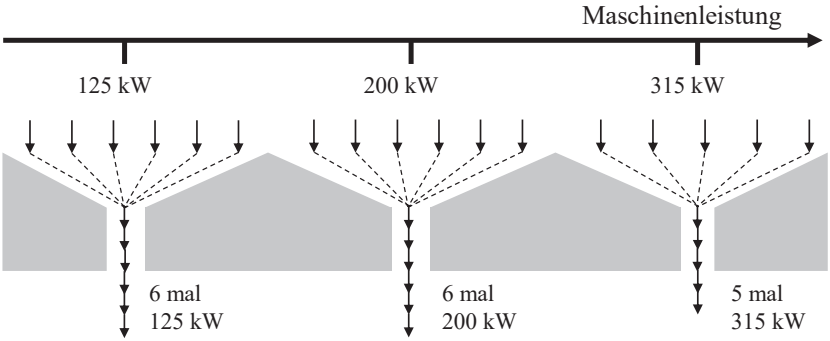


Abbildung 2-11: Anforderungsharmonisierung am Beispiel der Maschinenleistung (Ehrlenspiel et al. 2013, S. 728)

Renner (2007) hat in Bezug auf die Anforderungen an Sitze in der Automobilindustrie sowohl Variantentreiber unter den allgemein gültigen Anforderungen an Sitze wie Crashnormen identifiziert, als auch in den von der Fahrzeugklasse und Karosserieform abhängige Anforderungen:

- Bauraumanforderungen (Package) → Spurweite, Sitzhöhe etc.,
- Funktionale Anforderungen (Ausstattung) → umklappbare Sitze bei 3-türigen Fahrzeugen, Gurtsystem Befestigung, elektrische Verstellung, Komfortfunktion Sitzbreite etc.,
- Designanforderungen → räumliche Wirkung, Nahtbilder, Farben etc.,
- Monetäre Anforderungen → Herstellkosten, Montagekosten etc.,
- Fertigungsanforderung → Schweiß- oder Tiefziehkonstruktion.
  (vgl. Renner 2007, S. 145; Ponn et al. 2011, S. 269)

Diese Aufzählungen von Renner zeigen, dass die Anforderungen in Merkmale übersetzt werden und diese Merkmale voneinander differieren, wodurch die Varianten ent-

stehen. Allerdings fehlt in dem Ansatz das Vorgehen zur Identifizierung dieser variantenverursachenden Anforderungen. Renner beschreibt lediglich einen manuellen Prüfungsprozess der Anforderungen. Ein manueller Prozess ist allerdings bei einer großen Anforderungsmenge ungeeignet.

Stechert (2010)

Das Ziel von Stecherts Ansatz ist die Erfassung und Aufbereitung von Anforderungen an mechatronische Produkte. In seiner Arbeit betrachtet er insbesondere die interdisziplinäre Entwicklungsarbeit von Maschinenbau, Elektronik und Informatik sowie die Zusammenarbeit mit vielen verschiedenen Entwicklungspartnern. Seine Methode besteht in der Modellierung von mehreren Partialmodellen mit unterschiedlichem Abstraktionsgraden (Systemidee-, Ziel-, Produktlebenslauf-, Produktumgebungs-, Stakeholder-, Anforderungs-, Systemkontext- und Testmodell) in der Modellierungssprache SysML. Im Hinblick auf die Anforderungen an komplexe mechatronische Produkte werden diese in Stecherts Ansatz manuell nach Funktion, Struktur und Verhalten klassifiziert und ihre Abhängigkeiten in einem Anforderungsmodell dargestellt. Die Variabilität von Anforderungen wird in einer Anwendungsfall-Anforderungsmatrix abgebildet, die zur Identifizierung der Anforderungsspreizung dient. Die Erstellung erfolgt automatisiert durch eine Makroprogrammierung in dem Tabellenkalkulationsprogramm Excel aus dem Anforderungsmodell heraus. Jede Ausprägung einer Anforderung, im Modell als Eltern-Kind-Beziehung abgebildet, wird hier in einer separaten Zeile angelegt und den Anwendungsfällen (Spalten) in den Produktlebenslaufphasen zugewiesen. Durch eine Priorisierung der Anwendungsfälle können anschließend die einzelnen Ausprägungen der Anforderungen ebenfalls priorisiert werden, sodass hier analog Renner (2007) eine Anforderungsharmonisierung erfolgen kann. Zudem können durch eine Verknüpfung der Anforderungen mit dem Zielmodell bei einer vorliegenden Anforderungsspreizung mögliche Variantentreiber in den Zielen identifiziert werden (Stechert 2010, S. 114).

Die Erarbeitung der Modelle erfolgen in Stecherts Ansatz manuell. Im Anforderungsmodell müssen alle ermittelten Anforderungen und ihre Eigenschaften manuell im Modell dokumentiert und strukturiert werden. Ebenfalls müssen die Beziehungen zwischen den Anforderungen und zu den Objekten anderer Partialmodelle manuell erzeugt werden. Der Aufwand für die erstmalige Erzeugung dieser Modelle für ein Produkt ist entsprechend hoch, insbesondere für hochkomplexe Produkte wie einem Automobil, dass aus einer Vielzahl unterschiedlicher Teilsysteme und Komponenten besteht. Ebenfalls wird die Leistung des Anforderungsvergleichs und die Identifizierung

ähnlicher Anforderungen nicht eindeutig erläutert und scheint durch die manuelle Erstellung des Anforderungsmodells abgedeckt zu werden. Die Automatisierung in diesem Ansatz besteht in der Erzeugung der Matrizen durch ein Excel-Makro aus den Partialmodellen heraus. Für die Anwendung auf eine große Anzahl bereits bestehender Anforderungen, die unstrukturiert in einer Vielzahl von Anforderungsdokumenten vorliegen und in diesem Ansatz zunächst manuell in das Anforderungsmodell übertragen werden müssten, ist der Ansatz von Stechert aufgrund des hohen Aufwands nicht geeignet.

Mamrot et al. (2013) - DeCoDe

Der Ansatz von Mamrot et al. zielt auf die Verwaltung einer großen Anzahl von Anforderungen als auch der Handhabung von Varianten ab. Sie klassifizieren Anforderungen in unveränderlich und änderbar in Anlehnung an Sitte et al. (2005; 2011) DeCoDe (Demand Compliant Design) Methode. Abbildung 2-12 zeigt die Zusammensetzung der Anforderungen der Varianten V1 und V2.

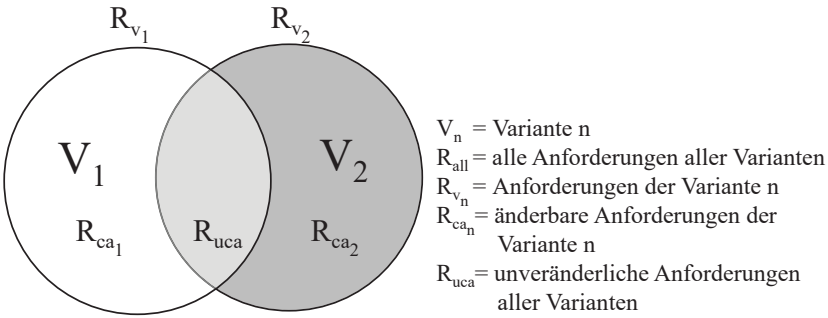


Abbildung 2-12: Teilmengen der Anforderungen für die Varianten V1 und V2 (Mamrot et al. 2013, S. 352)

Jede Variante besteht aus einer Gesamtmenge an Anforderungen R_{v_n} die sich aus änderbaren Anforderungen R_{ca_n} und unveränderlichen Anforderungen R_{uca} zusammensetzt. Das Vorgehen von Mamrot et al. ist in drei Schritte aufgeteilt: Zunächst muss ein Indikator für die Unterscheidung zwischen unveränderlichen und änderbaren Anforderungen gefunden werden. Durch die mathematische Beschreibung der Teilmengen analog dem Venn-Diagramm in Abbildung 2-12 sollen bei der Entwicklung einer neuen Variante V3 zunächst deren Anforderungen, mit denen der Varianten V1 und V2 verglichen werden. Das Variantenpaar mit der größten Schnittmenge R_{uca} soll damit identifiziert werden. Allerdings lassen Mamrot et al. an dieser Stelle aus, wie dieser Vergleich stattfinden soll. Nach erfolgreicher Identifizierung dieser Schnittmenge

können anschließend in Schritt zwei aus den änderbaren Anforderungen der neuen Variante ein Lösungsraum D mit entsprechenden Lösungsvariablen definiert werden. Je feingranularer die Anforderungen beschrieben sind, desto besser lassen sich diese quantitativ im Lösungsraum verorten. Abschließend beschreiben Mamrot et al. im dritten Schritt, dass sich die Unterschiede und Ähnlichkeiten zwischen Varianten zum einen durch die Zusammensetzung der änderbaren und unveränderlichen Anforderungen und zum anderen durch die Variabilität im Lösungsraum ergeben.

Dieser Ansatz beschreibt die Zusammensetzung der Anforderungen in unterschiedlichen Varianten und teilt die Ansicht, dass sich aus der Verschiedenheit der Anforderungen Varianten ergeben. Ebenso wird der Aspekt des Lösungsraums als Interpretation und Umsetzung der Anforderungen in der Theorie betrachtet. Allerdings lassen Mamrot et al. offen, wie sie zur Einteilung der Anforderungen in unveränderlich und änderbar kommen oder wie sie einen Vergleich der Anforderungen durchführen.

Boutkova (2014) – MIA, KDVM und R2F Ansatz

Boutkovas Arbeit basiert auf dem merkmalsbasierten Ansatz, bei dem die Gründe für die Variabilität einer Anforderung als Merkmale bezeichnet werden und in einem Merkmalsbaum in Beziehung stehen. Der Merkmalsbaum wird dabei in einem separaten, eigenständigen Dokument verwaltet. (Rupp 2014, S. 473f.) Boutkovas Methoden haben das Ziel eines merkmalbasierten Variantenmanagements in Anforderungsdokumenten. Zur Modellierung der Variabilität von Anforderungen werden im Merkmalsidentifizierungsansatz (MIA) Merkmale in den Anforderungen semi-automatisch identifiziert und anschließend im Requirements to Feature Ansatz (R2F) anhand der Ausprägungen ihrer Variabilitätskriterien geordnet. Der R2F-Ansatz soll dabei die Wiederverwendung von Anforderungen unterstützen. Als Variabilitätskriterien bezeichnet Boutkova variable Eigenschaften des Produktes. Sie stellen eine Untergruppe der Merkmale dar. Anhand der Variabilitätskriterien werden im KDVM-Ansatz (konfigurationsgekoppelter dezentraler Variabilitätsmodellierungsansatz) mehrere Variabilitätsmodelle sowohl für ganze Systeme als auch für einzelnen Komponenten erstellt. Die Variabilitätskriterien auf Systemebene bilden die funktionale, die Kriterien auf Komponentenebene die technische Sicht ab. Anhand von mehreren Verträglichkeitsmatrizen werden die möglichen Konfigurationen dargestellt.

Innerhalb des Werkzeugs MIA verwendet Boutkova Ansätze aus dem Bereich Natural Language Processing (s. Kapitel 2.3.2): PoS-Tagging zur Identifizierung von Substantiven, Lemmatisierung zur Reduzierung der Wörter auf ihren Wortstamm, Bereinigung anhand einer Stopp-Wort-Liste sowie von Duplikaten. Die Anwendung von MIA

erfolgt anhand deutschsprachiger Anforderungen an mechanischen Komponenten der Mercedes Benz AG (Boutkova 2014, S. 55). In der Auswertung misst Boutkova die Güte des Verfahrens anhand der Kennzahlen Genauigkeit (precision) und Trefferquote (recall). Sie unterscheidet vier Datensätze sowie zwei Durchläufe. Im ersten Durchlauf werden alle Anforderungen mithilfe MIA analysiert. Die Genauigkeit (precision) liegt hier je nach Datensatz zwischen 0,03 und 0,22 und die Trefferquote (recall) zwischen 0,31 und 0,89 (Boutkova 2014, S. 59). In einem zweiten Durchlauf werden ausschließlich variable Anforderungen in den Datensätzen durch MIA analysiert. Hierbei verbessert sich die Genauigkeit auf einen Bereich zwischen 0,06 und 0,3.

Im Fazit beschreibt Boutkova die erzielten Ergebnisse mit dem semi-automatischen Ansatz MIA als akzeptabel hinsichtlich der Genauigkeit. MIA würde mehr Merkmalskandidaten vorschlagen, als nötig und dennoch nie alle Merkmale finden. Durch die Teilautomatisierung der Merkmalsidentifikation sollte eigentlich Zeit und Aufwand reduziert werden. Boutkovas Ansatz beschreibt keine semantische Analyse oder ein Vergleich von Anforderungen hinsichtlich ihrer semantischen Ähnlichkeit. Sie versucht die Variabilität anhand der Merkmale, die sie in den Substantiven in den Anforderungen vermutet, automatisiert zu identifizieren. Allerdings ist aufgrund der erforderlichen Überprüfung der Ergebnisse des MIA Ansatzes aufgrund der Genauigkeit dennoch mit einem erhöhten zeitlichen Aufwand zu rechnen (Boutkova 2014, S. 134). Dies zeigt, dass eine rein lexikalische Analyse in der Anforderungsanalyse für die erforderliche Genauigkeit nicht zielführend ist. Hinsichtlich des semantischen Kontextes einer Anforderung stellt Boutkova fest, dass durch eine Integration der Überschriften des Anforderungsdokumentes in die Anforderungsanalyse der erforderliche Kontext ergänzt und somit die Trefferquote verbessert werden könnte (Boutkova 2014, S. 60).

2.3 Natural Language Processing (NLP)

Durch die steigende Anzahl an Anforderungen (Kapitel 2.1.2) ist es erforderlich, Methoden zur automatisierten Analyse von Text für natürlichsprachige Anforderungen zu betrachten. In diesem Kapitel soll daher eine kurze Einführung in die Begriffe und Methoden des Natural Language Processing im Hinblick auf Textvergleiche und eine Einordnung in den Bereich der Künstlichen Intelligenz gegeben werden. Des Weiteren werden Arbeiten vorgestellt, in denen bereits Ansätze des NLP im Anforderungsmanagement eingesetzt werden. Da die meisten Veröffentlichungen im Bereich Künstliche Intelligenz und Natural Language Processing auf Englisch erfolgen und um Missverständnisse vorzubeugen, werden im Folgenden auch immer die englischsprachigen Bezeichnungen aus der Literatur genannt.

2.3.1 Künstliche Intelligenz (KI), Machine Learning und neuronale Netze

Mit Steigerung der Rechenleistung von Prozessoren und dem Trend zur Digitalisierung hat der Einsatzbereich von **Künstlicher Intelligenz (KI)**, Artificial Intelligence (AI), in den letzten Jahren massiv zugenommen. 1955 wurde der Begriff Artificial Intelligence erstmalig mit dem Forschungsantrag von McCarthy et al. zur Erforschung einer Maschine, die selbstständig lernen, Sprache verwenden, Probleme lösen und sich selbst verbessern kann, genannt. Allgemein bezeichnet Künstliche Intelligenz Maschinen, die kognitive Fähigkeiten besitzen, die sonst nur Menschen aufweisen (Aust 2021, S. 8). Dabei wird zwischen starker und schwacher KI unterschieden. Während eine starke KI, strong/general AI, in der Lage ist, menschliches Denken wie Planung, logisches Denken, Abstraktion, Kommunikation und Entscheidungsfindung nachzubilden, ist eine schwache KI, weak/narrow AI, zur Lösung einer ganz spezifischen Aufgabe wie Schach, Bilderkennung, Stimmerkennung oder Vorschläge für Suchmaschinen entwickelt (Paschek et al. 2017, S. 3; Buxmann et al. 2019, S. 123; Aust 2021, S. 26).

Mittlerweile wird im Hinblick auf die Datenmenge, die weltweit laut IDC Prognose von 33 Zettabyte in 2018 auf 175 Zettabyte in 2025 wachsen wird (Reinsel et al. 2018), in immer mehr Bereichen KI zur Datenanalyse und Prognose der Big Data eingesetzt. Einige Beispiele für KI-Plattformen sind Watson von IBM, Cognitive Toolkit von Microsoft, TensorFlow von Google, Torch von Facebook oder DSSTNE von Amazon (Hecker et al. 2017, S. 11). Systeme künstlicher Intelligenz werden dabei entweder physisch in Form von Robotern, Drohnen, Fahrassistenten oder Überwachung und Steuerung von SmartHome verkörpert oder sie treten als digitale Systeme wie Bots in Social Media, virtuellen Assistenten, Bildauswertungsdienste etc. auf. Dabei können

sie sowohl autonom, als auch kooperativ oder lernend agieren (Hecker et al. 2017, S. 8).

Abbildung 2-13 zeigt verschiedene Einsatzmöglichkeiten von KI in den Bereichen der Computerlinguistik, der Robotik und der Bildverarbeitung und gibt einen Überblick über die Machine Learning Ansätze, die einer KI zu Grunde liegen. Auf diese Ansätze wird im Folgenden eingegangen. Gleichzeitig zeigt sie auch, dass NLP dem Einsatzgebiet der Computerlinguistik zuzuordnen ist und im NLP alle Machine Learning Ansätzen Anwendung finden können.

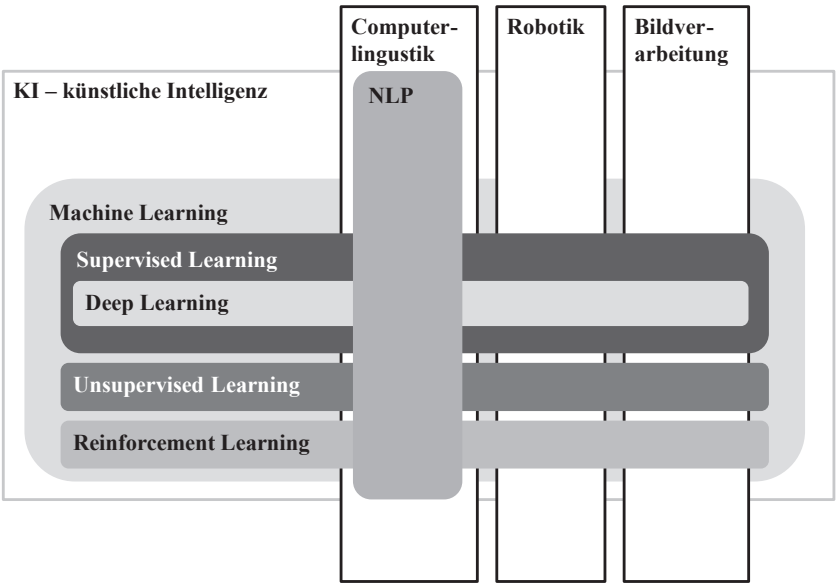


Abbildung 2-13: Einordnung des NLP und Abgrenzung zu Machine Learning

Machine Learning

Das **maschinelle Lernen (ML)**, Machine Learning, ist dabei ein zentrales Element der KI. Durch ML können KI selbstständig anhand von Beispieldaten statistische Regeln ableiten und daraus Modelle erstellen. Ohne dies müssten durch Programmierer alle Berechnungsvorschriften des Programms festgelegt werden. Innerhalb des ML existieren drei verschiedene Kategorien von Algorithmen: das überwachte, das unüberwachte und das bestärkende Lernen.

Im **überwachten Lernen (supervised learning)** wird der Algorithmus anhand von bereits gekennzeichneten Daten, labeled data, trainiert. Somit muss das zu lernende Ergebnis der Trainingsdaten bereits vorhanden sein und das Muster wird vorgegeben.

Der Algorithmus kann anhand der vorgegebenen Muster aus unbekannten Daten Prognosen ableiten. Beispiele hierfür sind Regressionsanalysen, k-nächster Nachbar, Support-Vektor Maschine, Entscheidungsbaum, Random Forests und Neuronale Netze. (Ng et al. 2018, S. 8f.; Aust 2021, S. 57) Das **unüberwachte Lernen (unsupervised learning)** wird genutzt, um mithilfe eines Algorithmus ein unbekanntes Muster in einem Datensatz zu erkennen. In diesem Fall liegen keine gekennzeichneten Trainingsdaten vor. Beispiele für solche Algorithmen sind Clusteranalysen, die Hauptkomponentenanalyse (principal component analysis PCA), Assoziationsanalysen und Soziale Netzwerkanalysen. (Ng et al. 2018, S. 8; Aust 2021, S. 65)

Die letzte Kategorie der Algorithmen im ML bildet das **bestärkende Lernen (reinforcement learning)**. Diese Algorithmen können anhand von neuen Daten kontinuierlich ihre Strategie und somit die Prognosen verändern im Gegensatz zu den Algorithmen des überwachten und unüberwachten Lernens, die nach Abschluss des Antrainierens unverändert bleiben (Ng et al. 2018, S. 10). Aust (2021, S. 70) beschreibt das bestärkende Lernen, dass *„[…] ein Agent mit seiner Umgebung interagiert, indem er gewisse Aktionen zu einer Zeit t durchführt und Rückmeldung durch die Beobachtung der Umgebung erhält. Zudem gibt es eine Belohnungsfunktion, welche maximiert werden soll.“*. Durch die Belohnungs- oder Anreizfunktion erhält der Algorithmus zu definierten Zeitpunkten Rückmeldungen zu seinen gewählten Aktionen und kann so seine Strategie verbessern. Ein Beispiel für eine solche Belohnungsfunktion wäre beim Schachspielen, dass der Algorithmus durch seine gewählten Züge das Spiel gewinnt. Dadurch konnte zum Beispiel der Algorithmus AlphaGoZero von Google in dem asiatischen Brettspiel Go viel besser als Menschen werden (Aust 2021, S. 70). Weitere Beispiele für bestärkendes Lernen sind die Algorithmen A/B-Test und Abnehmendes-Epsilon-Strategie (Ng et al. 2018, S. 8).

Neuronale Netze und Deep learning

Häufig fällt im Zusammenhang mit KI auch der Begriff des **deep learning**. Deep learning beschreibt dabei weniger eine neue Kategorie des ML als vielmehr eine Unterklasse von neuronalen Netzen mit vielen Zwischenschichten (Aust 2021, S. 12; Chollet 2017, S. 6), mit der es möglich ist, auch eine große Menge unstrukturierter Daten zu verarbeiten. Allerdings benötigen diese Algorithmen viel Zeit als auch leistungsstarke Computer, um die Rechenzeit zu beschleunigen und eine größere Menge an Trainingsdaten (Buxmann et al. 2019, S. 12 und S. 145). Ein **künstliches neuronales Netz (KNN)**, artificial neural network, dient der Verarbeitung von Daten in mehreren Abstraktionsschritten in den Schichten (layer) des Netzes. Bestandteile eines KNN sind dabei die Knoten (units), die auf den verschiedenen Schichten liegen und

in denen ein Berechnungsschritt stattfindet, und ihren Verbindungen, durch die das Berechnungsergebnis eines Knoten in einem anderen Knoten weitergegeben wird.

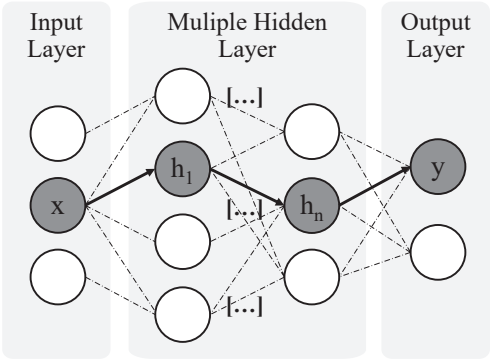


Abbildung 2-14: Neuronales Netz und seine Schichten (Buxmann et al. 2019, S. 14)

Die Anzahl der Knoten und der Schichten bildet die Parameter eines KNN und seine Struktur wird auch als Topologie bezeichnet. Bei den Schichten eines KNN unterscheidet man die Input-Schicht x, eine bis mehrere tausend Zwischenschichten h_n (hidden layers) und der Output-Schicht y (Abbildung 2-14). Um die Daten in den Berechnungsschritten der Knoten verarbeiten zu können, müssen die Daten zunächst in einen Zahlenvektor transferiert werden. (Aust 2021, S. 164-169) Mit jeder Schicht nimmt die Abstraktion zu.

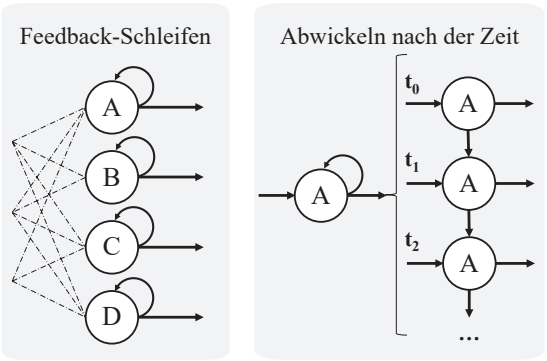


Abbildung 2-15: Prinzip RNN und LSTM (Aust 2021, S. 187)

KNN, bei denen die Werte aus den Berechnungsschritten beginnend mit der Input-Schicht von einer Schicht in die nächste bis zur Output-Schicht weitergegeben werden,

bezeichnet man auch als **Feed-Forward-Netz** (Aust 2021, S. 186). Davon zu unterscheiden sind die **recurrent neural network (RNN)** und **convolutional neural network (CNN)**. RNN weisen Rückkopplungsschleifen auf, in der Knoten sowohl Verbindungen zu anderen Knoten derselben Schicht als auch den vorherigen Schichten aufweisen (Abbildung 2-15). Somit wird jede aktuelle Schicht in seinem Zustand wieder zu einer neuen Input-Schicht für das Netz. (Aust 2021, S. 186) Diese Art von Gedächtnisstruktur eines künstlichen neuronalen Netzes wurde zu **long-short-term memory (LSTM)** weiterentwickelt, in der sich die Knoten an Informationen erinnern oder vergessen können und so dem exploding-gradient Problem entgegenzuwirken (vgl. Hochreiter et al. 1997; Goodfellow et al. 2018, S. 445ff.).

Die letzte Art der neuronalen Netze, die in dieser Arbeit genannt werden soll, sind die **convolutional neural networks (CNN)**. Die Bezeichnung convolution, dt. Faltung, zeigt, dass hier eine spezielle Form der linearen Operation zwischen den Knoten des Netzes erfolgt. Durch diese Faltungsoperation wird ein gewichtetes Mittel berechnet (Goodfellow et al. 2018, S. 369ff.). Dies erfolgt immer bei nah beieinanderliegenden Knoten, die dadurch auf einen Knoten, das gewichtete Mittel, reduziert werden. Schichten, die diese Faltung aufweisen, werden als convolution layer (Faltungsschicht) bezeichnet. (Aust 2021, S. 172)

Diese kurze Einleitung in die verschiedenen Ansätze des ML und der Überblick über die verschiedene Architekturen eines neuronalen Netzes soll zeigen, dass immer abhängig von der zu lösenden Aufgabe (Prognose nach bekannten Muster oder Entdeckung eines unbekannten Musters), die Form und Menge der vorliegenden Trainingsdaten (gekennzeichnete Trainingsdaten, ungekennzeichnete und unstrukturierte Daten, begrenzte Datenmenge oder unbegrenzte Datenmenge) sowie der zur Verfügung stehenden Rechenleistung und Zeit das geeignete Verfahren der Künstlichen Intelligenz gewählt werden muss.

2.3.2 NLP: Textvorbereitung (Preprocessing)

Nachdem im vorherigen Kapitel ein allgemeiner Überblick über KI und ihre Technologien gegeben wurde, soll nun spezifisch auf den Anwendungsfall der Sprachverarbeitung, das **Natural Language Processing (NLP)**, eingegangen werden. Sprache ist das Format, in dem Menschen ihre Informationen und Wissen teilen, sowohl geschrieben als auch gesprochen. Die Verarbeitung von natürlicher Sprache (Kapitel 2.1.2) durch Maschinen fällt dabei in das Fachgebiet der **Computerlinguistik.** Die Computerlinguistik nutzt Verfahren der Künstlichen Intelligenz zur Spracherkennung, gram-

matikalischen Analyse, Generierung von Sprache, Semantik und Wissensrepräsentation von sprachlichem und nicht-sprachlichem Wissen, in denen sich die Fachgebiete der Computerlinguistik und der Künstlichen Intelligenz schneiden. (Carstensen et al. 2010, S. 4) Die Methodenbereiche der Computerlinguistik umfassen dabei analog zur Linguistik die Phonologie, Morphologie, Syntax, Semantik und Pragmatik (Carstensen et al. 2010, S. 3). Ihre Aufgaben sind die Entwicklung von Grammatikformalisierung, Programmen zur Verarbeitung linguistischer Daten und Sprachsoftware als auch der Realisierung maschineller Sprachverarbeitung auf Computern, dem NLP. (Carstensen et al. 2010, S. 2)

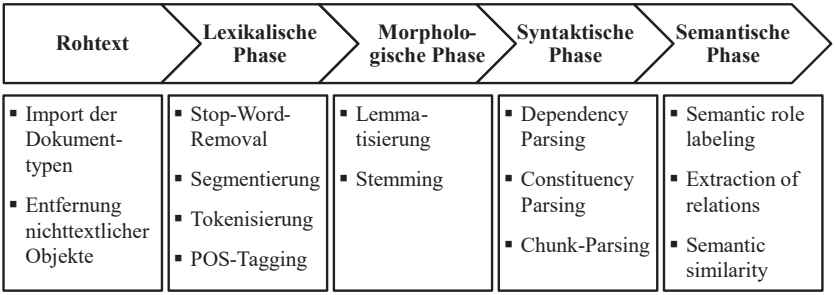


Abbildung 2-16: Phasen des NLP bis zur Semantischen Analyse (eigene Darstellung in Anlehnung an Waltl 2018, S. 12)

Essenziell für die Verarbeitung von Sprachen und Texten durch Algorithmen ist die Vorbereitung der Textkorpora, das **Preprocessing**. Die einzelnen Phasen des Preprocessing und der semantischen Verarbeitung von Texten zeigt Abbildung 2-16. Das Preprocessing erstreckt sich von der Verarbeitung des Rohtextes über eine lexikalische und morphologische Verarbeitung bis zur syntaktischen Phase. Erst nach Durchlaufen dieser Verarbeitungsschritte kann der Text semantisch verarbeitet werden. Die Methoden dieser Vorbereitung werden in Kapitel 2.3.2 vorgestellt. Anschließend werden in Kapitel 2.3.3 Methoden des NLP zur semantischen Analyse von Texten vorgestellt.

Zu Beginn der Sprachverarbeitung liegt das Ausgangsmaterial als unstrukturierte Textkorpora, teils noch mit Bildelementen versehen, vor. Diese Textkorpora müssen für eine maschinelle Verarbeitung zur linguistischen Analyse zunächst so aufbereitet werden, dass nur die sprachlich relevanten Elemente ausgelesen werden. Dazu zählt auch das **Stop-Word-Removal**, die Entfernung s.g. Stoppworte. Diese treten sehr häufig im Textkorpus auf, beinhalten aber kaum bis keine Information. Stoppwörter sind

klassischerweise Artikel, Hilfsverben, Modalverben, Präpositionen und Konjunktionen. (Hoppe 2020, S. 47f.) Diese Stoppwörter können als Stoppwortlisten verwaltet und ggf. je nach Anwendungsfall durch spezifische Wörter ergänzt werden.

Anschließend müssen die so bereinigten Textkorpora in einzelne Einheiten (Wörter, Phrasen, Sätze, Absätze), den **Token**, segmentiert werden. (Carstensen et al. 2010, S. 264). Das Verfahren zur Segmentierung des Textkorpus wird als **Tokenisierung** bezeichnet. (Hoppe 2020, S. 20; Wang et al. 2016, S. 639; Carstensen et al. 2010, S. 264) Für die Segmentierung des Textkorpora werden linguistische Regeln angewandt, um den Anfang und das Ende eines Tokens zu definieren. Entspricht ein Token einem Wort, wird dieses durch Leerzeichen und/oder Interpunktion begrenzt. Werden Worte allerdings durch Bindestriche getrennt oder weisen Apostrophe auf, ist nicht immer eindeutig, was als eine Worteinheit im Sinne der Tokenisierung definiert werden soll. Betrachtet man Sätze, wird es komplexer, denn nicht jedes Satzzeichen bedeutet das Ende eines Satzes und nicht jeder Großbuchstabe markiert einen Satzanfang z.B. bei Abkürzungen. (Carstensen et al. 2010, S. 264ff.) Zur Tokenisierung unterscheidet man symbolische und statistische Verfahren. Bei einem symbolischen Verfahren wird ein manuelles Regelwerk erstellt und durch Abkürzungslisten ergänzt. Die Erarbeitung eines solchen robusten Regelwerks kann sehr aufwendig werden und ist gleichzeitig auf eine Domäne bzw. Sprache spezialisiert. Statistische Verfahren sind hingegen in der Erarbeitung aufwendiger, aber zu neuen Domänen und Sprachen flexibler anzupassen. (Carstensen et al. 2010, S. 267)

Der nächste Schritt im Preprocessing nach Erstellung der Token ist die **Wortartannotierung, part of speech tagging (POS tagging)**. In diesem Schritt werden die einzelnen Token anhand ihrer jeweiligen Wortart (Substantiv, Verb, Adjektiv etc.) gekennzeichnet. Für die deutsche und englische Sprache existieren Systeme, s.g. tagger, die diese grammatikalische Klassifizierung durchführen. (Carstensen et al. 2010, S. 271; Hoppe 2020, S. 41) Im Englischen ist der Stanford Loglinear POS Tagger (Wang et al. 2016, S. 639) und im Deutschen sind das Stuttgart-Tübingen-Tagset (STTS), TreeTagger und spaCy (Hoppe 2020, S. 41) die bekanntesten POS-tagger.

Die bisher genannten Verfahren dienen zur grammatikalischen Strukturierung des Textkorpora und werden auch als **Lexikalische Phase** bezeichnet. Auf sie folgt die **Morphologische Phase**. Der Fokus dieser Phase liegt in der Grundform bzw. in der Struktur des einzelnen Wortes: Präfix, Wortstamm und Suffix. Der Teil des Wortes, der die inhaltliche Bedeutung trägt, wird als **Lexem** bezeichnet. Um dieses zu identi-

fizieren, gibt es zwei Ansätze: Die Lemmatisierung und das Stemming. In der **Lemmatisierung** wird das Wort anhand eines Lexikons auf seine Grundform reduziert z.B. das Verb gegangen → gehen. (Wang et al. 2016, S. 639; Hoppe 2020, S. 49)

Alternativ zur Lemmatisierung kann ein **Stemming** durchgeführt werden. Hierbei wird das Wort auf seinen Wortstamm reduziert z.B. das Verb gegangen auf gegang. Bei diesem Verfahren werden die aufgrund Konjunktion, Deklination, Tempus, Genus, Numerus etc. gebildeten Nachsilben eines Wortes entfernt. Zurück bleibt der Wortstamm. (Hoppe 2020, S. 49) Anhand des Beispiels sieht man den Unterschied dieser beiden Verfahren im Ergebnis: Während in der Lemmatisierung auch das aufgrund des Tempus veränderte Verb auf die Grundform gehen transferiert wurde, wurde im Stemming lediglich die Nachsilbe entfernt, die Tempusform bleibt allerdings erhalten. Problematisch sind die Fehler Over- und Understemming, bei denen zu viel oder zu wenig entfernt wurde. Dadurch können zwei inhaltlich unterschiedliche Wörter auf denselben Wortstamm reduziert werden (z.B. Kommunismus und kommunizieren → kommuni) oder zwei inhaltlich gleiche Wörter als solche nicht erkannt werden (z.B. geht und gegangen → geh und gegang). Der durch diese Verfahren ermittelte Wortstamm ist grammatikalisch oft nicht korrekt. Der bekannteste Stemmer ist der Porter-Stemmer, auch Snowball, der für mehrere Sprachen u.a. deutsch erhältlich ist (Porter et al. 2002).

Auf die morphologische folgt die **Syntaktische Phase**. Die **Syntaxanalyse (Parsing)** hat das Ziel der Darstellung von Wortbeziehungen innerhalb eines Satzes, deren sprachliche Verknüpfung zu Wortgruppen und die Begriffsklärung von ambigen (mehrdeutigen) Ausdrücken. Sie ist für eine anschließende inhaltlichen Interpretationen der Textkorpora erforderlich. (Carstensen et al. 2010, S. 281ff.) Die meisten Ausdrücke sind mehrdeutig, sodass es für eine semantische Analyse erforderlich ist, zu verstehen, worauf sie sich beziehen. Oder es können in einem Satz eine Gruppe aus Wörtern grammatikalisch wie eine Einheit verwendet werden. Zur Syntaxanalyse werden die Ansätze Constituency Parsing und Dependency Parsing unterschieden. Während beim Constituency Parsing ein Satz in einzelne Phrasen aufgeteilt wird, den s.g. Konstituenten als **Nominalphrase NP** (z.B. ein Nomen inklusiver seines Artikels und Adjektivs) oder **Verbalphrase VP** (z.B. ein Verb und eine adverbiale Bestimmung), werden bei einem Dependency Parsing Verbindungen zwischen den einzelnen Wörtern gezogen. (Covington 2001) Ein Beispiel für einen Dependency Parser ist der Stanford Dependency Parser von Chen et al. (2014). Für eine detailliertere Erläuterung des Parsing wird auf Carstensen et al. (2010, S. 304ff.) verwiesen.

Eine Sonderform des Parsing stellen die **Chunk-Parser** dar. Hier werden nur Teilstrukturen, s.g. **chunks** (Chunks 2012), gekennzeichnet im Gegensatz zu jedem einzelnen Konstituenten eines Satzes bei herkömmlichen Parsern. Dadurch sind sie leistungsstärker und genauer, liefern aber nur lokale syntaktische Abhängigkeiten. (Carstensen et al. 2010, S. 276)

Die Herausforderung des Parsing ist, dass es für einen Satz auch mehrere korrekte Syntaxbäume geben kann und diese inhaltlich unterschiedlichen Bedeutungen haben können. Diese Mehrdeutigkeiten bezeichnet man in der Linguistik als **syntaktische Ambiguität**. Ein Beispiel wäre der Satz "Er liest das Buch seiner Schwester vor.", bei dem der Zusatz „seiner Schwester" sich sowohl auf das Buch als auch auf das Lesen beziehen kann. Darüber hinaus gibt es noch weitere Ambiguitäten, die die linguistische Analyse durch Algorithmen erschweren: Unter einer **lexikalischen Ambiguität** werden einzelne Wörter verstanden, die mehrere Bedeutungen haben z.B. Bank als Geldinstitut oder Sitzgelegenheit. Eine **semantische Ambiguität** tritt auf, wenn die Referenz eines Nebensatzes uneindeutig ist (Beispiel 1) oder Quantoren und Verneinungen mehrdeutig sind. (Carstensen et al. 2010, S. 331f.; Gleich et al. 2010, S. 220f.; Berry et al. 2003)

Beispiel 1: „Der Bauer verkauft seinen Ochsen, weil er alt und krank ist." → Beschreibt der Nebensatz hier den Bauern oder den Ochsen?

Beispiel 2: „Alle Bürger haben eine Sozialversicherungsnummer." →Besitzen alle Bürger dieselbe Nummer?

Gleich et al. (2010, S. 220) und Berry et al. (2003, S. 3ff.) erweitern die Beispiele der Ambiguitäten um pragmatische Ambiguität, Vagheit und Sprachfehler. Nach Durchlaufen der lexikalischen, morphologischen und syntaktischen Phase zur Vorbereitung des Textes erfolgt nun in der **Semantischen Phase** die Interpretation des Bedeutungsinhalts des vorliegenden Textes. Da es im Hinblick auf die Anwendung von NLP in dieser Arbeit um den semantischen Vergleich von Anforderungen geht, wird die Semantische Analyse in einem eigenen Kapitel 2.3.3 erläutert.

2.3.3 NLP: Semantik – Analyse und Vergleich

Die **Semantik** ist die inhaltliche Bedeutung eines Wortes, Satzes oder Textes (vgl. Carstensen et al. 2010, S. 12f.). Bei einer semantischen Analyse bzw. einem semantischen Vergleich soll untersucht werden, ob verschiedene lexikalische Einheiten oder größere Textstrukturen inhaltlich die gleiche Bedeutung repräsentieren, obwohl sie sich in ihrer Satzstellung und Wortverwendung unterscheiden (Alabdulkareem et al. 2015, S. 58; Gipp 2014, S. 45f.). Dabei ist die semantische Ähnlichkeit von der lexikalischen Ähnlichkeit zu unterscheiden. Die **lexikalische Ähnlichkeit** misst, wie viele Wörter eines Satzes, Absatzes oder Dokuments identisch sind. So würde der Vergleich von zwei Sätzen mit identischem Satzbau und identischen Vokabeln eine lexikalische Ähnlichkeit von eins ergeben. (Gipp 2014, S. 45f.) Zur Messung dieser Ähnlichkeit ist ein String-Vergleich ausreichend. (Gomaa et al. 2013, S. 13) Komplizierter ist die Messung der **semantischen Ähnlichkeit**, da hierbei auf Konzepte eines Wortes als Hintergrundwissen zurückgegriffen werden muss. Diese Methoden werden in korpusbasierte Ähnlichkeit und wissensbasierte Ähnlichkeit unterteilt. (Gomaa et al. 2013, S. 15)

2.3.3.1 Wissensbasierte Methoden

Die Basis der wissensbasierten Methoden, engl. knowledge based methods, sind Wortnetze. Wortnetze sind lexikalisch-semantische Wortschätze einer Sprache, die elektronisch in Form von Netzen die Beziehungen zwischen einzelnen Wörtern in einer Sprache abbilden z.B. Synonyme, Antonyme, Hyperonyme und Hyponyme. Diese Beziehungen bilden die Bedeutung eines Wortes ab. (Hoppe 2020, S. 118; Carstensen et al. 2010, S. 517; Gomaa et al. 2013, S. 15) Ferner werden solche Wörterbücher auch als Thesaurus bezeichnet. Gabrilovich et al. (2007, S. 1609f.) stellen fest, dass der Aufbau eines solchen Wortnetzes erheblichen Aufwand und lexikalische Expertise erfordert. Daher würden die bestehenden Wortnetze oft nur einen Teil der Sprache abdecken und keine Eigennamen, Neologismen, Slang oder technische Fachbegriffe beinhalten. Beispiele für solche Wortnetze sind WordNet (Fellbaum 1998) und FrameNet (Baker et al. 1998) für die englische Sprache und GermaNet (Hamp et al. 1997) sowie OdeNet (Siegel et al. 2021) im Deutschen.

Zur Bestimmung der semantischen Ähnlichkeit wird bei Wortnetzen auf die Berechnung der Pfadlänge im Wortnetz zwischen den Wörtern zurückgegriffen. Hierfür gibt es viele verschiedene Berechnungsmethoden, die ausführlich von Arweiler (2008) untersucht und erläutert werden.

2.3.3.2 Korpusbasierte Methoden

Für die Bestimmung der semantischen Ähnlichkeit mittels korpusbasierter Methoden werden die Wörter, bzw. der Text, in Vektoren eines hochdimensionalen Vektorraums umgewandelt. Je nach angewandter Methode werden die Vektoren als Repräsentation des Wortes, engl. **word-embeddings**, oder Textes unterschiedlich gebildet. Nach Bildung des Vektors kann die semantische Ähnlichkeit zwischen zwei Wörtern oder Texten bestimmt werden, in dem der Kosinus als Winkel zwischen den Vektoren gemessen wird. Diese Art der semantischen Ähnlichkeit wird als **Kosinus-Ähnlichkeit, engl. cosine similarity**, bezeichnet. (Hoppe 2020, S. 91)

In Abbildung 2-17 werden schematisch zwei Vektoren dargestellt. Vektor $\vec{a}$ zeigt die semantische Abbildung des Wortes A. Vektor $\vec{b}$ zeigt hingegen die semantische Abbildung des Wortes B. Um nun die semantische Ähnlichkeit zwischen diesen beiden Wörtern zu bemessen, wird in dem Vektorraum der Winkel θ zwischen den beiden Vektoren verwendet. Anhand des Kosinus dieses Winkels wird die Ähnlichkeit mittels der Formel (**1**) berechnet.

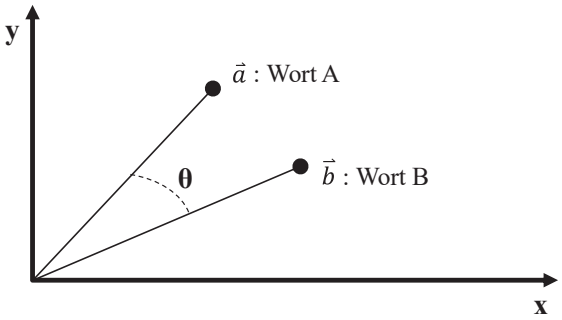


Abbildung 2-17: Semantische Ähnlichkeit als Cosine Similarity

$$similarity = \cos\theta = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| \cdot |\vec{b}|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}} \quad (1)$$

Die Grundidee hinter den korpusbasierten Methoden ist, dass sich die Bedeutung eines Wortes aus dessen Kontexten ergibt, in denen das Wort vorkommt. Daher analysieren diese Methoden, wie oft ein Wort in welchem Kontext vorkommt und wie er in der Sprache verteilt ist (Lund et al. 1996). In der englischsprachigen Literatur werden diese Methoden als co-occurrence techniques bezeichnet, zu denen die Ansätze Bag

of Words, Latent Semantic Analysis oder Hyperspace Analogues to Language zählen (Li et al. 2006, S. 1139).

Die Methode **Bag of Words (BoW)** erstellt eine Liste der relevanten Begriffe eines Textes und zählt deren Häufigkeit. Die Gesamtzahl der Begriffe, die die zu vergleichenden Texte gemein haben, wird durch die kumulierte Summe aller Begriffe der beiden Texte geteilt und ergibt die s.g. Distanz. Je höher die Anzahl der Wörter ist, die beide Texte gemein haben, desto höher ist der Wert der Distanz und umso semantisch ähnlicher sind sich die Texte. Das Problem dieses Verfahrens ist allerdings, das auch Sätze, die eine hohe inhaltliche Ähnlichkeit aufweisen, unterschiedliche Vokabeln und Satzstellungen verwenden können, sodass diese Methode nicht funktioniert. Zum Beispiel haben die beiden Sätze „Es ist dunkel." und „Hier gibt es kein Licht." nur das Wort „es" gemein und somit nur einen sehr geringen Distanzwert. Daher eignet sich BoW weniger für den Vergleich einzelner Sätze als zum semantischen Vergleich längerer Texte und Dokumente. (Aust 2021, S. 210)

Um dieses Problem zu lösen, wurde von Mikolov et al.(2013) eine der am häufigsten verwendeten Embedding Methoden **Word2Vec** entwickelt (Rothe 2016, S. 20). Word2Vec besteht aus einer Weiterentwicklung des BoW, dem Continuous Bag of Words, als auch der Methode Skip-gram. Bei diesem Ansatz wird zunächst ein Vokabular V als Vektorraum definiert, sowie eine Indexfunktion, mit der die Wörter der Trainingsdaten zu einem zugehörigen Wort im Vokabular indiziert und abgebildet werden (Rothe 2016, S. 20ff.). Dadurch wird ein Wort in einem für alle Texte festgelegten Vektorraum abgebildet, sodass beim Vergleichen von den Vektoren zweier Wörter diese bei semantischer Ähnlichkeit nahe beieinander liegen (Aust 2021, S. 166).

Die **Latent Semantic Analysis (LSA)** ist eine von Landauer et al. (1997) entwickelte statistische Methode, in der das gemeinsame Auftreten eines Wortes in einem großen Textkörper genutzt wird. (Gabrilovich et al. 2007, S. 1609f.). Zunächst wird der Text in einer Matrix abgebildet, in der die einzelnen Zeilen jeweils für ein Wort und die Spalten für einen Absatz stehen. In den Zellen stehen dann die Häufigkeiten, mit denen das Wort in dem jeweiligen Absatz vorkommt. Anschließend wird mit der mathematischen Methode singular value decomposition (SVD) die Anzahl der Spalten reduziert. (Landauer et al. 1998, S. 263f.) Hierzu stellen Kao et al. (2007, S. 95) fest, dass die precision mit der Länge der zu vergleichenden Objekte zunimmt, sodass LSA für sehr kurze Dokumente ungeeignet ist.

Analog zu LSA nutzt auch die Methode **Hyperspace Analogues to Language (HAL)** die statistischen Informationen eines Wortes in einem großen Textkörper. Dabei fokussiert die Methode ein Fenster von n Wörtern vor und nach dem analysierten Wort. In dieses Fenster n gehören die Inhaltswörter, also Substantive und Verben. (Lund et al. 1996, S. 204ff.) Im Gegensatz zu LSA (Wort-Absatz-Matrix) wird bei HAL eine Wort-Wort-Matrix aufgebaut (Li et al. 2006, S. 1139). Weitere co-occurrence-Methoden sind die Wikipedia basierte **Explicit Semantic Analysis (ESA)** (Gabrilovich et al. 2007), der auf der Google-Suchmaschine aufbauende **Normalized Google Distance (NGD)** Ansatz (Cilibrasi et al. 2007), **Pointwise Mutual Information - Information Retrieval (PMI-IR)** (Turney 2001) und **Extracting DIStributionally similar words using COoccurrences (DISCO)** (Kolb 2009). Eine detailliertere Gegenüberstellung dieser Ansätze wird in der Untersuchung von Gomaa et al.(2013) dargestellt.

Ergänzend sind noch die **merkmalsbasierten Methoden** wie die von McClelland et al. (1986) zu nennen, bei denen für ein Wort ein Merkmalsvektor (feature vector) gebildet wird. Dazu hat McClelland et al. (1986, S. 279) für Substantive und Verben Merkmale mit unterschiedlichen Ausprägungen definiert. Für Substantive sind dies z.B. human (human, non-human), gender (male, female, neuter) oder object type (food, toy, tool, utensil, furniture, animate etc.) und für Verben doer, cause, touch etc. Li et al. (2006, S. 1140) stellen allerdings fest, dass diese Methode nur für genau definierte Konzepte geeignet ist und die Definition effektiver Merkmale und ihrer Ausprägung sei sehr aufwendig und unpraktikabel.

2.3.3.3 Hierarchical Temporal Memory (HTM) und Semantic Folding Theory (SFT)

Ein von den bislang vorgestellten Methoden abweichender Ansatz zur Analyse semantischer Ähnlichkeit ist die von Webber (2016) entwickelte und durch das Unternehmen cortical.io vertriebene **Semantic Folding Theory (SFT)**, die auf dem **Hierarchical Temporal Memory (HTM)** Modell von Hawkins et al. (2005) basiert. Daher soll zunächst das HTM-Modell erklärt werden und anschließend, wie mit Hilfe des Semantic Folding die semantische Ähnlichkeit bestimmt werden kann.

HTM steht im Deutschen für hierarchischer, temporaler Speicher, dessen Theorie modelliert, wie der menschliche Neokortex Fähigkeiten wie Mustererkennung, natürliche Sprache oder Objekterkennung ausführt. Diese Theorie wurde erstmals von Hawkins (2006) beschrieben und durch die Firma Numenta Inc. weiterentwickelt. HTM wird

von Hawkins et al. (2011, S. 9) als bestimmter Typ von neuronalen Netzen beschrieben, da das Modell Neuronen als Zellen in Schichten (layers), Regionen und Hierarchien strukturiert. Dabei unterscheidet sich HTM durch die Art der Speicherung und Abrufens von Daten von herkömmlichen Modellen, da HTM Speicher Daten hierarchisch und zeitbasiert organisiert, in dem der Algorithmus selbst entscheidet, wo und wie Informationen gespeichert werden. Diese Speicherform ist dabei nicht auf die Abbildung einer bestimmten kognitiven Fähigkeit begrenzt, sondern kann auf eine Vielzahl von Daten angewandt werden. Die Regionen bilden die Hauptspeichereinheit des HTM und sind in Hierarchien angeordnet, die aufsteigend konvergieren und absteigend divergieren anhand von Feedback-Verbindungen. Durch diese hierarchische Organisation soll der Aufwand zum Antrainieren des Modells als auch dessen Speicherverbrauch effizienter werden, da elementare Muster, die auf einer unteren Hierarchieebene bereits antrainiert wurden, auf einer höheren Ebene wiederverwendet werden und dadurch die einzelnen Bestandteile nicht neu antrainiert werden müssen (Hawkins et al. 2011, S. 11ff.). Innerhalb einer Region im HTM-Modell sind die Zellen in Schichten und Kolumnen angeordnet und stellen feed-forward Zellen dar.

Neben der Analogie zwischen dem biologischen Aufbau des menschlichen Neokortex und dem Aufbau des HTM-Modells bildet HTM auch das Merkmal hemmender Neuronen ab. Durch hemmende Neuronen sind im menschlichen Neokortex immer nur ein kleiner Teil von Neuronen gleichzeitig aktiv. Dies bildet HTM durch die s.g. **Sparse Distributed Representation (SDR)** ab, durch die zur Abbildung einer Information immer nur ein kleiner Teil an Zellen der HTM-Region gleichzeitig aktiv (1) ist, der Rest ist inaktiv (0). Das Abbilden eines Inputs in eine SDR ist der erste Schritt einer HTM-Region, sodass bei einem Anteil von 40% aktiver Bits des Inputs nach der Umwandlung in eine SDR nur 2% aktive Zellen verbleiben (Hawkins et al. 2011, S. 25). Dies wird in HTM durch eine Hemmungsfunktion realisiert, durch die eine konstant bleibende Anzahl an Kolumnen einer Region aktiv bleiben. Während Deep learning Ansätze in den Bereich des supervised Machine learning zählen und somit eine große Menge an bereits gekennzeichneten Trainingsdaten benötigen, zählt das HTM-Modell zu den unsupervised machine learning Ansätzen und benötigt nur eine geringe Trainingsdatenmenge.

Die grundlegende Forschungsfrage im NLP ist nach wie vor, wie Sprache in einem menschlichen Gehirn gespeichert und verarbeitet wird. Viele bestehende Ansätze beruhen dabei auf statistischen Methoden (vgl. Kapitel 2.3.3.2 korpusbasierte Ansätze). Die Theorie von Hawkins et al. (2011) orientiert sich dagegen an einer rechnergestützte Theorie zur Modellierung des menschlichen Kortex. Diese Theorie nutzt

Webber (2016) zur Verarbeitung der semantischen Repräsentation von Sprache. Dabei übersetzt die Semantic Folding Theory Textdaten in SDR. Das Antrainieren eines neuen Sprachraums erfolgt in der SFT durch einen unsupervised Ansatz. Dabei wird anhand von Referenzdokumenten des Sprachraums eine **Semantic Map** gebildet. Jedes Bit dieser Semantic Map entspricht dabei einem semantischen Merkmal des repräsentierten Sprachraums. Die Texte der Referenzdokumente zum Antrainieren des Sprachraums werden zunächst in einzelne Textfragmente, s.g. Snippets, zerlegt (Abbildung 2-18). Jedes Textfragment stellt einen einzelnen Kontext dar. Diese Textfragmente der Referenzdokumente werden auf einer 2D-Matrix, der Semantic Map, verteilt. Dabei werden Textfragmente mit einem höheren Anteil gleicher Wörter näher in der Matrix beieinander angeordnet. Die Größe der Textfragmente zur Bildung der Semantic Map kann dabei je nach Anwendungsfall eingestellt werden. Kurze Fragmente bestehend aus 1-3 Sätzen verbinden Synonyme, während größer gewählte Fragmente eher zu Verknüpfungen von Wörtern mit übergeordneten, allgemeineren Konzepten führen.

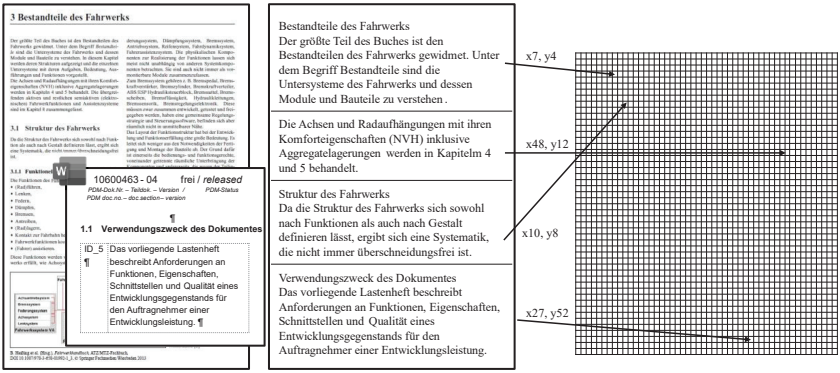


Abbildung 2-18: Erzeugung einer Semantic Map
(Eigene Darstellung mit Ausschnitt aus Heißing 2007, S. 149)

Anschließend wird eine Liste aller in den Referenzdokumenten vorkommenden Wörtern erstellt. Wort für Wort wird anhand der Semantic Map geprüft, in welchen Kontexten das Wort auftritt und dadurch an der zur Semantic Map entsprechende Bitposition in einem 2D-Vektor aktiv (1) gesetzt. Dadurch wird für jedes in den Referenzdokumenten enthaltene Wort ein individueller 2D-Binärvektor gebildet, der **Semantic Fingerprint**.

Diese Wortrepräsentationen als 2D-Vektoren werden in einer Datenbank indexiert. (Webber 2016, S. 31) Anhand dieser Datenbank können für alle Wörter des antrainierten Sprachraums Semantic Fingerprints abgerufen und hinsichtlich ihrer semantischen Ähnlichkeit verglichen werden. Der Vergleich zweier Semantic Fingerprints erfolgt dabei anhand der aktiven Bits: Je höher der Überlappungsgrad der aktiven Bits von zwei Fingerprints, desto semantisch ähnlicher sind sich zwei Begriffe. (Webber 2016, S. 40)

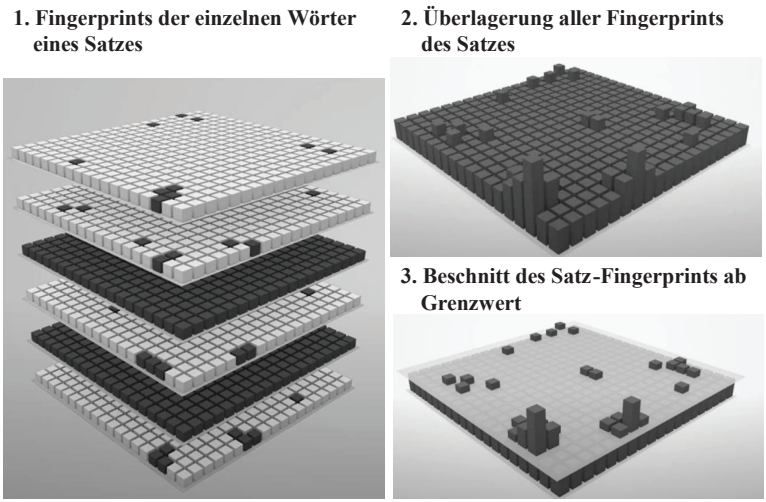


Abbildung 2-19: Erzeugung eines Satz-Fingerprints aus einzelnen Wort-Fingerprints (eigene Darstellung in Anlehnung an Cortical.io 2017, 03:52-04:07 [Video])

Um den Semantic Fingerprint eines ganzen Satzes (bzw. Absatzes oder Dokumentes) zu erstellen, werden die Semantic Fingerprints der einzelnen Wörter dieses Satzes übereinandergestapelt und so die aktiven Bits aggregiert (Abbildung 2-19, Webber 2016, S. 36ff.). Dadurch entstehen bei gemeinsamen Bits Säulen von aktiven Bits. Um weiterhin gemäß des SDR-Ansatzes aus der HTM-Theorie nur wenige Bits aktiv zu halten, werden die Säulen bei einem definierten Grenzwert abgeschnitten. (Webber 2016, S. 36ff.) Durch dieses Abschneiden wird der Kontext des Satzes, in dem ein mehrdeutiges Wort verwendet wird, deutlich. So weist der Begriff Bank als Semantic Fingerprint Bits in den Bereichen Sitzgelegenheit als auch Geldinstitut aktiv auf. Wird allerdings der Semantic Fingerprint des Satzes „Er sitzt auf der roten Bank im Park und sieht den Enten zu.“ gebildet, haben die restlichen Semantic Fingerprints der anderen Wörter des Satzes keinen Bit in den Feldern Geldinstitut aktiv, sodass beim Be-

schneiden der Bit-Säulen der Kontext Geldinstitut entfällt. Diese aggregierten Semantic Fingerprints eines Satzes können so auch auf Satzebene verglichen und dadurch ihre semantische Ähnlichkeit gemessen werden.

2.3.3.4 Pragmatik

Um die Grundlagen der Computerlinguistik, insbesondere des Natural Language Processing, abzuschließen und auf die besondere Herausforderung der Anaphern und Kataphern in der automatisierten Verarbeitung einzugehen, wird an dieser Stelle noch der auf die Semantik folgende Bereich der Pragmatik erläutert. Als Pragmatik bezeichnet man den Gebrauch sprachlicher Ausdrücke in bestimmten Situationen und somit dem kontextabhängigen Sinn der Sprache. Die für diese Arbeit relevanten Phänomene der Pragmatik sind die Anaphorik und Kataphorik. In der Sprache werden von Verfassern und Sprechern häufig Referenzausdrücke verwendet, die sich auf eine bestimmte Entität beziehen. **Anaphern** sind ebensolche Referenzausdrücke, die sich auf eine Entität zurück beziehen, die bereits zuvor durch den Verfasser oder Sprecher ausgedrückt wurden. Diese sind Nominalphrasen, Eigennamen sowie Personal- und Demonstrativpronomen. (Carstensen et al. 2010, S. 399; Ristad 1993) In folgendem Beispiel werden im zweiten Satz zwei Anaphern verwendet: Das Demonstrativpronomen dieser, dass sich auf die im Satz zuvor genannte Theorie bezieht und das Personalpronomen ihm, dass sich auf Max Planck bezieht.

Beispiel Anapher „Max Planck forscht an der Theorie, dass Energie aus Quanten besteht. Basierend auf dieser Annahme wurden von ihm weitere Untersuchungen eingeleitet."

Kataphern referenzieren in die entgegengesetzte Richtung. Ihre Referenz wird erst im nachfolgenden Text aufgelöst. Ähnlich wie bei Anaphern können verschiedene Pronomen als Katapher wirken wie im folgenden Beispiel das Pronomen dessen. Gleichzeitig hat der Doppelpunkt auch eine Funktion als Vorwärtsreferenz auf den nachfolgenden Text.

Beispiel Katapher „Für dessen Entdeckung erhielt Max Planck den Nobelpreis für Physik: Das plancksche Wirkungsquantum.

Für die Auflösung von Anaphern im NLP gibt es für die englische Sprache bereits einige Ansätze. Der **Hobbs-Algorithmus** (Hobbs 1978), das **Fokus-Modell** (Sidner 1983) und das **Centering-Modell** (Grosz et al. 1995; Walker et al. 1998). Während der Hobbs-Algorithmus Semantik und Hintergrundwissen unberücksichtigt lässt, nutzt

das Centering-Modell supervised-learning Ansätze aus dem Bereich der KI und benötigt daher gekennzeichnete Trainingsdaten. Die ML-Ansätze seien insbesondere bei der Pronomenauflösung effektiv. (Carstensen et al. 2010, S. 400-404) Für die Anwendung dieser Ansätze auf die deutsche Sprache gibt es laut Carstensen et al. (2010, S. 407) wenige Untersuchungen wie die von Strube et al. (2002) mit dem Fokus auf Texte aus dem Tourismusbereich, der Untersuchung von Versley (2007) und den TüBa-D/Z-Korpus (Telljohann et al. 2004).

2.3.4 NLP im AM

In diesem Abschnitt soll der aktuelle Stand der Forschung in dem spezifischen Bereich des Einsatzes von NLP-Methoden im Anforderungsmanagement aufgezeigt werden. Die erste Lösungshypothese der vorliegenden Arbeit ist der automatische, semantische Vergleich mit dem Fokus auf deutschsprachige Anforderungen in mechanischen Komponentenlastenheften. Basierend auf diesem Fokus werden die Kriterien zur Bewertung der vorhandenen Literatur abgeleitet: Für den Objektbereich gelten die Kriterien technische Anforderungen, mechanische Komponenten und Deutschsprachigkeit, die Lösungshypothese wird durch das Kriterium semantische Ähnlichkeit bewertet. Die vollständige Bewertung ist in Anhang C dargestellt. Hieraus ergeben sich vier Ansätze mit der höchsten Punktzahl, deren Bewertung im Kriterium semantische Ähnlichkeit ungleich null ist. Auf diese vier Ansätze soll im Folgenden eingegangen werden:

Jörg (2005) – ReMaS und GermaNet

Das von Jörg entwickelte Requirements Management System ReMaS hat das Ziel, Anforderungen mit CAD-Modellen und somit dem PDM-System zu verknüpfen. Anhand linguistischer Analysen sollen die Hierarchie der Anforderungen und ihrer Abhängigkeiten und Beziehungen untereinander in einem semantischen Anforderungsnetz und Constraints dargestellt werden. Als Constraints bezeichnet Jörg Einschränkungen und Zwangsbedingungen, die mathematisch beschrieben den Lösungsbereich einschränken (Jörg 2005, S. 97ff.). In das ReMaS integriert ist eine Anforderungsbibliothek zur Wiederverwendung von Anforderungen, in der Jörg ausgeprägte und nicht ausgeprägte Anforderungen unterscheidet (Jörg 2005, S. 107). Die linguistische Analyse besteht aus einer Zerlegung des Fließtextes in einzelne Anforderungen und einer maschinellen Übersetzung mittels der Software Personal Translator zur Vereinheitlichung von Synonymen und Fachbegriffen sowie dem wissensbasierten Ansatz GermaNet. Die in GermaNet abgebildeten Synonyme und Homonyme werden zur Standardisierung von Begriffen genutzt und die in GermaNet dargestellte Begriffshierar-

chie zur Ableitung der Anforderungsbeziehungen in einem semantischen Anforderungsnetz herangezogen. Zur Implementierung muss zunächst ein benutzerspezifisches Wörterbuch definiert werden. Das Konzept ReMaS wurde anhand deutschsprachiger Spezifikationen für eine gepanzerte Autotür und einen Verbrennungsmotor bei der Mercedes-Benz AG verifiziert (Jörg 2005, S. 157ff.).

Jörg fokussiert in seiner Arbeit hauptsächlich auf Anforderungen mit quantifizierbaren Parametern, um diese im Rahmen seiner Verknüpfung mit CAD-Modellen zu nutzen. Eine konkrete Auswertung der Zerlegung des Fließtextes in einzelne Anforderungen fehlt. In der Schlussfolgerung wird festgestellt, dass die Zerlegung aufwändige Analysekomponenten bedarf (Jörg 2005, S. 178) und grundsätzlich angenommen wird, dass eine Anforderung in einem einzelnen sprachlichen Satz abgebildet wird (Jörg 2005, S. 181). Auf die Problematik der Anaphern wird nicht eingegangen. Ebenso wird der Anteil nicht-parametrisierter Anforderungen nicht weiter erläutert. In seinem Fazit stellt Jörg fest, dass die linguistische Analyse nicht mit der ursprünglichen Zielsetzung umgesetzt werden konnte. Probleme haben hier insbesondere die branchen- und firmenspezifischen Terminologien in den Anforderungen verursacht, sodass für zukünftige linguistische Analysen die gebräuchlichen Tools der Linguistik auf die Bedürfnisse der Anforderungsanalyse angepasst werden müssten. Zusätzlich sei der Aufwand zur Integration der linguistischen Tools sehr umfangreich. (Jörg 2005, S. 178)

Körner (2014) – RECAA

RECAA (Requirements Engineering Complete Automation Approach) besteht aus dem Einsatz der Werkzeuge RESI, Auto Annotator und REFS mit dem Ziel, die Anforderungserhebung in der Softwareentwicklung zu automatisieren. RESI steht für Requirements Engineering Specification Improver und unterstützt den Analysten, indem die in natürlicher Sprache formulierten Anforderungen auf Fehler, sprachliche Mängel und Unklarheiten analysiert werden. RESI verwendet dabei wissensbasierte Ansätze wie WordNet und ConceptNet. Außerdem werden herkömmliche Werkzeuge des Preprocessing wie PoS-Tagging und Stemming verwendet. Dabei stellt Körner (2014, S. 63) fest, dass sich die Leistung mit einer speziell für die Domäne erarbeitete Wissensbasis, Ontologie, verbessert. In einem nächsten Schritt werden durch das Werkzeug Auto Annotator den Anforderungen thematische Rollen zugeordnet. Die Annotierung basiert dabei auf dem Ansatz SALEMX von Gelhausen (2010), durch das aus natürlichsprachigen Anforderungen UML-Modelle erzeugt werden. Der Auto Annotator soll diese bei Gelhausen manuell und zeitaufwendig definierten Annotationen automatisiert erkennen und annotieren. Hierbei wird wieder eine Kette aus Preproces-

sing Methoden wie PoS-Tagging und Parsing verwendet sowie WordNet und ResearchCyc. Körner (2014, S. 88) erläutert hierbei auch die Auflösung von Referenzen (Anaphern), die er durch eine Kombination aus JavaRap, Benennungen und Wortreferenzen umsetzt. Diese Ansätze fokussieren ausschließlich englischsprachige Texte. Das Requirements Engineering Feedback System (REFS) ermöglicht eine bidirektionale Verbindung zwischen dem UML-Modell und den Anforderungen, sodass bei Änderungen eine Rückkopplung stattfindet.

Der Einsatzbereich dieser Methode ist die Anforderungserhebung und fokussiert ausschließlich englischsprachige Anforderungen und die Softwareentwicklung. Die verwendeten Methoden wie WordNet oder die Werkzeuge zur Auflösung von Referenzen stehen nur für die englische Sprache zur Verfügung. Eine Übertragbarkeit des Auto Annotators auf andere Sprache lässt Körner (2014, S. 165) offen. Ein semantischer Vergleich von Anforderungen erfolgt in dieser Methode nicht. In seinem Ausblick bemerkt Körner (2014, S. 179), dass insbesondere bei der Auflösung von Referenzen im Hinblick auf Anforderungen noch weiterer Forschungsbedarf besteht, da bestehende Ansätze sich hauptsächlich auf die Auflösung von Personalpronomen konzentrieren.

Arora et al. (2015) - CIA

CIA steht für Change Impact Analysis und hat das Ziel, den Einfluss von Änderungen von Produktanforderungen auf natürlichsprachige Anforderungen zu analysieren. Der Ansatz verwendet dabei NLP-Techniken wie das Constituency Parsing und Tokenisierung. In den durch das Parsing identifizierten Phrasen werden alle darin enthaltenen Token paarweise miteinander verglichen. Für diesen Vergleich untersuchen Arora et al. (2015) sowohl syntaktische als auch semantische Vergleiche unter Verwendung des SEMILAR Toolkits von Rus et al. (Rus et al. 2013). Die Methode wird anhand eines Anforderungsdokuments einer Satellitensoftwarekomponente der Firma SES TechCom. sowie eines Anforderungsdokuments für eine Mobile Service Plattform überprüft. Für das Anforderungsdokument für Satellitensoftware wurden die besten Ergebnisse mit einem syntaktischen Vergleich mittels Levenshtein similarity erzielt, wohingegen im Fall der Mobile Service Plattform mit einem semantischen Vergleich über die Pfadlänge (Kapitel 2.3.3.1) bessere Ergebnisse erreicht wurden. Die Levenshtein similarity berechnet die Anzahl der erforderlichen Ziffernänderungen auf String-Basis, die erforderlich ist, um ein Wort in ein anderes zu transformieren (Manning et al. 2009, S. 53ff.).

Die Autoren empfehlen in ihrem Fazit eine Kombination beider Vergleiche. Die beiden Anwendungsfälle zeigen, dass dieser Ansatz sich auf englischsprachige Softwareanforderungen fokussiert. Es wird nicht erläutert, wie die einzelnen Anforderungen aus den Anforderungsdokumenten extrahiert und vereinzelt werden.

Cybulski et al. (2000) - RARE und IDIOM

Cybulski et al. (2000) beschreiben in ihrem Ansatz RARE (Reuse-Assisted Requirements Engineering) eine Methode zum Vergleich einer neuen Anforderung mit bestehenden Anforderungen im Hinblick auf eine potentiellen Wiederverwendung dieser. Hierbei konstatieren Cybulski et al. (2000, S. 196), dass ein rein lexikalischer Vergleich der Anforderungen nicht erfolgreich sein könne, da die semantische Ähnlichkeit der Begriffe nicht berücksichtigt wird. Sie schätzen wissensbasierte Ansätze wie die Verwendung eines domänenspezifischen Thesaurus als erfolgsversprechender ein. Einen solchen Thesaurus verwenden sie, um die zuvor aus den Anforderungen extrahierten Schlüsselbegriffe in Begriffe einer Facettenklassifikation zu übersetzen. In einer solchen Facettenklassifikation wird ein Objekt anhand voneinander unabhängiger Begriffe klassifiziert. Anhand dieser Facettenklassen wird anschließend der Verwandtschaftsgrad zwischen Anforderungen bestimmt. (Cybulski et al. 2000, S. 207f.) Dieser Ansatz wird in Form eines Softwaretools IDIOM umgesetzt. Allerdings betrachtet RARE ausschließlich englischsprachige Anforderung in der Softwareentwicklung. Des Weiteren wird der Ansatz nicht auf Fließtext, sondern auf vereinzelte Anforderungen angewendet, sodass keine Zerlegung beschrieben wird. Die Güte des Ansatzes wird ebenfalls nicht weiter erläutert.

2.4 Handlungsbedarfe und Abgrenzung zu vorhandenen Ansätzen

Die vorangegangenen Kapitel haben die Begriffe der drei Themenbereiche Anforderungsmanagement in der Produktentwicklung, Variantenmanagement und dem Natural Language Processing eingeführt und definiert. Durch die Erläuterung dieser Grundlagen wurde ein allgemeines Verständnis für die in dieser Arbeit verwendeten Ansätze geschaffen. In allen drei Themenbereichen wurden die Herausforderungen in der Praxis deutlich: die wachsende Anforderungsmenge in hochkomplexen Produkten, die steigende Variantenvielfalt von Produkten und Komponenten durch interne und externe Variantentreiber als auch im NLP die Herausforderung, Anforderungen in natürlicher Sprache in Fließtextdokumenten automatisiert zu verarbeiten.

Gleichzeitig wurden in jedem der Themenbereiche Ansätze aus der Forschung im Hinblick auf die Themenstellung dieser Arbeit bewertet und vorgestellt. Der Objektbereich dieser Arbeit sind deutschsprachige, technische Anforderungen an mechanische Komponenten. Die Lösungshypothesen sind einerseits der automatisierte Anforderungsvergleich zur semantischen Ähnlichkeit und andererseits variable Anforderungen als Variantentreiber. Das Ziel, der in Rahmen dieser Arbeit entwickelten Methode, ist die Unterstützung der Wiederverwendung von Anforderungen.

Bei der Untersuchung vorhandener Ansätze zur Wiederverwendung von Anforderungen wurde in Kapitel 2.1.3 gezeigt, dass obwohl in der Literatur viele Ansätze zu einem strukturierten Vorgehen durch Pattern, Templates oder Anforderungsbibliotheken beschrieben werden, dass in der Praxis am häufigsten eingesetzte Vorgehen Copy-and-Paste ist. Die in der Theorie umfangreich beschriebenen Ansätze setzen sich laut den Studien von Darimont et al. (2017), Palomares et al. (2014) und Chernak (2012) in der Praxis nicht durch, da sie die Bedürfnisse der Anwender in der Praxis nicht treffen. Die meisten Ansätze teilen eine Klassifizierung von Anforderungen in gemeinsame und variable oder änderbar und unveränderliche Anforderungen. Dies zeigt, dass die Strukturierung von Anforderungen zur Wiederverwendung eng mit einem Variantenmanagement für Anforderungen verbunden ist. Allerdings beschreiben die wenigsten Ansätze das Vorgehen zur Klassifizierung der Anforderungen oder beschreiben ein manuelles Vorgehen (vgl. Renner 2007; Mamrot et al. 2013; Stoiber et al. 2010).

Der Ansatz von Boutkova (2014) zeigt einen semi-automatischen Ansatz, um basierend auf automatisch identifizierten Merkmalen Anforderungen in einer Datenbank durch Merkmalsbäume und Variantenkriterien zu klassifizieren. Ebenso beschreiben

Knauss et al. (2014) einen semi-automatischen Ansatz zur Klassifizierung von Anforderungen hinsichtlich Themen und Überschriften. Andere Ansätze beschreiben den manuellen Aufbau einer Ontologie, eines semantischen Netzes oder Wissensbasis als Modell, um daraus die Anforderungen klassifizieren zu können (vgl. Gülke 2014; Karatas et al. 2014; Stechert 2010; Das et al. 2021). Der manuelle Aufbau einer solchen Ontologie bedarf einen erheblichen Aufwand. Einen Ansatz zur automatisierten Klassifizierung von Anforderungen in die in der Literatur zur Wiederverwendung geforderten Gruppen projektübergreifend, -spezifisch oder ausgeprägt wurde nicht gefunden. Die in der Praxis vorliegende große Anforderungsmenge erfordert allerdings einen automatisierten Ansatz zur Klassifizierung. Daraus leiten sich die ersten Forschungsfragen der vorliegenden Arbeit ab:

(1.1) Können Anforderungen basierend auf einem automatisierten Vergleich in die Gruppen projektübergreifend, ausgeprägt oder spezifisch klassifiziert werden?

(1.2) Ist durch die Klassifizierung eine produktspezifische Konfiguration eines Lastenheftes aus einer Anforderungsbibliothek einer Komponente möglich?

Anschließend wurden in Kapitel 2.2.2 die bekannten internen und externen Variantentreiber erläutert sowie in Kapitel 2.2.3 die in der Literatur beschriebene Ansätze zur Identifizierung von variantenverursachenden Anforderungen als auch zur Modellierung der Variabilität von Anforderungen dargelegt. Die Arbeiten von Renner (2007) und Mamrot et al. (2013) unterstützen die Lösungshypothese dieser Arbeit, dass eine Spreizung von Anforderungen und somit die Unterschiedlichkeit von Anforderungen zu Varianten führen.

Almefelt et al. (2006, S. 128) stellen im Rahmen ihrer Studie bei einem schwedischen Automobilhersteller und Zulieferern die Frage, in welchem Umfang es überhaupt möglich sei, gleiche Komponenten zu entwickeln bei gleichzeitig unterschiedlichen Anforderungen in Abhängigkeit der Marke? Gleichzeitig beschreiben die Ansätze entweder eine erforderliche Vereinheitlichung einer solchen Spreizung durch Optimierung und Harmonisieren (Renner 2007) oder eine Priorisierung bestimmter Ausprägungen über Anwendungsfälle (Stechert 2010) oder über die Identifizierung eines Anforderungssets bestehender Produkte mit einer möglichst großen Überlappung mit den neuen Anforderungen (Mamrot et al. 2013).

Die vorgestellten Ansätze zeigen nicht, wie variantenverursachende Anforderungen identifiziert werden können. Einzig zeigt der Ansatz MIA von Boutkova (2014) eine Möglichkeit zur Identifizierung von Variantentreibern auf: Über die Modellierung der Variabilität von Anforderungen in Merkmalsbäumen werden in ihrem Ansatz MIA

zunächst Merkmale in den Anforderungen automatisiert identifiziert, die sie ferner als Variabilitätskriterien in ihren Variabilitätsmodellen nutzt. Der automatisierte Ansatz weist je nach Datensatz eine precision zwischen 0,03 und 0,22 und einem recall zwischen 0,31 und 0,89 (Boutkova 2014, S. 59) auf, sodass die Ergebnisse anschließend immer manuell überprüft und erweitert werden müssen und hier nur von einem semiautomatischen Ansatz die Rede ist. Des Weiteren stellen Mamrot et al. (2013, S. 351) fest: *„In the variant management a comparison of the requirements has not been performed so far.”.*

Basierend auf den Lösungshypothesen eines Anforderungsvergleichs zur automatisierten Klassifizierung variabler Anforderungen und variablen Anforderungen als Variantentreiber ergeben sich zwei weitere Forschungsfragen dieser Arbeit:

(2.1) Sind spezifische oder ausgeprägte Anforderungen variantenverursachend?

(2.2) Können durch einen automatisierten Anforderungsvergleich mit anschließender Klassifizierung variantenverursachende Anforderungen identifiziert werden?

Die Automatisierung einer Methode zur Analyse von natürlichsprachigen Anforderungen erfordert Ansätze aus dem Bereich des Natural Language Processing. Diese wurden in Kapitel 2.3 erläutert. Der erste Schritt der automatisierten Analyse ist die Zerlegung eines Fließtextes in einem Anforderungsdokument in einzelne Anforderungen. Das Vorgehen sowie die Regeln zur Zerlegung des Fließtextes in einzelne Anforderungen und was eine Anforderung von der nächsten abgrenzt, wird in den untersuchten Ansätzen des Einsatzes von NLP im Anforderungsmanagements nicht erläutert (Anhang C: Bewertung Literatur - NLP Ansätze im Anforderungsmanagement). Jörg (2005, S. 178) bestätigt, dass die Zerlegung eine aufwendige Analyse erfordert. Gleichzeitig nimmt er an, dass jeweils ein sprachlicher Satz einer Anforderung entspricht. Dabei zeigt die in Kapitel 2.3.3.4 beschriebene Anaphorik und Kataphorik die Problematik der inhaltlichen Verweise und Referenzen auf vorangehende oder nachfolgende Sätze, die für das inhaltliche Verständnis einer Anforderung in einem Vergleich miteingeschlossen werden müssen. Zusätzlich beschreibt die Grundlagenliteratur des NLP für die Tokenisierung, der Abgrenzung eines Wortes oder Satzes, den Aufwand zur Erstellung von Heuristiken, um zu definieren, wie ein Text in einzelne zu betrachtende Einheiten zerlegt werden muss (Carstensen et al. 2010, S. 267). Die Literatur zum NLP betrachtet dabei die spezifischen Strukturen einer Anforderung in einem Anforderungsdokument nicht. Zusätzlich stellt Boutkova (2014, S. 60) fest, dass auch die Kapitelüberschrift des Anforderungsdokuments einen wichtigen inhalt-

lichen Kontext für die einzelne Anforderung gibt, sodass die Überschrift bei der Zerlegung des Fließtextes in alle darunter liegende vereinzelte Anforderungen einbezogen werden muss. Hieraus ergibt sich die nächste Forschungsfrage hinsichtlich der Zerlegung:

(3.1) Wie können durch definierte Zerlegungsregeln im Preprocessing einzelne Anforderungen voneinander aus einem Fließtext abgegrenzt werden?

Nach erfolgter Zerlegung der Anforderungsdokumente in einzelne Anforderungen sollen diese paarweise verglichen werden, um die Unterschiedlichkeit und Spreizung der Anforderungen zu analysieren und dadurch die Klassifizierung vornehmen zu können. Hierzu stellen Körner (2014, S. 53), Cybulski et al. (2000, S. 195f.) und Kickermann (1995, S. 121f.) fest, dass ein lexikalischer Vergleich von Anforderungen fehlschlagen muss, da die semantische und somit inhaltlich konzeptuelle Bedeutung von Anforderungen nicht berücksichtigt wird. Gelhausen (2010, S. 13) bemerkt, dass so gut wie keine real existierende Anforderungsdokumente eine so simple Grammatik vorweisen, dass bestehende Ansätze diese ohne weiteres verarbeiten können. Daher wurden ausschließlich Ansätze zur Analyse der semantischen Ähnlichkeit untersucht. Bei der Auswahl geeigneter Ansätze zur semantischen Analyse müssen die vorliegenden Daten und somit der Objektbereich dieser Arbeit berücksichtigt werden: deutschsprachige, technische Anforderungen an mechanische Komponenten. Interessant ist dabei die Feststellung von Körner (2014, S. 54), dass *„Bis heute […] industrielle Anwendungen wie RationalRose, CaliberRM, CaseComplete, HP QualityCenter, DOORS oder RequisitePro […] keine Lösungsansätze zur Verarbeitung natürlicher Sprache aus der Forschung übernommen [haben].“*. Dies kann im Rahmen der Untersuchung geeigneter Vergleichsansätze in dieser Arbeit bestätigt werden.

Der Fokus auf mechanische Komponenten bringt einige Einschränkungen der vorhandenen Ansätze mit sich. Ein Großteil der existierenden Literatur im Anforderungsmanagement stammt aus dem Bereich der Softwareentwicklung und entsprechend wenden die mehrheitlichen Ansätze NLP-Techniken auf Anforderungen an Software an. Diese Ansätze haben i.d.R. das Ziel einer automatisierten Prozessmodellierung z.B. in UML (Gelhausen 2010; Körner 2014) oder einer Verbindung zwischen Veränderungen im Programmcode und der zugehörigen Anforderung (Arora et al. 2015). Anforderungen an mechanische Komponenten haben i.d.R. einen statischen und keinen prozessualen Charakter im Sinne eines Softwarecodes. Daher lassen sich Ansätze für Softwareanforderungen schlecht auf mechanische Komponenten übertragen.

Durch die Einschränkung auf deutschsprachige Anforderungen sind viele vorhandene Ansätze nicht praktikabel. Wie bereits Gelhausen (2010, S. 13) feststellt, sind für englische Texte entwickelte Ansätze aufgrund anderer Wortstellungen, Satzkonstruktionen und der wesentlich geringeren Beugung/Flexion nicht ohne weiteres auf die deutsche Sprache übertragbar. Als Beugung bezeichnet man in der Linguistik die Anpassung der Wortform, z.B. Deklination und Konjugation, an Numerus, Tempus, Modus, Genus, Kasus etc. Aus diesem Grund greifen viele veröffentlichte Ansätze auf Ontologien wissensbasierter Ansätze wie GermaNet zurück (Körner 2014, S. 54; Jörg 2005). Der Aufbau eines lexikalischen Wortnetzes wie GermaNet mit der Abbildung einer kompletten Sprache und sämtlicher Wortbeziehungen untereinander ist sehr aufwendig. Daher greifen die veröffentlichten Ansätze auf bestehende Ontologien zurück. Das Problem hierbei stellen aber die technischen Anforderungen dar. Die betrachteten Anforderungen weisen einen hoch technologischen, kontextspezifischen und teils sogar unternehmensspezifischen Sprachgebrauch auf, der durch das bestehende GermaNet nicht abgebildet wird. Sämtliche technologischen Fachbegriffe sowie unternehmensspezifischen Bezeichnungen müssten zunächst identifiziert, ihre lexikalischen Beziehungen analysiert und in einer erweiterten Ontologie modelliert werden. Dieser Aufwand ist sehr groß, zumal er in regelmäßigen Abständen wiederholt werden müsste, da bei Technologiesprüngen und neuen Entwicklungsfeldern wieder neue Begrifflichkeiten hinzukommen. Somit scheiden aufgrund des Objektbereichs technische Anforderungen wissensbasierte Ansätze aus der weiteren Betrachtung aus.

Allerdings sind auch viele der korpusbasierten Ansätze nicht für den Anwendungsfall semantischer Vergleich einzelner Anforderungen geeignet. Diese im Englischen co-occurrence Techniken basieren auf dem Konzept, dass die Bedeutung eines Wortes sich aus dessen Kontexten ergibt, und somit analysieren, wie oft ein Wort in welchem Kontext vorkommt und wie er in der Sprache verteilt ist. Kao et al. (2007, S. 95) und Aust (2021, S. 210) stellen fest, dass die Genauigkeit solcher Ansätze wie BoW und LSA mit der Länge der zu vergleichenden Objekte zunimmt und für kurze Objekte ungeeignet sind. Des Weiteren benötigen solche Ansätze zum Antrainieren des Sprachmodells eine große Trainingsdatenmenge, um die statistischen Verteilungen der Begriffe und somit ihre kontextuelle Bedeutung zu erfassen. Für den allgemeinen Sprachgebrauch stehen den Ansätzen auch große Datenmengen online zur Verfügung (z.B. Wikipedia etc.). Im spezifischen Fall deutschsprachige Anforderungen in Komponentenlastenheften ist die Auswahl der Trainingsdaten wesentlich eingeschränkter. Daher verspricht der Ansatz Semantic Folding von Webber (2016) ein anderes Vorgehen zur Darstellung der inhaltlichen Konzepte eines Wortes sowie einer Anforderung

und somit einen abweichenden Ansatz im Anwendungsfall semantischer Vergleich von Anforderungen im Vergleich zu den bestehenden wissensbasierten oder korpusbasierten Methoden. In dieser Arbeit wird daher der Ansatz Semantic Folding für den speziellen Fall des semantischen Anforderungsvergleichs mit der folgenden Fragestellung untersucht:

(3.2) Kann eine Software inhaltlich ähnliche, aber unterschiedlich formulierte Anforderungen erkennen und einander zuordnen?

Vor dem Hintergrund dieser sechs Forschungsfragen wurde ein Konzept für einen automatisierten Anforderungsvergleich zur Identifizierung variantenverursachender Anforderungen entwickelt, dass in Kapitel 3 vorgestellt wird. In Kapitel 4 erfolgt eine Auswertung der erzielten Ergebnisse sowie eine abschließende Diskussion der Forschungsfragen basierend auf den Erkenntnissen dieser Arbeit.

3. Methode und Konzept

Die hier vorgestellte Methode soll einerseits variantenverursachende Anforderungen identifizieren, andererseits soll durch die Methode eine Gruppierung von Anforderungen für die Erzeugung einer Anforderungsbibliothek zur Wiederverwendung von Anforderungen in Folgeprojekten ermöglicht werden. Dieses Kapitel beschreibt das Konzept der Methode, die technische Umsetzung der Automatisierung des Vergleichs und ein Ansatz zum Aufbau einer Anforderungsbibliothek aus den Ergebnissen des Anforderungsvergleichs.

3.1 Grundlegendes Konzept

Die zugrunde liegende Idee des Konzeptes zur Identifizierung von variantenverursachenden Anforderungen beruht auf der Hypothese, dass Unterschiede in den Anforderungen zu unterschiedlichen Merkmalen des Produktes bzw. einer Komponente des Produktes führen und dadurch eine Varianz des Produktes oder einer einzelnen Komponente verursachen. Daher liegt die Basis der Methode in einem inhaltlichen Vergleich von Anforderungen aus verschiedenen Entwicklungsprojekten. Durch den Vergleich sollen Unterschiede in den Anforderungen identifiziert und diese hinsichtlich unterschiedlicher Merkmale an dem Produkt des Entwicklungsprojektes überprüft werden können.

Insbesondere bei komplexen Produkten werden für einzelne Bauteile bzw. Komponenten eigene Anforderungsdokumente erstellt. Für den in dieser Methode beschriebenen Vergleich sollen diese Komponentenlastenhefte betrachtet werden. Dabei werden Anforderungstexte und -dokumente derselben Komponente aus verschiedenen Entwicklungsprojekten bzw. Produkten auf Anforderungsebene miteinander verglichen. Anforderungen, die nicht als Text, sondern als Abbildungen, Diagramme oder in technischen Zeichnungen dargestellt sind, können mit der hier beschriebenen Methode nicht verglichen werden.

Die Anforderungstexte können in verschiedenen Formaten vorliegen. Die am weitesten verbreiteten Dokumentationsformen sind Lastenhefte in einem .docx oder .pdf Format sowie Anforderungslisten in Tabellenform in .xlsx-Format. Des Weiteren können Anforderungen auch in Anforderungsdatenbanken dokumentiert sein wie z.B. in den Systemen DOORS von IBM oder codeBeamer der Firma Intland. Diese Datenbanken weisen jeder einzelnen Anforderung eine ID zu, anhand deren die Anforderung identifiziert und von anderen abgegrenzt werden kann. Im Gegensatz dazu müssen bei Vorliegen eines Fließtextes in Lastenheften Anforderungen zunächst noch voneinander

abgegrenzt und vereinzelt werden. Dabei sollen Anforderungen als Textfragmente nach der Zerlegung bei allen Dokumentationsformen eine gleiche Textstruktur-Ebene aufweisen. Das heißt, es sollen nicht Absätze mit Stichpunkten oder Sätze mit ganzen Dokumenten verglichen werden, sondern alle Textfragmente auf der gleichen Ebene verglichen werden. Allerdings sind Ausnahmen zu berücksichtigen, bei denen mehrere Sätze oder ein Stichpunkt mit einem Satz als eine Anforderung zu betrachten ist. Werden die Anforderungen paarweise verglichen, kann ermittelt werden, in wie vielen Entwicklungsprojekten mit dem zugehörigen Lastenheften die jeweilige Anforderung vorkommt. Basierend auf dem Vorkommen sowie der in der Anforderung beschriebenen Quantität können in Anlehnung an Karatas et al. (2014) (Kapitel 2.1.3) folgende drei Gruppen von Anforderungen unterschieden werden:

Anforderung	Gruppe	Projekte				
		A	B	C	D	E
Die Verschraubungen zur Karosserie dürfen im Crashfall nicht versagen.	projekt-übergreifend					
Die Anlage- und Funktionsflächen müssen frei von PVC sein.	spezifisch					
Die maximale Einpresskraft der Gummilager ist ___kN.	ausgeprägt	35	12	30	50	45

Abbildung 3-1: Auswertung Anforderungsvergleich - Häufigkeit und Gruppierung

Projektübergreifende (Basis-) Anforderungen:
Anforderungen, die in allen Entwicklungsprojekten bzw. Produkten vorkommen, werden als projektübergreifende Basisanforderungen klassifiziert. (vgl. Mayer-Bachmann 2008, S. 15) Sie weisen keinen Unterschied zwischen den Projekten auf und sind projektunabhängig. Da sie keine Unterschiede aufweisen, werden sie als nicht variantenverursachend betrachtet.

Ausgeprägte Anforderungen:
Ausgeprägte Anforderungen können in allen oder auch nur in einzelnen Projekten vorkommen. Sie zeichnet aus, dass sie entweder eine zu quantifizierende Größe oder ein Merkmal in einer bestimmten Ausprägung beschreiben. Zu quantifizierende Größen können durch reine Zahlenwerte z.B. bei Jahreszahlen oder durch Zahlenwerte mit anschließenden Einheiten erkannt werden. Merkmale in einer bestimmten Ausprägung können Farben, Materialien, Antriebsarten, Länder, Verbindungsarten, Verschraubungsklassen etc. sein (Tabelle 3-1), die zwar in jedem Projekt als Anforderung vorkommen, aber in einer unterschiedlichen Ausprägung dieses Merkmals festgelegt sein

können. (Ponn et al. 2011, S. 39; Kickermann 1995, S. 70) Durch die projektabhängige unterschiedliche Ausprägung können diese Anforderungen Varianz der Komponente erzeugen.

Projektspezifische Anforderungen:
Als projektspezifisch werden Anforderungen betrachtet, die nur in einzelnen Projekten vorkommen und kein ausgeprägtes oder quantitatives Merkmal aufweisen. Projektspezifische Anforderungen stellen projektabhängige Sonderfälle dar, die auch z.B. über mehrere Produkte einer Produktlinie vorkommen (vgl. Mayer-Bachmann 2008, S. 15). Da sie nur vereinzelt vorkommen und sich somit von Anforderungsdokumenten anderer Projekte unterscheiden, stellen sie mögliche Variantenverursacher dar.

Einheiten	Farben	Material	Verbindung	Schrauben	Antrieb	Regionen/ Länder
°C	Grau	Kunststoff	Schelle	Außen-sechskant	Allradantrieb	Weltweit
kN	Anthrazit	Stahl	geclipst	M10	Heckantrieb	EU
kg	Rot	Aluminium	Schweißnaht	Kreuz-schlitz	Frontantrieb	China
Jahre	Blau	Gusseisen	Bolzen	M8	getrieben	USA
N/mm^2	Natur	Gummie-rung	Durchsteck-verbindung	Linsen-senkkopf	nicht ange-trieben	Großbri-tannien
Rm	Weißlich	Alu	geklebt	Torx	AWD	EAC
%	Bleigrau	Keramik	eingepresst	Vierkant	2WD	Japan
…	…	…	…	….	…	…

Tabelle 3-1: Auszug möglicher Ausprägungen

Durch den inhaltlichen, paarweisen Vergleich der Anforderungen einer Komponente aus verschiedenen Entwicklungsprojekten wird deren Vorkommen in den verschiedenen Projekten ermittelt und sie werden darauf aufbauend in die vorgestellten drei Gruppen kategorisiert. Dieser inhaltliche Vergleich erfordert aufgrund der großen Anzahl von Anforderungen in den Anforderungsdokumenten und der Vielzahl der Kombinationsmöglichkeiten zwingend eine Automatisierung. Dieses Erfordernis lässt sich durch ein Rechenbeispiel verdeutlichen: Der Komponentenentwicklung liegen für von ihr verantwortetes, mechanisches Bauteil 20 verschiedene Lastenhefte aus unterschiedlichen Entwicklungsprojekten vor. Jedes dieser Lastenhefte enthält im Durchschnitt 500 Anforderungen. Möchte sie jede Anforderung aus jedem dieser Lastenhefte paarweise miteinander vergleichen, um herauszufinden, ob es sich um die gleichen oder unterschiedlichen Anforderungen handelt, ergibt die Kombinatorik 50.005.000 mögliche Vergleiche (k=2 und n=20x500 Anforderungen=10.000 Anforderungen).

Die Berechnung der Kombinationsmöglichkeiten erfolgt anhand des Modells: Kombination mit Wiederholung ohne Beachtung der Reihenfolge anhand der Formel (2).

$$\binom{n+k-1}{k} = \binom{10.001}{2} = 50.005.000 \quad (2)$$

Dieses Beispiel verdeutlicht allein aufgrund der großen Anzahl an Kombinationsmöglichkeiten, dass für die Methode der Anforderungsvergleich automatisiert werden muss. Welche technischen Ansätze dafür gewählt wurden und nach welchen Regeln die Fließtexte automatisiert in einzelne Anforderungen zerlegt werden, wird in Abschnitt 3.2.1 erläutert.

Im Anschluss an den inhaltlichen Vergleich der Anforderungen aus den verschiedenen Projekten kann anhand der Erhebung des Vorkommens einer Anforderung eine Klassifizierung in die zuvor genannten drei Gruppen projektübergreifend, ausgeprägt und projektspezifisch erfolgen. Zur Identifizierung variantenverursachender Anforderungen mit dem Ziel einer Anforderungsharmonisierung und -optimierung werden anschließend nur diejenigen Anforderungen betrachtet, die als ausgeprägt oder als projektspezifisch klassifiziert wurden (Abbildung 3-2).

Ausgeprägte Anforderungen mit Schwankungen in den Ausprägungen werden hinsichtlich einer Harmonisierung, also einer Vereinheitlichung oder zumindest einer Angleichung der Ausprägungen, durch diese Methode transparent, sodass sie hinsichtlich einer Wiederverwendung hinterfragt werden können. Dabei soll die Anforderung in Bezug auf die Auswirkungen und Folgen bei Entfall einer oder mehrerer Ausprägungen durch die Harmonisierung betrachtet werden (Abwägung einer Über- oder Untererfüllung, vgl. Renner 2007). Projektspezifische Anforderungen als Sonderfälle einzelner Projekte sollten für einen möglichen Entfall zur Optimierung hinterfragt werden.

Als Ergebnis dieser Methode entsteht aus dem automatisierten Vergleich der Anforderungen und der automatisierten Gruppierung anhand des Vorkommens der Anforderungen eine Übersicht über alle Anforderungen, die für eine Anforderungsbibliothek zur Wiederverwendung in Folgeprojekten genutzt werden können und gibt Transparenz über die Spreizung der Anforderungen in den unterschiedlichen Projekten zur Identifikation potenzieller Variantentreiber. Die Analyse unterteilt die Anforderungen in projektübergreifende Anforderungen, die direkt in das Folgeprojekt als Basisanforderung übernommen werden können sowie in ausgeprägte Anforderungen mit markierter Ausprägung, deren quantitativer Wert für das Folgeprojekt noch bestimmt wer-

den muss. Dabei liefert die Analyse für die Ausprägung einen Bereich möglicher, bereits genutzter Werte. Darüber hinaus zeigt das Ergebnis der Methode auch projektspezifische Anforderungen auf, die nicht ohne vorherige Prüfung auf Notwendigkeit für das Folgeprojekt übernommen werden sollten. Somit stellt die Auflistung der gruppierten Anforderungen anhand der Methode die Basis dar, um anschließend die Anforderungen in eine Datenbank als Anforderungsbibliothek zur Wiederverwendung zu übertragen.

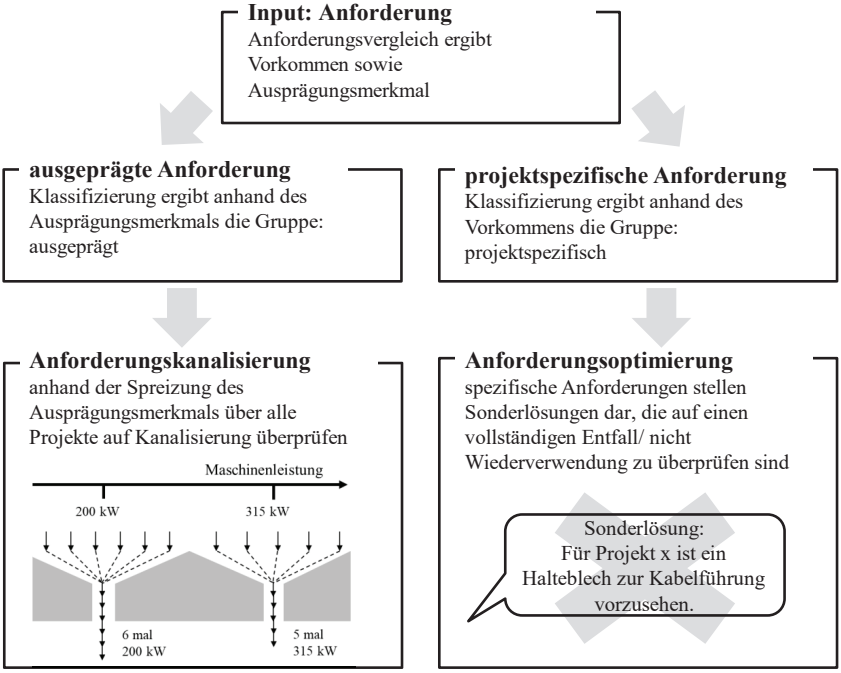


Abbildung 3-2: Anforderungsharmonisierung und -optimierung (eigene Darstellung in Anlehnung an Renner 2007)

3.2 Automatisierung des Anforderungsvergleichs

Der automatisierte Anforderungsvergleich lässt sich in zwei Phasen unterteilen: Die Textzerlegung (Preprocessing) und den semantischen Vergleich. In der ersten Phase Textzerlegung müssen aus dem Fließtext eines Anforderungsdokumentes die einzelnen Anforderungen identifiziert und in separate Textfragmente zerlegt werden. Gleichzeitig werden nicht verarbeitbare Abbildungen und Dokumentbereiche, die keine Anforderungen enthalten (z.B. Titelseite, Verzeichnisse etc.), entfernt. Die erforderlichen Schritte der Textzerlegung werden in Abschnitt 3.2.1 dargelegt. Erst nach der Zerlegung in einzelne Textfragmente können diese inhaltlich miteinander verglichen werden. Dies erfolgt in der zweiten Phase: dem semantischen Vergleich. Wie bereits in Kapitel 2.3.3 erläutert, unterscheidet man semantische und lexikalische Ähnlichkeit. Für den Vergleich der Anforderungen auf deren Vorkommen in den einzelnen Anforderungsdokumenten ist die semantische Ähnlichkeit im Gegensatz zu einem rein lexikalischen Vergleich zu betrachten (Kapitel 2.4). Die Automatisierung des semantischen Vergleichs wird in Kapitel 3.2.3 erklärt.

3.2.1 Textzerlegung – Preprocessing

Vor einem Vergleich der Anforderungen müssen die verschiedenen Dokumentationsformen auf zu vergleichende Textfragmente mit einheitlicher Textstruktur-Ebene übertragen werden. Die Aufbereitung der unterschiedlichen Formatierungen der Anforderungsdokumente sowie die Zerlegung der Texte in einzelne Anforderungen ist das Ziel des Preprocessing. Die Vorbereitung der Anforderungsdokumente besteht dabei aus den Schritten: Bereinigen, Zerlegen und Verknüpfen.

Der **erste Schritt Bereinigen** identifiziert und eliminiert alle Dokumentabschnitte, in denen keine Anforderungen enthalten sind, sowie nicht-textliche Elemente im Dokument. Das Dokument wird auf Formatierungen wie Titelseite und Verzeichnisse analysiert und diese entfernt. Fallabhängig kann es auch Kapitel geben, in denen allgemeine und rechtliche Bestimmungen im Umgang mit dem vorliegenden Dokument definiert werden. Da diese keine Anforderungen an die zu entwickelnde Komponente enthalten, werden sie ebenfalls entfernt. Dies kommt insbesondere bei der Dokumentationsform Lastenheft vor. Zusätzlich wird das Dokument auf Abbildungen analysiert und von diesen ebenfalls bereinigt, da nicht-textliche Anforderungen nicht verarbeitet werden können. Dies gilt auch für Anforderungen in Datenbanken: Enthält eine Anforderungs-ID in der Datenbank ein OLE-Objekt bzw. eine Abbildung, wird diese

übersprungen. Ein OLE-Objekt (Object Linking and Embedding) ist eine Verknüpfung oder Einbettung zu einer externen Datei, welche in einer anderen Applikation erstellt wurde.

Des Weiteren können manche Lastenhefte mehrsprachig und dadurch mehrspaltig formatiert sein (Abbildung 3-3): Hier enthalten beide Beispiele im .docx-Format in einer Spalte die deutschsprachige Anforderung und in einer weiteren die englischsprachige Anforderung. Die Lastenhefte, auf die diese Methode angewandt werden soll, sind daher zunächst auf diese Struktur zu prüfen und die auszulesende Spalte zu definieren. Ebenso ist dies für die Exporte aus einer Anforderungsdatenbank wie DOORS in .xlsx-Format durchzuführen, in denen in der Regel mehr als die in Abbildung 3-3 gezeigten zwei Spalten ausgeleitet werden.

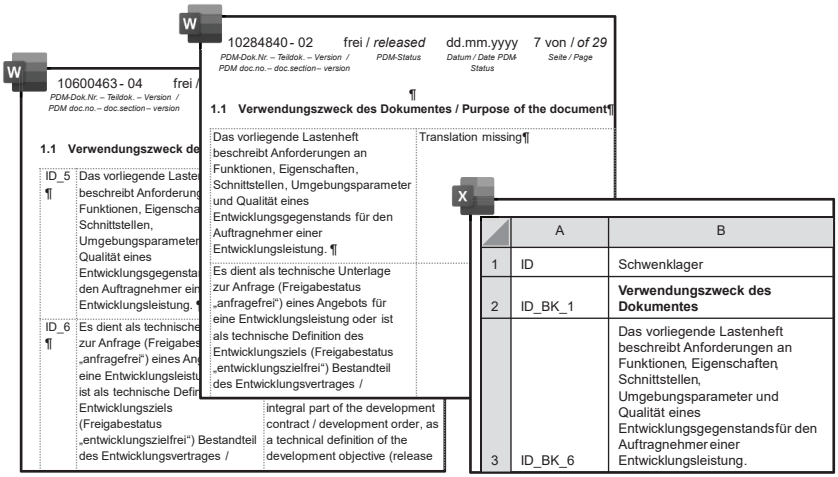


Abbildung 3-3: Unterschiedliche Formatierungen für Lastenhefte und Datenbankexporte

Im **zweiten Schritt Zerlegen** wird der Fließtext in einzelne Anforderungen, den Token, zerlegt. Bei dieser Tokenisierung wird der Text auf Punktation und Absatzmarken geprüft und Regeln zur Erkennung von Anfang und Ende eines Tokens aufgestellt. Wird das Satzzeichen Punkt detektiert, wird geprüft, ob es sich um einen Punkt bei einer Abkürzung, Aufzählung, Kapitelstruktur oder einem Zahlentrennzeichen handelt. Ist dies nicht der Fall, wird das Textfragment hier abgeschnitten. Tritt eine Absatzmarke ohne vorheriges Satzzeichen auf, wird ebenfalls das Textfragment abgeschnitten. Ebenfalls bei der Kombination aus aufeinanderfolgenden Satzzeichen und Absatzmarke. Aufgrund der Formatierung als Tabelle auch in .docx-Formaten (Abbildung 3-3) könnte man davon ausgehen, dass die Zellformatierung den Anfang und das Ende

festlegt. Allerdings wurden bei der Analyse vorhandener Anforderungsdokumente festgestellt, dass eine Anforderung entweder über mehrere Zellen verteilt wurde, insbesondere bei Aufzählungen, oder umgekehrt in einer Zelle ein ganzer Absatz mit mehreren Anforderungen enthalten war. Gerade die Aufzählungen bedürfen einer gesonderten Regelung: Enthält ein Textfragment einen Doppelpunkt am Ende des Fragmentes, so weist dies auf einen Zusammenhang mit dem darauffolgenden Textfragment hin, mit dem es zusammengelegt wird (s. Kataphorik Kapitel 2.3.3.4).

Ist ein Textfragment als Stichpunkt oder Aufzählung formatiert (s. Beispiele Tabelle 3-2), so muss abhängig von der Länge des Fragmentes entschieden werden, ob es mit dem vorhergehenden Fragment verknüpft werden muss.

Dokument A - Original	
ID_1	Das Entwicklungsobjekt muss so ausgelegt sein, dass seine volle Funktionsfähigkeit sichergestellt wird über:
ID_2	- mindestens XX Jahre oder
ID_3	- mindestens XX km

Dokument A – nach dem Preprocessing
Das Entwicklungsobjekt muss so ausgelegt sein, dass seine volle Funktionsfähigkeit sichergestellt wird über: - mindestens XX Jahre oder - mindestens XX km

Tabelle 3-2: Preprocessing - Beispiel zur Verknüpfung von Stichpunkten

Z.B. kann der Stichpunkt *„- minimale Umgebungstemperatur -40°C“* allein keine Anforderung darstellen. Er muss mit dem vorhergehenden Textfragment *„Der Entwicklungsgegenstand ist für folgenden Temperaturbereich auszulegen“* verknüpft werden. Hingegen kann der Stichpunkt *„• Kein Auftreten von Perforation/Durchrostung innerhalb von XX Jahren nach Inbetriebnahme“* selbst als Anforderung betrachtet werden und muss nicht zwingend verknüpft werden.

Da durch die Zerlegung in vielen Fällen der Kontext eines Satzes verloren geht, ist in bestimmten Fällen eine anschließende **Verknüpfung** einzelner Textfragmente in einem **dritten Schritt** erforderlich. Im Fall von Konjunktionen und Verweisen ist eine Verknüpfung erforderlich, da diese Textfragmente entweder auf ein vorangehendes oder auf ein nachfolgendes Textfragment verweisen. Dies wird in der Linguistik als Anapher oder Katapher bezeichnet (Kapitel 2.3.3.4). Tabelle 3-3 zeigt ein solches Beispiel.

Dokument A - Original	
ID_10	Der Lagerbock ist dem Steinschlag im Fahrzeug ausgesetzt.
ID_11	**Deshalb** ist der Achsträger an dieser Stelle 0,8 mm in der Wandstärke zu erhöhen gegenüber der Auslegung.

Dokument A – nach dem Preprocessing
Der Lagerbock ist dem Steinschlag im Fahrzeug ausgesetzt. **Deshalb** ist der Achsträger an dieser Stelle 0,8 mm in der Wandstärke zu erhöhen gegenüber der Auslegung.

Tabelle 3-3: Preprocessing - Beispiel zur Verknüpfung bei Anaphern

Da es für den deutschsprachigen Bereich nur wenige, auf ihren Anwendungsfall spezifische Algorithmen zur Auflösung solcher Verknüpfungen gibt (Carstensen et al. 2010, S. 407) und diese besonders zur Pronomenauflösung effizient sind, wurden an dieser Stelle regelbasiert eine Auflistung von in den Anforderungsdokumenten verwendeten Referenzausdrücken angelegt (Tabelle 3-4). Handelt es sich um einen Vorwärtsbezug, wurde das nachfolgende Textfragment mit dem aktuellen zusammengelegt. Im Fall eines Rückwärtsbezugs wurde das vorangegangene Textfragment mit dem aktuellen verknüpft.

Katapher / Vorwärtsbezug	Anapher / Rückwärtsbezug	
:	dies(-e,-es,-en,-er)	er, sie, es
,dass	hierfür	ihr (-e,-es,-en,-er)
im Folgenden	hierbei	sein (-e,-es,-en,-er)
unten (genannt/beschrieben/erläutert)	außerdem	dessen
folgende (-n, -s,-r)	alternativ	falls
nachstehend (-e,-en)	optional	betroffen (ist/sind)
nachfolgend (-e,-er,-es,-en)	Ausnahmen	ggf.
folgendermaßen	dafür	Anmerkung
lt. unteren (Werten/…)	darum	Bemerkung
untenstehend (-e, -en)	deshalb	Hinweis
…	dabei	wie oben (beschrieben/dargestellt/erläutert/genannt)
	dazu	welche
	dementsprechend	vorstehend
	zusätzlich	(…)
	darüber hinaus	…

Tabelle 3-4: Preprocessing - Beispiele für Referenzbegriffe/ Anaphern und Kataphern

Neben der Verknüpfung von Textfragmenten spielen die Kapitelüberschriften ebenfalls eine wichtige Rolle für den Kontext einer Anforderung (Boutkova 2014, S. 60). So kann die gleiche Anforderung in zwei unterschiedlichen Kapiteln vorkommen, aber einen unterschiedlichen Kontext haben. Z.B. kommt die Anforderung

„Die Schnittstelle ist als Durchsteckverschraubung mit M8 Gewinde auszuführen."

sowohl im Kapitel „Schnittstelle Komponente A zu Komponente B" als auch im Kapitel „Schnittstelle Komponente A zu Komponente C" vor. Daher wird zu jedem Textfragment die zugehörige, vorstehende Überschrift der jeweiligen untersten Ebene kopiert (Tabelle 3-5).

Dokument A - Original		Dokument A – nach dem Preprocessing
3.6.1 Schnittstelle zu Komponente B		3.6.1 Schnittstelle zu Komponente B Die Schnittstelle ist als Durchsteckverschraubung mit M8 Gewinde auszuführen.
ID_7	Die Schnittstelle ist als Durchsteckverschraubung mit M8 Gewinde auszuführen.	
3.6.5 Schnittstelle zu Komponente C		3.6.5 Schnittstelle zu Komponente C Die Schnittstelle ist als Durchsteckverschraubung mit M8 Gewinde auszuführen.
ID_8	Die Schnittstelle ist als Durchsteckverschraubung mit M8 Gewinde auszuführen.	

Tabelle 3-5: Preprocessing - Beispiel zur Kontextrelevanz der Überschrift

Nach Durchlaufen dieser drei Schritte des Preprocessing: Bereinigen, Zerlegen und Verknüpfen liegen die zerlegten Textfragmente, die Anforderungen, je Anforderungsdokument als Liste vor. Jedes Textfragment erhält eine ID, mit der sie dem ursprünglichen Dokument und dadurch einem Produkt bzw. Entwicklungsprojekt zugeordnet werden kann. Tabelle 3-6 zeigt schematisch das Prinzip der ID für jedes Textfragment je Dokument auf.

Dokument A	Dokument B	Dokument C
A1	B1	C1
A2	B2	C2
…	…	…
An	Bn	Cn

Tabelle 3-6: Textzerlegung - Index-Nr. der einzelnen Textfragmente/ Anforderungen

Basierend auf dieser Zerlegung können nun die einzelnen Textfragmente in der nächsten Phase paarweise, semantisch miteinander verglichen werden.

3.2.2 Anforderungsvergleichs und Anforderungsklassifizierung

Die automatisierte Anforderungsklassifizierung in die drei zuvor definierten Anforderungsgruppen projektübergreifend, ausgeprägt und projektspezifisch erfolgt anhand von zwei Merkmalen: Das erste Merkmal ist das Vorkommen einer Ausprägung in der Anforderung. Diese Ausprägung zeichnet sich durch Zahlenwerte mit anschließenden Einheiten sowie Farben, Materialien, Antriebsarten, Länder, Verbindungsarten, Verschraubungsklassen etc. (Tabelle 3-1) aus. Wird ein solches Ausprägungsmerkmal gefunden, wird dieses farblich markiert und die Anforderung als ausgeprägt klassifiziert. Diese Klassifizierung erfolgt rein regelbasiert anhand der Liste möglicher Ausprägungen, die fortlaufend in Optimierungsschleifen manuell erweitert werden kann.

Das zweite Merkmal zur Anforderungsklassifizierung besteht aus dem Vorkommen einer inhaltlich ähnlichen Anforderung in den zu vergleichenden Anforderungsdokumenten als Häufigkeitsverteilung einer Anforderung über alle zu vergleichenden Entwicklungsprojekte. Um dieses Vorkommen analysieren zu können, ist ein semantischer Vergleich aller Anforderungen in den zu vergleichenden Anforderungsdokumenten erforderlich. Für diesen Vergleich wird die Technologie des Semantic Folding angewandt. Das Vorgehen wird detailliert in Kapitel 3.2.3 beschrieben. Das Ergebnis dieses Anforderungsvergleichs der Dokumente wird in der Ergebnistabelle (Tabelle 3-7) mit folgender Struktur dargestellt:

In der obersten Zeile stehen spaltenweise die verglichenen Anforderungsdokumente mit ihrem Dateinamen (z.B. mit einer eindeutigen Dokument-Nr. mit Version im Dateinamen). Anhand der Dateinamen kann auch eine Ordnung der Dokumente nach Chronologie der Entwicklungsprojekte oder nach Zugehörigkeit zu einer Produktlinie sinnvoll sein.

Zusätzlich enthält die erste Spalte der Tabelle die Klassifizierung der Anforderung in eine der Gruppen projektübergreifend, ausgeprägt oder spezifisch. Die zweite Spalte zeigt auf, in wie vielen Dokumenten die Anforderung vorkommt. Hierzu werden je Zeile die als ähnlich identifizierten Anforderungen gezählt (s. Tabelle 3-7 Spalte „Anzahl“). Die Anzahl des Vorkommens einer Anforderung ist anschließend für die Eingruppierung der Anforderungen in projektübergreifend oder projektspezifisch notwendig. Hier wird als Regel in Abhängigkeit der Anzahl der verglichenen Dokumente festgelegt, dass bei einem Vorkommen in über 50% der verglichenen Dokumente die

Anforderung als projektübergreifend eingestuft wird. Bei Vorkommen von kleiner oder gleich 50% wird die Anforderung als spezifisch klassifiziert.

Gruppe	Anzahl	Dokument A	Dokument B	Dokument C
Projekt-übergreifend	3	Bei der Auslegung des Entwicklungsgegenstandes ist darauf zu achten, dass Montage und Demontage im Kundendienst keine Sonderwerkzeuge erfordern.	Der Entwicklungsgegenstand muss so gestaltet sein, dass er mit werkstattüblichen Normwerkzeugen zerstörungsfrei zu demontieren ist.	Bei der Auslegung des Entwicklungsgegenstandes ist darauf zu achten, dass Montage und Demontage im Kundendienst keine Sonderwerkzeuge erfordern.
spezifisch	1			Ein Halteblech zur Kabelführung ist vorzuhalten.
ausgeprägt	3	In RCAR-Struktur Crashversuchen bis zu 15 km/h ist eine Beschädigung des Achsträgers nicht zulässig.	Der Achsträger darf bei folgender Geschwindigkeit in Crashversuchen nicht beschädigt werden: • 20 km/h	RCAR-Struktur: Bei RCAR-Struktur Crashversuchen bis zu 16 km/h darf der Achsträger nicht beschädigt werden.

Tabelle 3-7: Ergebnistabelle des Anforderungsvergleichs

Danach folgen in der Ergebnistabelle die Spalten der Anforderungen aus den jeweiligen Dokumenten. In einer Zeile stehen dabei die jeweils als semantisch ähnlich identifizierten Anforderungen. Wurde in einem Dokument keine semantisch ähnliche Anforderung identifiziert, bleibt die Zelle leer. Anhand der Ergebnistabelle und der Klassifizierung der Textfragmente kann nun für zukünftige Entwicklungsprojekte eine Anforderungsbibliothek zur Wiederverwendung aufgebaut werden.

Im folgenden Kapitel wird detailliert auf das Vorgehen und die Methode des semantischen Vergleichs zur Ermittlung inhaltlich ähnlicher Anforderungen eingegangen.

3.2.3 Semantischer Vergleich

Das Ziel des semantischen Vergleichs ist die Ermittlung einer semantischen, inhaltlichen Ähnlichkeit aller in den betrachteten Dokumenten vorliegenden Anforderungen untereinander. Aus der Ermittlung dieser semantischen Ähnlichkeit wird abgeleitet, wie häufig eine Anforderung in unterschiedlichen Formulierungen in den Dokumenten

vorkommt. Dieser Abschnitt beschreibt die Erzeugung einer von Software verarbeitbaren Repräsentation der inhaltlichen Bedeutung der Textfragmente sowie den Ablauf des Vergleichs der einzelnen Textfragmente für einen paarweisen Vergleich. Abschließend werden die Generierung und Definition eines Grenzwertes, ab dem zwei inhaltlich ähnliche Anforderungen als gleich zu betrachten sind, aufgezeigt.

3.2.3.1 Semantische Repräsentation der Anforderungen

Die Textfragmente aus den Anforderungsdokumenten nach dem Preprocessing können auf unterschiedliche Weise miteinander verglichen werden: lexikalisch und semantisch. Wie bereits in Kapitel 2.4 erläutert, ist aufgrund des natürlichsprachigen Charakters von Anforderungen ein semantischer Vergleich erforderlich. Die semantische Ähnlichkeit zwischen zwei Textfragmenten wird automatisiert von Technologien aus dem Bereich NLP bestimmt. In Kapitel 2.3 wurden hierzu mehrere Ansätze erläutert. Jeder dieser Ansätze bildet die Bedeutung von Texten auf unterschiedliche Arten für Software verarbeitbar ab. Für korpusbasierte Ansätze erfolgt die Messung der semantischen Ähnlichkeit durch den Cosinus zwischen zwei Vektoren in einem hochdimensionalen Raum. In der hier beschriebenen Methode wurde der Ansatz Semantic Folding des Kooperationspartners cortical.io für die Anwendung der Methode umgesetzt und getestet. Hierbei wird der betrachtete Sprachraum in einer s.g. Semantic Map repräsentiert, eine zweidimensionale Karte, aus der für jedes Wort ein Semantic Fingerprint als zweidimensionaler Binärcode gebildet wird.

Zum Aufbau der Semantic Map, dem Modell des betrachteten Sprachraums im Semantic Folding Ansatz, muss zunächst das Sprachmodell für den entsprechenden Anwendungsfall trainiert werden. Für dieses Antrainieren sind repräsentative Dokumente, die den Sprachumfang des jeweiligen Anwendungsfalls abbilden, auszuwählen. In der hier betrachteten Methode soll das Sprachmodell technische Anforderungen in Lastenheften abbilden können. Deshalb werden vorhandene Lastenhefte der betrachteten Produkte und Bauteile ausgewählt. Dabei wird beachtet, dass die Lastenhefte aus verschiedenen Produktlinien und Erscheinungsjahren stammen. Des Weiteren werden Dokumente, auf die in den Lastenheften verwiesen wird wie z.B. DIN-Normen, VDI- und VDA-Richtlinien, genutzt. Zusätzlich ist auch das Abbilden der unternehmensspezifischen Sprache wichtig, in der spezifische Abkürzungen und Begriffe angewendet werden. Deshalb werden Abkürzungsverzeichnisse als auch Dokumente zu unternehmensspezifischen Standards und Vorschriften zum Antrainieren verwendet. Um zusätzlich die technische Sprache der Entwickler abzubilden, wird darüber hinaus

Fachliteratur zu den Themen Maschinenelemente, Konstruktionslehre, Bauteilauslegung sowie Anwendungsfallspezifische Literatur ausgewählt.

Die Dokumente zum Antrainieren des Sprachmodells werden im verwendeten Ansatz Semantic Folding automatisiert von nicht-textlichen Elementen bereinigt und anschließend in kleine Textfragmente zerlegt. Diese Textfragmente sind 2-3 Sätze lang und werden als **Snippets** bezeichnet. Jeder Snippet wird auf der Semantic Map einem Bit zugewiesen. Anhand des Vorkommens ähnlicher Begriffe werden die Snippets in Clustern auf der Semantic Map angeordnet, sodass Snippets mit einem hohen Anteil gleicher Wörter nahe beieinander liegen. Am Ende dieses Prozesses zum Antrainieren des Sprachmodells entsteht ein spezifisch auf den Anwendungsfall antrainiertes Sprachmodell, dass die unternehmensspezifische Sprache in den Anforderungsdokumenten als Semantic Map abbildet. Anhand dieses Sprachmodells können nun durch die Software für jedes Wort ein Semantic Fingerprint gebildet werden, der die Bedeutung eines Wortes, dessen Konzept, in einem maschinenlesbaren Binärcode abbildet.

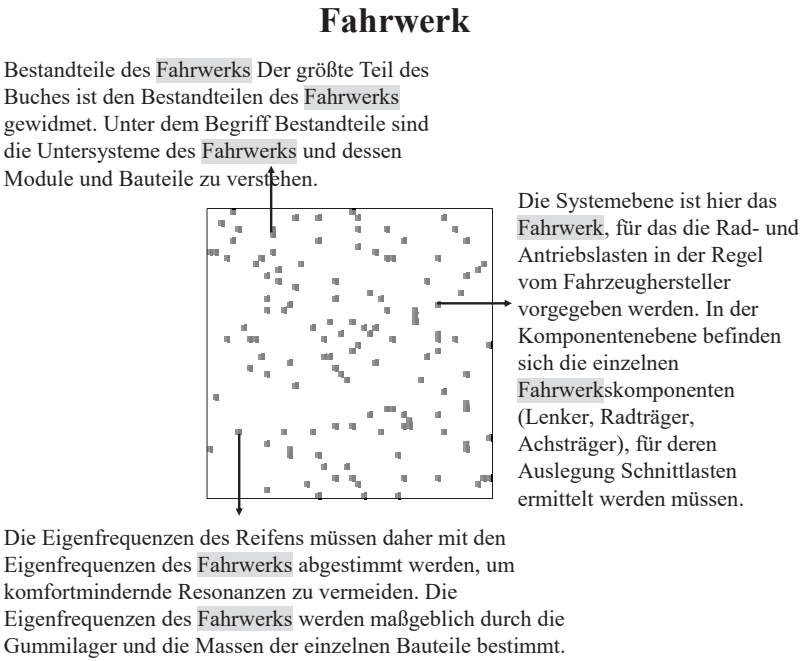


Abbildung 3-4: Beispielhafte Darstellung eines Semantic Fingerprints des Fahrwerks

Einen solchen Semantic Fingerprint zeigt Abbildung 3-4 für den Begriff Fahrwerk. In der Bildmitte ist der zweidimensionale Binärcode zu sehen, bei dem nur vereinzelte Bits aktiv sind. Hinter jedem Bit liegt ein Snippet aus der Semantic Map. Bei der Bildung des Semantic Fingerprints für den Begriff Fahrwerk wird jeder Snippet nach dem Vorkommen des Wortes Fahrwerks durchsucht. Kommt der Begriff vor wie in den Textelementen in Abbildung 3-4, wird der Bit aktiv.

Die Semantic Fingerprints der einzelnen Wörter bilden übereinandergelegt den Semantic Fingerprint eines Satzes. Dabei folgt die Methode den in Kapitel 2.3.3.3 in Abbildung 2-19 gezeigten Schritten. Vor der Abbildung eines Satzes wird der Satz durch die Software von s.g. Stopp-Wörtern bereinigt. Diese sind Begriffe, die keinen Informationsgehalt haben, sondern eine grammatikalische und syntaktische Funktion aufweisen. Listen für solche Stopp-Wörter sind zahlreich im Internet veröffentlicht. Im spezifischen Anwendungsfall des betrachteten Sprachgebrauchs (Kapitel 4.1) wurden insgesamt 88 Stopp-Wörter identifiziert, die in Tabelle 3-8 aufgelistet werden. Für diese Wörter wird kein Semantic Fingerprint gebildet und sie spielen daher bei der Abbildung der Sätze in einer Repräsentation keine Rolle.

alle	da	dieser	fuer	jeder	sein	war
als	dabei	dieses	für	kann	seine	waren
am	daher	durch	gab	keine	seiner	wenn
an	darf	ein	hat	konnte	sich	werden
ans	das	eine	hatte	können	sind	wie
auch	dass	einem	hatten	könnten	so	wiederum
auf	dem	einen	hierbei	mit	sowie	wird
aus	den	einer	im	muss	um	wodurch
bei	der	eines	in	müssen	und	zu
beim	des	es	indem	nach	vom	zum
beziehungsweise	die	er	ins	nicht	von	zur
bis	diese	etc	inzwischen	nur	vor	
bzw	diesem	folgende	ist	oder		

Tabelle 3-8: Liste der entfernten Stopp-Wörter

Um nach der Abbildung der Sätze in einem Semantic Fingerprint diese inhaltlich zu vergleichen, wird der Überlappungsgrad der Bits zweier Semantic Fingerprints in Prozent [%] gemessen (Abbildung 3-5). Je höher der Prozentsatz, umso ähnlicher sind sich die verglichenen Textfragmente.

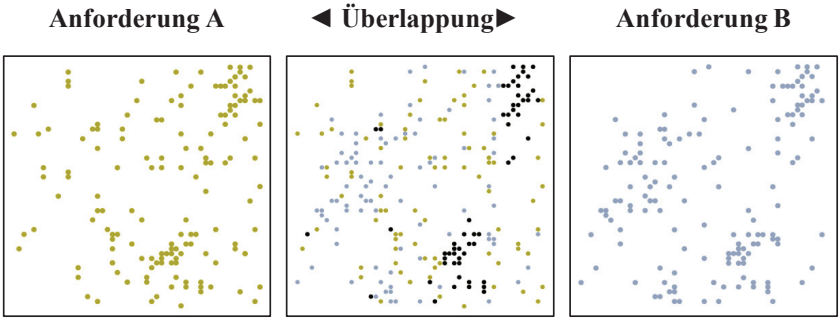


Abbildung 3-5: Messung der semantischen Ähnlichkeit (Eigene Darstellung in Anlehnung an Webber 2016, S. 40)

3.2.3.2 Ermittlung des Grenzwertes für semantische Ähnlichkeit

Nach dem Antrainieren der Sprachmodelle ist es erforderlich, abhängig vom jeweiligen Anwendungsfall, einen Grenzwert für die semantische Ähnlichkeit zu bestimmen. Anhand dieses ermittelten Grenzwertes entscheidet die Software, ob zwei Anforderungen als semantisch gleich oder verschieden betrachtet werden sollen. Um festzulegen, ab welchem Grenzwert zwei Textfragmente als inhaltlich ähnlich zu betrachten sind, müssen mehrere Beispiele, sowohl Negativ- als auch Positivbeispiele, manuell ausgearbeitet werden. Dabei werden Testdokumente für einen Vergleich ausgewählt, aus denen mehrere Beispiele definiert werden.

Bei der Auswahl der Beispiele zur Bestimmung des Grenzwertes wird darauf geachtet, Anforderungspaare im Grenzbereich zu finden. Dies sind z.B. inhaltlich ähnliche Anforderungen, die fälschlicherweise als verschieden eingestuft werden können (1. Fall Tabelle 3-9) oder unterschiedliche Anforderungen, deren Unterschied nur sehr gering ist und die daher fälschlicherweise als inhaltlich gleich eingestuft werden können (2. Fall Tabelle 3-10). Es sollten insbesondere schwierige Beispiele ausgewählt werden, um im Grenzfall die Feinabstimmung des Grenzwertes der semantischen Ähnlichkeit genauer bestimmen zu können.

Beispiel 1	Im gesamten Montageablauf muss Verletzungsgefahr (z.B. durch scharfe Kanten) ausgeschlossen werden.	Das gesamte Bauteil ist frei von scharfen Kanten und Graten zu gestalten. Es gelten die Anforderungen für Werkstückkanten nach [DIN ISO 13175].
Beispiel 2	Der Entwicklungsgegenstand ist als linke und rechte Variante spiegelbildlich auszuführen.	Es sind ein Links und ein Rechtsteil zu entwickeln. Das Rechts-Teil ist spiegelbildlich zum Links-Teil zu entwickeln.
Beispiel 3	Die Gewährleistungs- und Kulanzfälle pro 100 Einheiten innerhalb von 24 Monaten nach Produktion dürfen für den spezifizierten Entwicklungsgegenstand weltweit 0,00075 Fälle nicht überschreiten.	Folgende Grenzwerte werden festgelegt: Grenzwert 24 MnP weltweit: Beanstandungsquote Zuverlässigkeit /100 ppm

Tabelle 3-9: Grenzwerteinstellung 1. Fall: inhaltlich ähnliche Anforderungen

Beispiel 1	Die Anbindung des Radlagers ist mittels vier jeweils einschnittigen Verschraubungen mit Gewinde im Radlager auszuführen.	Die Anbindung des Bremsenschutzblechs ist mittels vier jeweils einschnittigen Verschraubungen mit Gewinde im Schwenklager auszuführen.
Beispiel 2	Der Entwicklungsgegenstand ist so auszuführen, dass auch bei Überlast die Verbindung zwischen Lager und radführenden Komponenten stets erhalten bleibt.	Der Entwicklungsgegenstand ist derart zu gestalten, dass bei Überlastung aus Missbrauch eine erkennbare, rissfreie plastische Verformung hervorgerufen wird.
Beispiel 3	Hierbei muss u.a. die Erreichung einer min. Dämpfkraft von 4 kN bei 4 m/s sichergestellt sein.	Ferner darf bis 7 m/s eine max. Dämpfkraft von max. 12 kN nicht überschritten werden.

Tabelle 3-10: Grenzwerteinstellung 2. Fall: inhaltlich verschiedene Anforderungen

Zur Kontrolle der Testergebnisse sind diese Beispiele für die Testdokumente in die in Tabelle 3-7 vorgestellte Form der Ergebnistabelle zu bringen, sodass bei einem Testlauf zur Bestimmung des Grenzwertes ersichtlich ist, welche Anforderung in keinem anderen Dokument gefunden werden sollte. In Tabelle 3-11 ist in der unteren Zeile in Dokument B ein falscher Treffer dargestellt. Die Anforderung unterscheidet sich gemäß den zuvor definierten Beispielen von den Anforderungen aus Dokument A und C. Der Grenzwert ist hier zu niedrig.

Gruppe	Anzahl	Dokument A	Dokument B	Dokument C
ausgeprägt	3	Hierbei muss u.a. die Erreichung einer min. Dämpfkraft von 4 kN bei 4 m/s sichergestellt sein.	Ferner darf bis 7 m/s eine max. Dämpfkraft von max. 12 kN nicht überschritten werden.	Hierbei muss u.a. die Erreichung einer min. Dämpfkraft von 11 kN bei 7m/s sichergestellt sein.

Tabelle 3-11: Grenzwerteinstellung - Beispiele für einen zu niedrigen Grenzwert

Ist der Grenzwert zu hoch eingestellt, werden inhaltlich ähnliche Anforderungen nicht gefunden. So wird in Tabelle 3-12 eine Anforderung aus Dokument B fälschlicherweise als spezifisch eingestuft, obwohl die anderen beiden Dokumente die gleiche Anforderung in anderer Formulierung enthalten.

Gruppe	Anzahl	Dokument A	Dokument B	Dokument C
spezifisch	1		Der Entwicklungsgegenstand ist als linke und rechte Variante spiegelbildlich auszuführen.	
projektübergreifend	2	Es sind ein Links- und ein Rechtsteil zu entwickeln. Das Rechts-Teil ist spiegelbildlich zum Links-Teil zu entwickeln.		Es sind ein Links- und ein Rechtsteil zu entwickeln. Das Rechts-Teil ist spiegelbildlich zum Links-Teil zu entwickeln.

Tabelle 3-12: Grenzwerteinstellung - Beispiele für einen zu hohen Grenzwert

Sowohl die Bildung der Semantic Map als Antrainieren des Sprachmodells, als auch die Bestimmung des Grenzwertes, müssen einmalig vor der Anwendung für jeden neuen Sprachumfang durchgeführt werden. Verändert sich der Sprachgebrauch in den Anforderungsdokumenten durch neue Technologien, Innovationen oder neue Konzepte, oder verschlechtert sich die Prognosegüte des Vergleichs, müssen diese Schritte ebenfalls erneut an entsprechenden Beispielen durchgeführt werden. Das Sprachmodell der Semantic Map lernt in der vorgestellten Methode nicht kontinuierlich mit jeder Anwendung der Methode mit, sondern muss in den zuvor beschriebenen Fällen separat erweitert und für neue Sprachumfänge antrainiert werden.

Wurde das Sprachmodell erfolgreich antrainiert, ein Grenzwert für die semantische Ähnlichkeit im jeweiligen Anwendungsfall bestimmt und die Textzerlegung/Preprocessing an die jeweiligen Formatierungen angepasst, kann ein automatisierter Anforderungsvergleich durchgeführt werden.

3.2.3.3 Vergleichsprozedur

Wie in Tabelle 3-6 dargestellt, liegen die Anforderungen nach der Zerlegung der Dokumente als einzelne Anforderungen in Textfragmenten vor. Im Folgenden ist von Textfragmenten die Rede, da es sich bei den Anforderungen teils um einzelne Sätze, aber auch Aufzählungen oder 2-3 Sätze gemäß des Preprocessing handeln kann. Dies hat seine Ursache in den in Kapitel 3.2.1 beschriebenen Formatierung einer Anforderung bestehend aus Aufzählungen oder Stichpunkten sowie die häufige Verwendung von Anaphern und Kataphern in natürlichsprachigen Texten. Um den semantischen Ähnlichkeitsgrad aller Anforderungen untereinander zu bestimmen, wird ein paarweiser Vergleich aller Anforderungen durchgeführt. Hierbei werden auch die Anforderungen innerhalb eines Dokumentes miteinander verglichen, um Duplikate zu identifizieren. Der Vergleich erfolgt dabei anhand der Repräsentationsform Semantic Fingerprint, der durch das Sprachmodell für jedes Textfragment gebildet wird. Der Vergleich wird am Beispiel der wie zuvor in der Tabelle 3-6 beschriebenen, indexierten Anforderungen wie folgt durchgeführt:

Zunächst wird beginnend mit der Anforderung A_1 aus Dokument A nach Duplikaten innerhalb des Dokuments A gesucht (Abbildung 3-6). Die Anforderung A_1 wird nacheinander paarweise mit jeder anderen Anforderung des Dokuments A von A_2 bis A_n verglichen. Wird dabei ein Duplikat erkannt, so werden die beiden Anforderungen mit einem Trennzeichen „---“ in ein Textfragment zusammengelegt. In Abbildung 3-6 wird z.B. zwischen den Anforderungen A_2 und A_3 eine hohe semantische Ähnlichkeit erkannt, sodass die Anforderungen für den nachfolgenden dokumentübergreifenden

Vergleich in ein Textfragment zusammengelegt werden. Die Identifikation von Duplikaten innerhalb eines Dokumentes wird als ersten Schritt des Vergleichs mit jedem zu vergleichenden Anforderungsdokument durchgeführt.

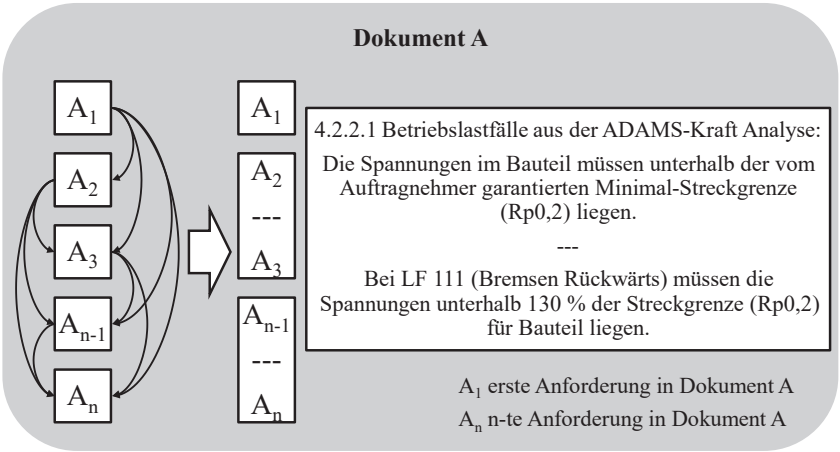
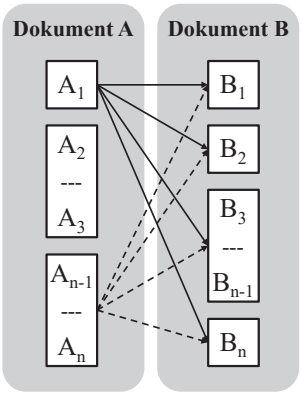


Abbildung 3-6: Identifikation von Duplikaten innerhalb eines Lastenheftes

Nach der Überprüfung auf Duplikate innerhalb des Dokuments erfolgt der Vergleich mit den Anforderungen der anderen Dokumente. Die erste Anforderung A_1 wird mit allen Anforderungen aus Dokument B verglichen. Ergibt ein Vergleich eine semantische Ähnlichkeit oberhalb des definierten Grenzwertes, wird die Anforderung aus Dokument B in die gleiche Zeile der Ergebnistabelle geschrieben (Abbildung 3-7). Werden mehrere Anforderungen in Dokument B mit einer semantischen Ähnlichkeit gefunden, wird die Anforderung mit der höchsten Ähnlichkeit in die Ergebnistabelle übernommen. Hier wären die Textfragmente A_1 und B_1 semantisch ähnlich. Ist Anforderung A_1 mit allen Anforderungen aus Dokument B verglichen worden, wird der Vergleich A_1 mit den Anforderungen aus Dokument C fortgesetzt usw. Wird keine ähnliche Anforderung gefunden, bleibt die Zelle der Ergebnistabelle leer. Ist der Vergleich von A_1 mit allen vorliegenden Dokumenten erfolgt, beginnt man mit der nächsten Zeile und Anforderung A_2. Dies erfolgt so lange, bis alle Anforderungen bis A_n aus Dokumente A verglichen worden. Für alle danach folgenden Vergleiche bleiben die Zellen des Dokumentes A leer (ausgegrauter Bereich in Abbildung 3-7) und die bereits in die Ergebnistabelle übertragenen Anforderungen werden nicht mehr berücksichtigt. Danach werden in Dokument B nur noch die verbleibenden x Anforderungen, die noch nicht zugeordnet wurden, mit den Anforderungen der anderen Dokumente verglichen.

Sind auch hier alle Anforderungen verglichen wurden, bleiben die folgenden Zellen des Dokumentes B leer.



$A_1 \dots A_n$ Anforderungen in Dokument A
$B_1 \dots B_n$ Anforderungen in Dokument B
$C_1 \dots C_n$ Anforderungen in Dokument C

Anzahl	Dokument A	Dokument B	Dokument C
2	A_1	B_1	
2	A_2 --- A_3		C_1
…	…	…	…
2	A_{n-2}	B_2	C_2
2	A_{n-1} --- A_n	B_n	C_4 --- C_{n-3}
…		…	…
1		B_3 --- B_{n-1}	
2		B_{n-2}	C_{n-2}
…			…
1			C_n

Abbildung 3-7: Paarweiser Vergleich - Treppenstruktur in Ergebnistabelle

Durch dieses Vorgehen ergibt sich eine Art Treppenstruktur der leeren Zellen (ausgegrauter Bereich) und man spart dadurch einige Kombinationen durch die Regel „Ziehen ohne Zurücklegen“ der Kombinatorik. Hier ein Beispiel anhand von insgesamt 10.000 (n) paarweise (k=2) zu vergleichenden Anforderungen und die Anzahl der Kombinationen:

Ziehen mit Zurücklegen

$$\binom{n+k-1}{k} = \binom{10.001}{2} = 50.005.000 \tag{3}$$

Ziehen ohne Zurücklegen

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} = \binom{10.000!}{2!\,(9.998)!} = 49.995.000 \tag{4}$$

Dieses Beispiel zeigt, dass durch die vorgegebene Treppenstruktur bereits bei einer Gesamtzahl von 10.000 Anforderungen 1.000 Kombinationen erspart bleiben. Bei entsprechend großem n werden die eingesparten Kombinationen und somit die eingesparte Rechendauer größer.

Nun kann basierend auf der Ergebnistabelle des Vergleichs Abbildung 3-7 die Auswertung und Klassifizierung der Anforderungen erfolgen. Diese Auswertung wird im folgenden Kapitel erläutert.

3.3 Aufbau einer Anforderungsbibliothek zur Wiederverwendung

In der Literatur werden für den Aufbau einer Anforderungsbibliothek Anforderungen in zwei Gruppen eingeteilt: gemeinsame und variable Anforderungen (Kapitel 2.1.3 Anforderungsbibliothek – gemeinsame und variable Anforderungen). Gemeinsame Anforderungen sind für alle Produktlinien relevant und werden im Folgenden als projektübergreifende Anforderungen bezeichnet. Diese Anforderungen können durch den Vergleich in einer Mehrheit der Lastenheften der Komponente gefunden werden. Hierbei handelt es sich um generische Anforderungen, die als Basisanforderungen für eine Anforderungsbibliothek dienen können und keine Varianten verursachen. Die variablen Anforderungen werden hingegen für das jeweilige Projekt explizit ausgewählt. Dabei können zwei Formen der Variabilität unterschieden werden: Einerseits können Anforderungen hinsichtlich der quantitativen Ausprägung ihres Merkmals variieren, z.B. die maximale Einpresskraft in Abbildung 3-1. Sie weisen je nach Variante einen anderen Wert als Ausprägung auf. Andererseits können Anforderungen hinsichtlich ihres Vorkommens in den Entwicklungsprojekten variieren. Im Folgenden als projektspezifisch bezeichnete Anforderungen weisen zwar keine variierende Ausprägung auf, kommen aber nur in einzelnen Projekten vor und sind daher in ihrem Vorkommen variabel.

Die Herausforderung im Aufbau einer Anforderungsbibliothek zur Wiederverwendung von Anforderungen in Folgeprojekten liegt in der Strukturierung einer historisch gewachsenen, unstrukturierten und in unterschiedlichen Dokumentationsformen vorliegende Anforderungsmenge. Diese Anforderungsmenge muss für den Aufbau der Bibliothek zunächst in gemeinsame Basisanforderungen und variable Anforderungen strukturiert werden. Durch die vorgestellte Methode mit einem inhaltlichen Anforderungsvergleich kann dies vollständig automatisiert erfolgen. Der Vergleich liefert eine Auswertung einer verglichenen, unstrukturierten Anforderungsmenge aus verschiedenen Dokumenten in die drei Gruppen projektübergreifende Basisanforderungen, ausgeprägte Anforderungen und projektspezifische Anforderungen (Abbildung 3-8).

Basierenden auf dieser automatisiert klassifizierten Anforderungsmenge lässt sich nun manuell ein projektspezifisches Lastenheft für ein Folgeprojekt konfigurieren. Die projektübergreifenden Basisanforderungen werden direkt in das zu konfigurierende Lastenheft übernommen. Bei den ausgeprägten Anforderungen ist zunächst für das Folgeprojekt ein Wert der Ausprägung manuell auszuwählen. Dabei liefert die Auswertung der Methode eine Übersicht der bereits verwendeten Standardwerte, die vorzugsweise im Folgeprojekt verwendet werden sollten. Zum Abschluss muss manuell

geprüft werden, ob für das Folgeprojekt projektspezifische Sonderfälle zutreffen, die bei einem ähnlichen Projekt bereits in projektspezifischen Anforderungen dokumentiert wurden. Ist dies der Fall, können diese spezifischen Anforderungen ebenfalls übernommen werden.

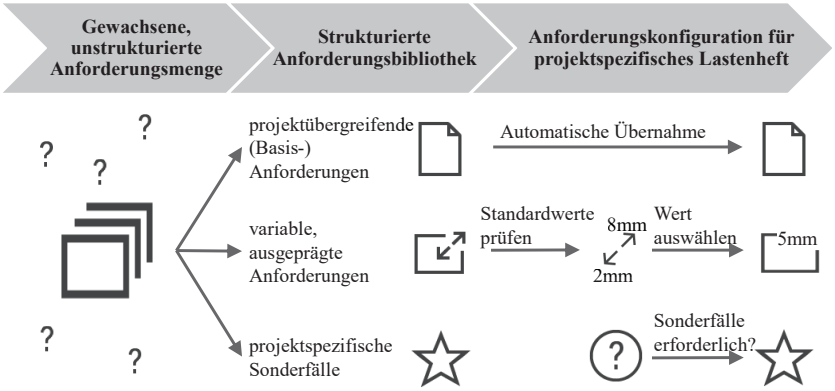


Abbildung 3-8: Anforderungsstrukturierung zur projektspezifischen Konfiguration

Zur Übernahme der Anforderungen in die Anforderungsbibliothek sind insbesondere die ausgeprägten und spezifischen Anforderungen auf ihre potenziell variantenverursachende Wirkung zu untersuchen und zu hinterfragen. Auch die bereits in eine Bibliothek aufgenommenen Anforderungen sind regelmäßig auf ihre Notwendigkeit zu überprüfen, um ein unkontrolliertes Anwachsen der Bibliothek zu verhindern. Das Ziel der Überprüfung ist es, nicht mehr benötigte Anforderungen aus der Bibliothek entfallen zu lassen bzw. für ein vergangenes Projekt zu archivieren und gleichzeitig Anforderungen aufgrund neuer Versionen von Normen oder Standards zu aktualisieren. Gerade hinsichtlich der Entwicklung von Baukästen oder Plattformen dient der automatisierte Vergleich zur Transparenz über die Variabilität der vorhandenen Anforderungen als auch deren Reduzierung. Zur Hinterfragung von Anforderungen und deren Optimierung hinsichtlich Standardisierung und Varianz bieten sich die von Renner (2007) entwickelten Ansätze der Anforderungsharmonisierung und -optimierung an. Basierend auf dem automatisierten Anforderungsvergleich aus Kapitel 3.2 sollen die als ausgeprägt und projektspezifisch eingestuften Anforderungen hinterfragt werden. Basierend auf der Hypothese, dass Unterschiede in den Anforderungen zu verschiedenen Merkmalen in der Realisierung der Komponente führen, werden Anforderungen zur Beherrschung der Variantenvielfalt hinsichtlich eines Entfalls oder einer Harmonisierung bewertet.

Projektspezifische Anforderungen, die nur in einzelnen Entwicklungsprojekten vorkommen, werden auf ihre Notwendigkeit hinterfragt und für eine mögliche Wiederverwendung oder Entfall für die Anforderungsbibliothek oder als nachträgliche Optimierungsmaßnahme eines schon in Serie befindlichen Bauteils bewertet. Ausgeprägte Anforderungen, deren Ausprägungen sich zwischen den Projekten unterscheiden, werden hinsichtlich einer möglichen Harmonisierung, bewertet. Dabei können einzelne Ausprägungen entfallen und durch einheitliche Werte ersetzt werden. Dies führt zu einer Über- oder Untererfüllung durch die Anforderung in einigen Projekten. Diese Über- oder Untererfüllung ist auf ihre Auswirkungen hin zu untersuchen.

3.4 Zusammenfassung

In diesem Kapitel wurde die Methode für einen automatisierten Anforderungsvergleich und -klassifizierung für den Aufbau einer Anforderungsbibliothek zur Wiederverwendung vorgestellt. Zunächst wurde die zugrunde liegende Idee, dass Unterschiede in Anforderungen Varianten verursachen, erläutert und die drei Gruppen projektübergreifende, ausgeprägte und spezifische Anforderungen vorgestellt. Anschließend wurden mögliche Strukturen und Formatierungen von Anforderungen in Anforderungsdokumenten aufgezeigt sowie die Herausforderungen, einzelne Anforderungen aus diesen verschiedenen Dokumentationsformen zu identifizieren und voneinander abzugrenzen. Des Weiteren wurde das Vorgehen für einen automatisierten Anforderungsvergleich vorgestellt, durch das die semantische Bedeutung von Anforderungen in s.g. Semantic Fingerprints vollautomatisiert verglichen und einander zugeordnet werden können. Für den semantischen Vergleich wird das Verfahren Semantic Folding der Firma cortical.io angewendet. Basierend auf dieser inhaltlichen Zuordnung kann das Vorkommen von Anforderungen in verschiedenen Entwicklungsprojekten analysiert und darauf aufbauend eine Klassifikation der Anforderungen in die zuvor genannten Gruppen erfolgen.

Zur Anwendung dieser Methode auf einen neuen Anwendungsfall müssen einmalig die Formatierung und der Aufbau der Anforderungsdokumente analysiert werden. Hierbei ist zu definieren, welche Abschnitte des Dokumentes Anforderungen enthalten und bei mehrsprachigen Dokumenten, welche Bereiche des Dokumentes die zu vergleichende Sprache beinhalten. Des Weiteren ist für jeden neuen Anwendungsfall einmalig ein Antrainieren des Sprachmodells erforderlich. Ein neuer Anwendungsfall stellt sowohl ein anderes Unternehmen, ein anderer Industriezweig, aber auch innerhalb eines Unternehmens eine neue Komponente oder Teilsystem des Produktes dar, welches bislang im Sprachmodell noch nicht abgebildet ist und ein anderes, neues Sprachgebiet darstellen. Für diese neuen Anwendungsfälle sind erneut geeignete Trainingsdokumente als repräsentative Dokumente des neuen, zusätzlichen Sprachgebiets auszuwählen. Gleichzeitig sind für neue Anwendungsfälle ebenfalls die Schritte zur Bestimmung und Kalibrierung des Grenzwertes (Kapitel 3.2.3.2) durchzuführen. Das Sprachmodell ist an dieser Stelle nicht selbstlernend und kann sich nicht eigenständig bei der Anwendung an unbekannten Dokumenten erweitern.

Wurde das Sprachmodell für ein Sprachgebiet einmalig antrainiert und der Grenzwert eingestellt, können die folgenden Schritte durch die Methode auf alle unbekannten

Dokumente desselben Sprachgebietes vollautomatisiert durchgeführt werden: Anforderungen abgrenzen, vergleichen und klassifizieren. Die Auswahl der zu vergleichenden Dokumente als auch die anschließende Übertragung geeigneter Anforderungen in eine Anforderungsbibliothek erfolgt manuell. Die Auswahl geeigneter Anforderungen für die Anforderungsbibliothek erfordert auch bei vorliegender Auswertung das Expertenwissen der Entwickler, da z.B. einige als projektübergreifend klassifizierte Anforderungen veraltet sind und somit archiviert werden müssten. Gleichzeitig bedarf auch die Auswahl der geeigneten Ausprägung eine Anforderung für die Konfiguration eines Lastenheftes das Expertenwissen der Entwickler und wird daher im Rahmen dieser Methode nicht automatisiert.

Die Grenzen der vorgestellten Methode sind Anforderungen, die nicht als Text vorliegen. Gerade bei mechanischen Komponenten werden häufig Anforderungen auch auf technischen Zeichnungen oder in Diagrammen dargestellt. Da die vorgestellte Methode allerdings auf Technologien des NLP beruht, die die Analyse natürlicher Sprache ermöglicht, können nur textlich festgehaltene Anforderungen verglichen werden. Des Weiteren ist der Fokus der vorgestellten Methode die Anwendung auf deutschsprachige Anforderungsdokumente für mechanische Komponenten. Eine Eignung der Methode auf englischsprachige Anforderungen oder auf Anforderungen aus dem Software-Engineering werden in dieser Arbeit nicht untersucht.

4. Anwendung und Validierung der Methode

Die entwickelte Methode zum automatisierten Anforderungsvergleich, um variantenverursachende Anforderungen zu identifizieren und zu optimieren, als auch um Anforderungen für eine Wiederverwendung zu klassifizieren, soll im Folgenden anhand der technischen Anforderungen in Komponentenlastenheften eines Fahrwerks der BMW AG angewendet und validiert werden. Dazu wird ein Sprachmodell für den Anwendungsfall mechanischer Fahrwerkskomponenten trainiert und anschließend der automatisierte Anforderungsvergleich in 42 Lastenheft-Vergleichen überprüft.

Zunächst werden in Kapitel 4.1 die Rahmenbedingungen des Anwendungsfalls erläutert. Hierbei wird der Aufbau der vorliegenden Anforderungsdokumente und der Fokus der betrachteten Komponenten beschrieben. Des Weiteren wird zur Einleitung das Vorgehen zum Aufbau des Sprachmodells und die Einstellung der Software im spezifischen Anwendungsfall erläutert. Anschließend wird in Kapitel 4.1.2 detailliert die Auswahl der Lastenhefte zur Validierung der Methode beschrieben.

In der Anwendung sollen folgende Fragestellungen (Kapitel 2.4) geprüft und beantwortet werden:

(1.1) Können Anforderungen basierend auf einem automatisierten Vergleich in die Gruppen projektübergreifend, ausgeprägt oder spezifisch klassifiziert werden?

(1.2) Ist durch die Klassifizierung eine produktspezifische Konfiguration eines Lastenheftes aus einer Anforderungsbibliothek einer Komponente möglich?

(2.1) Sind spezifische oder ausgeprägte Anforderungen variantenverursachend?

(2.2) Können durch einen automatisierten Anforderungsvergleich mit anschließender Klassifizierung variantenverursachende Anforderungen identifiziert werden?

(3.1) Wie können durch definierte Zerlegungsregeln im Preprocessing einzelne Anforderungen voneinander aus einem Fließtext abgegrenzt werden?

(3.2) Kann eine Software inhaltlich ähnliche, aber unterschiedlich formulierte Anforderungen erkennen und einander zuordnen?

Insgesamt decken die Fragestellungen drei Aspekte der entwickelten Methode ab, die die Validierung der Methode in folgende drei Unterkapitel unterteilt:

4.2 Automatisierung des Anforderungsvergleichs	Fragen (3.1), (3.2)
4.3 Klassifizierung der Anforderung zur Lastenheftkonfiguration	Fragen (1.1), (1.2)
4.4 Anforderungen als Variantentreiber	Fragen (2.1), (2.2)

Abschließend werden die Ergebnisse der Anwendung und die oben vorgestellten Fragen in Kapitel 4.5 diskutiert.

4.1 Anwendung am Beispiel der Fahrwerksentwicklung der BMW AG

Die Methode wird in der Entwicklung Fahrdynamik der BMW AG angewendet. Die Entwicklung Fahrdynamik verantwortet alle Komponenten des Fahrwerks, sowohl mechanische und elektronische Bauteile als auch Softwarekomponenten (Abbildung 4-1).

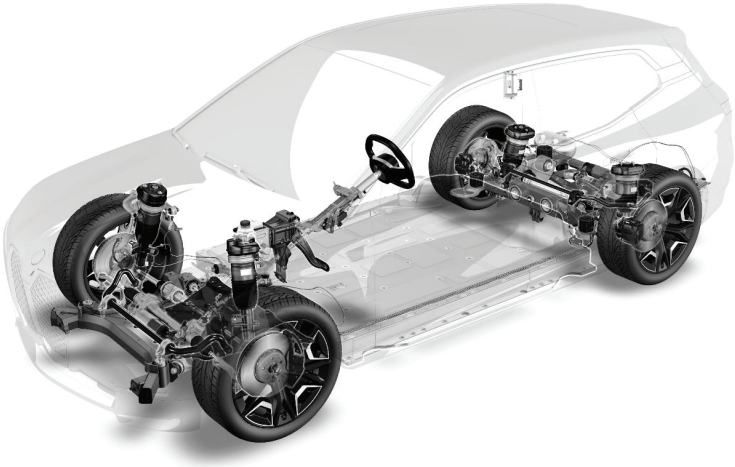


Abbildung 4-1: Fahrwerk des BMW iX mit seinen Komponenten (BMW AG 2021)

Für die Anwendung der entwickelten Methode wird der Fokus auf mechanische Komponenten des Fahrwerks gelegt. Die Vergabe einer mechanischen Komponente an einen Lieferanten besteht aus einer Vielfalt an Dokumenten: Das Komponentenlastenheft, technische Datenblätter zur Bauteilauslegung, Berechnungs-, Prüf- und Qualitätsvorschriften, Unternehmensstandards, CAD-Modelle und technische Zeichnungen. Das vertragliche Dokument ist das Lastenheft, dass auf die mitgeltenden Unterlagen verweist. Die Anwendung des hier entwickelten Anforderungsvergleichs erfolgt ausschließlich auf Komponentenlastenhefte. Es werden keine mitgeltenden Unterlagen als Anforderungsdokumente verglichen. Allerdings werden sie z.T. zum Antrainieren des Sprachmodells genutzt. Da die Vertragssprache im Anwendungsfall deutsch ist, wird das Sprachmodell und somit auch der automatisierte Anforderungsvergleich auf deutschsprachige Anforderungen angewendet.

Die Komponentenlastenhefte liegen entweder als Textdokument in dem Format .docx oder als Exportdatei aus der Datenbank DOORS in einem .xlsx-Dateiformat vor. Der Export eines Lastenheftes aus der Datenbank DOORS weist in Excel folgende Struktur für den Vergleich auf: In der Spalte A befindet sich eine Identifikationsnummer (ID), die in DOORS vergeben wird. In Spalte B steht der zur jeweiligen ID gehörige Anforderungstext. Kapitelüberschriften im Lastenheft haben in dieser Struktur ebenfalls eine ID. Sie können in der Exportdatei durch eine Textformatierung „fett“ erkannt werden. Abbildungen, Diagramme und Tabellen, die als OLE-Objekt in der Datenbank eingebettet sind, werden nicht exportiert. Hier bleibt die Zelle in Spalte B hinter der zugehörigen ID leer.

Die Lastenhefte im Textformat .docx weisen unterschiedliche Formatierungen auf. Der Großteil der Lastenhefte besteht aus drei Spalten: Die erste Spalte enthält die DOORS ID, die zweite die deutschsprachige Anforderung und die dritte Spalte enthält die englischsprachige Anforderung (Abbildung 3-3). Für den Anforderungsvergleich wird ausschließlich der Anforderungstext der deutschsprachigen Spalte verwendet. Zur Verwendung des Lastenheftes als Vertragsdokument und Vergabe an einen Lieferanten wird das Dokument in ein .pdf-Dateiformat übertragen.

Die Anwendung der Methode bei der BMW AG erfolgt in Kooperation mit der Firma cortical.io. Die Kooperation mit der Firma cortical.io sowie die Abgrenzung des Leistungsumfangs soll an dieser Stelle kurz erläutert werden. Das im Rahmen dieser Arbeit angewandte Verfahren zum semantischen Vergleich Semantic Folding wurde von Webber (2016) entwickelt und wird durch die Firma cortical.io vertrieben. Der Kooperationspartner cortical.io wurde im Rahmen der Zusammenarbeit damit beauftragt,

die Programmierung der in Kapitel 3.2 erläuterten Regeln und Anforderungen an eine Software zu übernehmen. Für das Antrainieren des Sprachmodells als Semantic Map wurden der Firma cortical.io geeignete Trainingsdokumente zur Verfügung gestellt (Kapitel 4.1.1), das Erzeugen der Semantic Map erfolgt durch cortical.io. Des Weiteren wurden dem Kooperationspartner für den Anwendungsfall Musterbeispiele bereitgestellt (Kapitel 4.1.1), mit deren Hilfe gemeinsam in drei Iterationsschleifen der Grenzwert bestimmt wurde.

Das Ergebnis der Zusammenarbeit ist eine Software, in der der definierte Sprachumfang von zwölf mechanischen Fahrwerkskomponenten in einem Sprachmodell abgebildet ist und durch die ein automatisierten Vergleich der Anforderungen und eine Auswertung deren semantischen Ähnlichkeit möglich ist. Die Auswahl der Trainingsdokumente als auch die Erarbeitung der Musterbeispiele zur Einstellung des Grenzwertes wird im Folgenden erläutert. Bei der Verwendung der Vergabedokumente werden zwei Fälle unterschieden: 1. Dokumente zum Antrainieren des Sprachmodells und 2. Dokumente zu Validierung, die automatisiert miteinander verglichen werden. Die Auswahl geeigneter Dokumente für die beiden Fälle wird im folgenden Abschnitt beschrieben.

4.1.1 Dokumente zum Antrainieren des Sprachmodells

Zum Antrainieren des Sprachmodells werden Dokumente der folgenden zwölf Fahrwerkskomponenten ausgewählt:

- Lenksäule,
- Lenkmanschette,
- Lenkspindel,
- konventionelles Federbein (Stoßdämpfer),
- Stützlager,
- Trag-/ Zusatzfeder,
- Bremsgerät,
- Radbremse,
- Achsträger,
- Radträger,
- Schwenklager
- und Radlager.

Abbildung 4-2 zeigt beispielhaft Bilder der Fahrwerkskomponente, die zur Anwendung der Methode betrachtet wurden.

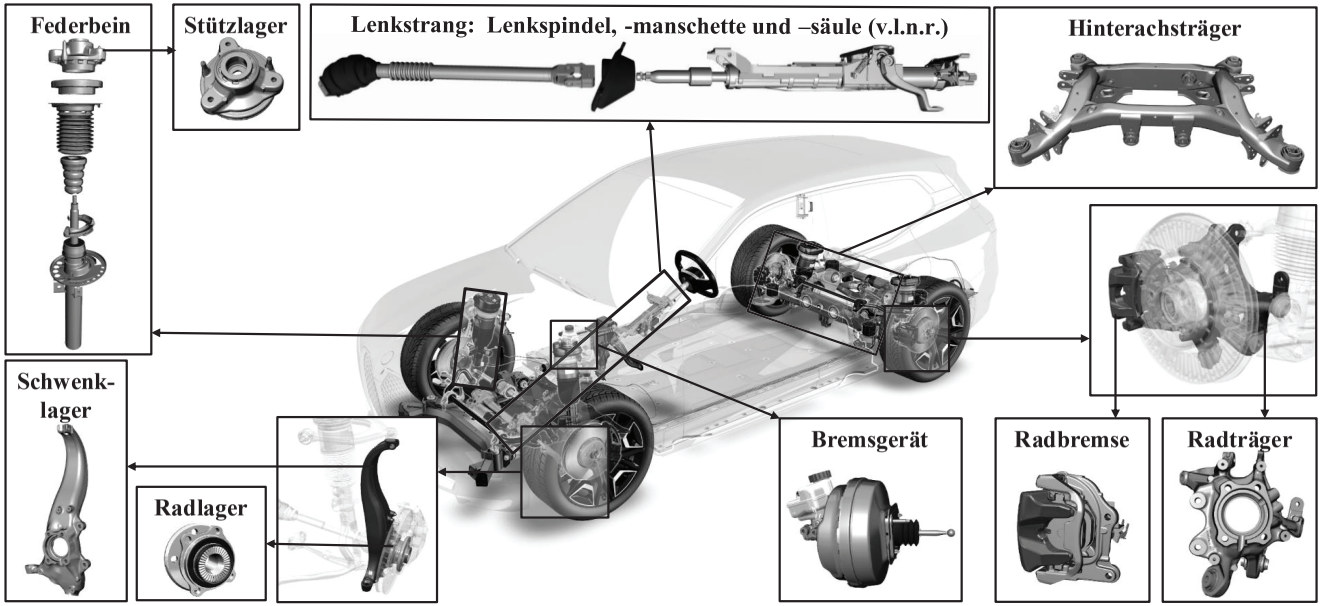


Abbildung 4-2: Übersicht der Fahrwerkskomponenten zur Validierung
(Eigene Darstellung unter Verwendung von Konstruktionsdaten der BMW AG sowie BMW AG 2021)

Insbesondere die Auswahl und Bereitstellung geeigneter Trainingsdaten ist allgemein für eine künstliche Intelligenz für die spätere Qualität der Ergebnisse des Algorithmus kritisch. Wie bereits in Kapitel 2.3.1 erläutert, benötigen die Ansätze des Deep Learning und KNN besonders große Trainingsdatensätze (Aust 2021, S. 47), die je nach Anwendungsbereich nicht immer vorhanden sind. Ein weiteres Problem stellt die Qualität der Trainingsdaten dar. Was plakativ mit der Aussage „garbage in – garbage out“ von Blanche (1957, S. 411) beschrieben wird, ist das Problem, dass die Ergebnisse eines Algorithmus, egal wie gut der Algorithmus selbst ist, nicht besser sein können, als die Daten, die dem Algorithmus zur Verfügung gestellt wurden. Sind zum Beispiel die gekennzeichneten Trainingsdaten für einen supervised-learning Ansatz fehlerhaft, lernt der Algorithmus diesen Fehler mit. Gleiches gilt auch für die Auswahl der Trainingsdaten, die repräsentativ für den Anwendungsfall stehen müssen und nicht nur spezifische Einzelfälle abbilden dürfen. Einige der bekanntesten Beispiele von KI mit schlechter Datengrundlage, die rassistisches oder sexistisches Verhalten aufgrund der Trainingsdaten gelernt haben, sind Microsofts Chatbot Tay auf Twitter (Hunt 2016) oder Googles Bilderkennung (Barr 2015). Daher werden bei der Auswahl der Trainingsdaten für den hier beschriebenen Anwendungsfall Komponentenlastenhefte so ausgewählt, dass sie die Bandbreite der Varianten aus verschiedenen Fahrzeugen, Technologien und Erscheinungsjahren repräsentieren z.B. unterschiedliche Lastenhefte für Vorder- und Hinterachse, Allrad-, Heck- oder Frontantrieb sowie die Zeitspanne der Erscheinungsjahre von 2004 – 2017 von 132 verschiedenen Fahrzeugderivaten aller Produktlinien und Konzern-Marken (einschließlich Rolls Royce und Mini).

Des Weiteren werden s.g. Musterlastenhefte verwendet, die für die einzelnen Komponenten bereits manuell als Teilsammlung möglicher Anforderungen durch die Entwickler erstellt wurden. Zusätzlich wurden für einige Komponenten Qualitätsvorschriften erarbeitet, die ebenfalls zum Antrainieren des Sprachmodells verwendet werden.

Eine Übersicht über die Anzahl der komponentenspezifischen Dokumente zum Antrainieren des Sprachmodells gibt Tabelle 4-1. Neben den komponentenspezifischen Dokumenten werden sowohl Unternehmensstandards als auch öffentlich zugängliche Literatur zur Fahrwerksentwicklung und Maschinenbau und allgemeine Standards bestehend aus VDI- und VDA-Richtlinien sowie DIN-Normen genutzt (Anhang D). Die Unternehmensstandards enthalten bauteilübergreifende Themen wie z.B. Bauteilkennzeichnung, Gefahrenstoffe oder Recyclinganforderungen. Insgesamt werden 88 Komponentenlastenhefte, zehn Musterlastenhefte, neun Abkürzungsverzeichnisse zu den Komponentenlastenheften, 109 BMW-spezifische Qualitätsvorschriften, 50 BMW-

spezifische Unternehmensstandards, acht allgemeine Standards (VDI- und VDA-Richtlinien sowie DIN-Normen), sowie 16 Bücher aus der Fachliteratur zu Maschinenbau und Fahrwerksentwicklung verwendet.

Komponente	Lastenheft	Musterlastenheft	Qualitätsvorschrift
Radträger	6	1	19
Schwenklager	8		14
Achsträger	10	1	45
Radlager	6	1	1
Lenkstrang	10	1	0
Stoßdämpfer / Luftfederbein	12 / 3	1 / 1	0
Stützlager	11	1	0
Trag- / Zusatzfeder	10	1	0
Bremsgerät	6	1	0
Radbremse	6	1	0
übergreifend			30
Summe	**88**	**10**	**109**

Tabelle 4-1: Komponentenspezifische Dokumente zum Antrainieren des Sprachmodells

Diese Dokumente dienen dem Antrainieren des Sprachmodells und der Erzeugung einer Semantic Map. Anhand dieser Dokumente ergibt sich eine Karte des Sprachumfangs. Durch diese Semantic Map kann von der Software für jedes Textfragment ein Semantic Fingerprint generiert werden.

4.1.2 Musterbeispiele zur Bestimmung des Grenzwertes

Vor der eigentlichen Anwendung des Anforderungsvergleichs muss noch ein Grenzwert bestimmt werden, ab dem zwei Textfragmente als inhaltlich gleich gelten sollen (Kapitel 3.2.3.2). Diese Ermittlung erfolgt anhand von Musterbeispielen. Diese Beispiele werden manuell erarbeitet. Dazu wird eine Anforderung in einem Lastenheft ausgewählt und in anderen Lastenheft nach einer gleichen Anforderung über die Stichwortsuche unter Nutzung von Synonymen gesucht. Anhand dessen wird eine Tabelle erzeugt, in der die Ergebnisse der Stichwortsuche festgehalten werden. Diese ist analog zur Ergebnistabelle des Anforderungsvergleichs (Tabelle 3-7) aufgebaut.

Jede Spalte steht für ein Dokument. In den Zeilen stehen jeweils die gefundenen, inhaltlich ähnlichen Anforderungen. Wenn in einem Dokument keine ähnliche Anforderung gefunden wird, wird die Zelle leer gelassen (Tabelle 4-2). Grenzfälle, die trotz einer gewissen inhaltlichen Überschneidung nicht als semantisch ähnlich eingestuft werden sollen, werden als False Positive Beispiele in einer separaten Spalte festgehalten.

Anforderung	Dokument A	Dokument B	Dokument C
Die Möglichkeit eines Falschverbaus ist auszuschließen.	Die Möglichkeit eines Falschverbaus ist auszuschließen.	Ein Falschverbau darf nicht möglich sein.	

Tabelle 4-2: Aufbau eines Musterbeispiels zur Bestimmung des Grenzwertes

Diese Musterbeispiele werden für die Komponenten Radträger, Schwenklager, Hinterachsträger, Lenksäule, Lenkspindel, Stoßdämpfer und Bremsgerät erstellt. Die genaue Anzahl der je Komponente erstellten Prüfbeispiele zeigt Tabelle 4-3. Anhand der Gesamtzahl der erarbeiteten Musterbeispiele und der Anzahl der betrachteten Komponenten kann der manuelle Aufwand eingeschätzt werden, der für das Antrainieren eines neuen Sprachraums erforderlich wird.

Komponente	Komponenten-lastenhefte	Musterbeispiele/ Zeilen
Bremsgerät	7	15
Hinterachsträger	6	5
Lenksäule	6	25
Lenkspindel	6	25
Radträger	7	15
Schwenklager	9	16
Stoßdämpfer, konv.	13	24

Tabelle 4-3: Übersicht der erarbeiteten Musterbeispiele zur Grenzwertbestimmung

Die zur Erstellung der Musterbeispiele verwendeten Lastenhefte werden durch die Software miteinander verglichen. Anschließend wird das Ergebnis des automatischen Vergleichs hinsichtlich der definierten Musterbeispiele manuell kontrolliert. Liegt ein False Positive Beispiel in einer Zuordnung vor, so muss der Grenzwert höher angesetzt werden. Werden zu viele inhaltlich ähnliche Anforderung bei der Zuordnung nicht gefunden, d.h. die Zelle bleibt leer, muss der Grenzwert herabgesetzt werden. Dieser

Prozess erfolgt iterativ, bis ein Optimum der Prüfbeispiele erreicht wird. Der so ermittelte Grenzwert für die Überlappung zweier Semantic Fingerprints liegt bei 35%. Textfragmente, die eine höhere Überlappung haben, werden als inhaltlich gleich betrachtet. Textfragmente, die eine Überlappung von unter 35% haben, werden als verschieden angesehen. Die oben gezeigte Auflistung von Dokumenten zeigt nur Dokumente zum Antrainieren des Sprachmodells. Für die Anwendung bzw. zur Validierung des automatisierten Anforderungsvergleichs werden Komponentenlastenhefte wie im folgenden Kapitel beschrieben ausgewählt.

4.1.3 Dokumente zur Validierung des automatisiert. Anforderungsvergleichs

In der Validierung des Anforderungsvergleichs werden jeweils zwei Lastenhefte der gleichen Komponente miteinander verglichen, sodass insgesamt 42 Lastenheftvergleiche erzeugt werden. Die verglichenen Lastenhefte sind nicht deckungsgleich zu den Lastenheften zum Antrainieren des Sprachmodells (Kapitel 4.1.1). Für den Vergleich werden Lastenhefte der Komponenten Hinterachsträger, Lenksäule, Lenkspindel, Radlager, Schwenklager, Stoßdämpfer und Stützlager verwendet. Die Kombinationen der beiden zu vergleichenden Lastenhefte erfolgt anhand der folgenden Faktoren in den Ausprägungen gleich oder ungleich:

- **Fahrzeuggeneration:** Der Zeitraum zwischen dem Beginn der Serienproduktion (SOP) der Erstanläufer[1] der zu vergleichenden Lastenhefte ist größer oder gleich fünf Jahre.
- **Fahrzeugarchitektur:** hier z.B. Front- oder Heckantriebsarchitektur
- **Produktlinie**
- **Derivat:** verschiedene Karosserieformen eines Produktes z.B. 3er Limousine, Cabrio, Coupe, 3er Kombi und SUV (s. Kapitel 2.2.1 Variantensystematik)
- **Antriebsart:** Verbrennungsmotor (ICE), Elektroantrieb (BEV) oder Plug-in-Hybrid (PHEV)

In Abbildung 4-3 werden auf der linken Seite alle der insgesamt 2^5 Kombinationen der Faktoren aufgezeigt. Da einige Kombinationen in der Praxis nicht auftauchen, reduzieren sich die Kombinationsmöglichkeiten der zu vergleichenden Lastenhefte auf insgesamt elf Kombinationen (rechte Seite der Abbildung). Zu diesen elf Kombinationen werden für die Komponenten jeweils zwei Lastenhefte ausgewählt und miteinander verglichen.

[1] Als Erstanläufer wird das Produkt (hier Fahrzeug) bezeichnet, in dem die Technologie bzw. die Komponente erstmalig verbaut wird.

Versuch	Fahrzeug-generation	Fahrzeug-architektur	Produkt-linie	Derivat	Antriebs-art
1	≠	≠	≠	≠	≠
2	=	≠	≠	≠	≠
3	≠	=	≠	≠	≠
4	=	=	≠	≠	≠
5	≠	≠	=	≠	≠
6	=	≠	=	≠	≠
7	≠	=	=	≠	≠
8	=	=	=	≠	≠
9	≠	≠	≠	=	≠
10	=	≠	≠	=	≠
11	≠	=	≠	=	≠
12	=	=	≠	=	≠
13	≠	≠	=	=	≠
14	=	≠	=	=	≠
15	≠	=	=	=	≠
16	=	=	=	=	≠
17	≠	≠	≠	≠	=
18	=	≠	≠	≠	=
19	≠	=	≠	≠	=
20	=	=	≠	≠	=
21	≠	≠	=	≠	=
22	=	≠	=	≠	=
23	≠	=	=	≠	=
24	=	=	=	≠	=
25	≠	≠	≠	=	=
26	=	≠	≠	=	=
27	≠	=	≠	=	=
28	=	=	≠	=	=
29	≠	≠	=	=	=
30	=	≠	=	=	=
31	≠	=	=	=	=
32	=	=	=	=	=

= gleich
≠ verschieden

Versuch	Fahrzeug-generation	Fahrzeug-architektur	Produkt-linie	Derivat	Antriebs-art
1	≠	≠	≠	≠	≠
2	=	≠	≠	≠	≠
4	=	=	≠	≠	≠
8	=	=	=	≠	≠
16	=	=	=	=	≠
17	≠	≠	≠	≠	=
18	=	≠	≠	≠	=
20	=	=	≠	≠	=
21	≠	≠	=	≠	=
29	≠	≠	=	=	=
32	=	=	=	=	=

Abbildung 4-3: Faktorkombinationen der zu vergleichenden Lastenhefte

4.2 Automatisierung des Anforderungsvergleichs

Der erste Baustein der Validierung der entwickelten Methode ist die Überprüfung des automatisierten, semantischen Anforderungsvergleichs. Dabei wird das in Kapitel 3 vorgestellten Sprachmodell zunächst hinsichtlich der Güte des Preprocessing, der Zerlegung des Fließtextes in einzelne, abgrenzbare Anforderungen, überprüft und anschließend die Güte des Erkennens von semantisch ähnlichen und semantisch unterschiedlichen Anforderungen durch das Sprachmodell ausgewertet. Dadurch sollen die ersten beiden Forschungsfragen beantwortet werden können:

(3.1) Wie können durch definierte Zerlegungsregeln im Preprocessing einzelne Anforderungen voneinander aus einem Fließtext abgegrenzt werden?

(3.2) Kann eine Software inhaltlich ähnliche, aber unterschiedlich formulierte Anforderungen erkennen und einander zuordnen?

4.2.1 Vorgehen

Allgemein werden die Ergebnisse einer KI zur Klassifikation durch eine Konfusionsmatrix, auch Wahrheitsmatrix, validiert und in den Verhältnissen der Werte als Güte-Kennzahlen ausgegeben. Eine **Konfusionsmatrix** (Abbildung 4-4) besteht aus zwei Zeilen und zwei Spalten.

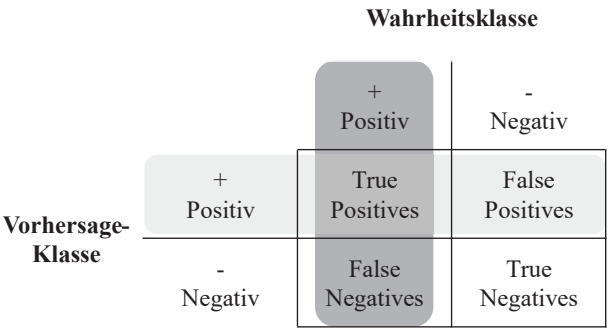


Abbildung 4-4: Konfusionsmatrix (Fawcett 2006, S. 862)

Die Zeilen zeigen die durch den Algorithmus vorhergesagte Klasse Positiv oder Negativ. Die Spalten zeigen die Wahrheitsklasse in Positiv und Negativ. Entspricht die vorhergesagte Klasse der Wahrheitsklasse, handelt es sich um einen korrekten Wert, der mit true bezeichnet wird. Stimmen vorhergesagte Klasse und Wahrheitsklasse nicht überein, handelt es sich um eine falsche Prognose, die mit false bezeichnet wird.

Dadurch ergeben sich vier mögliche Werte: **True Positive TP, False Positive FP, False Negative FN** und **True Negative TN**.

In diese vier Felder werden alle Prognoseergebnisse der KI kategorisiert und anschließend je Feld die Summe gebildet. Die Felder False Negatives und False Positives zeigen die Prognosefehler. Je nach Anwendungsfall ist aber ein Prognosefehler kritischer als der andere und muss entsprechend anders optimiert werden. So wäre bei der Auswertung der Prognosefehler bei z.B. Erdbebenwarnungen die false negatives besonders kritisch, da ein Erdbeben auftritt, ohne das eine Warnung erfolgte (Ng et al. 2018, S. 15). Anders sieht es z.B. bei Mammografie-Screenings als Brustkrebs-Testverfahren aus. Ein hohes false positives Ergebnis würde dazu führen, dass der Test besonders häufig Krebs diagnostiziert, ohne dass eine Erkrankung vorliegt. Dies kann zu hohen seelischen Belastungen der Betroffenen führen (Brodersen et al. 2013) und gleichzeitig ohne weiteren Test zu Krebstherapien bei gesunden Menschen führen. Daher sind diese Werte und die Güte-Kennzahlen immer vor dem Hintergrund des Anwendungsfalls zu hinterfragen.

Aus diesen vier Werten werden folgende Kennzahlen zur Messung eines Klassifikationsalgorithmus ausgegeben und gemäß der Formeln berechnet (Aust 2021, S. 59ff.): Die **Wirksamkeit (precision)** beschreibt den Anteil der korrekt positiven Ergebnisse im Verhältnis zur Summe aller als positiv klassifizierten Ergebnisse (Summe der Zeile Vorhersageklasse positiv).

$$precision = \frac{\sum TP}{\sum TP + \sum FP} \tag{5}$$

Die **Treffergenauigkeit (accuracy)** beschreibt den Anteil der korrekt klassifizierten Ergebnisse, unabhängig der positiven oder negativen Klassifikation.

$$accuracy = \frac{\sum TP + \sum TN}{\sum TP + \sum FP + \sum TN + \sum FN} \tag{6}$$

Als **Sensitivität (recall)** bezeichnet man den Anteil der korrekt positiven Ergebnisse an der Summe der in Wahrheit positiven Ergebnissen, also der Summe der Spalte Wahrheitsklasse positiv.

$$recall = \frac{\sum TP}{\sum TP + \sum FN} \tag{7}$$

Abschließend ist hier noch die gängige Kennzahl **F1-Score** als harmonisches Mittel aus precision und recall genannt.

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{8}$$

Die Werte der Gütekriterien precision und recall werden von Hayes et al. (2005) für den Anwendungsfall einer Methode zur automatisierten Verknüpfung von Anforderungen mit den Designspezifikationen im Kontext der Softwareentwicklung als s.g. requirements traceability matrix, einer Anforderungsnachverfolgungsmatrix, gemäß Tabelle 4-4 definiert. Diese Einstufung wird ebenfalls von Boutkova (2014, S. 60ff.) zur Bewertung der Prognosegüte des semiautomatischen Ansatzes MIA zur sprachlichen Anforderungsanalyse verwendet.:

Gütekriterium	Akzeptabel	Gut	Exzellent
Recall	>60%	>70%	>80%
Precision	>20%	>30%	>50%

Tabelle 4-4: Einstufung der Kriterien precision und recall für die Prognosegüte (Hayes et al. 2005, S. 35)

Zur Validierung sind zwei Datensätze zu unterscheiden: Der Trainingsdatensatz, der zum Antrainieren der KI verwendet wird, und der Testdatensatz, der zur Validierung der Prognoseergebnisse der KI verwendet wird (Ng et al. 2018, S. 16). So werden zur Validierung für die KI neue, unbekannte Daten als Input durch den Algorithmus klassifiziert und nicht die bereits durch das Trainieren der KI bekannten Daten. Liegen insgesamt für den Anwendungsfall geringe Datenmengen vor, sodass eine Aufteilung die Menge an Trainingsdaten zu massiv reduzieren würde, kann eine Kreuzvalidierung durchgeführt werden. Dabei werden die vorhandenen Daten in Segmente unterteilt und die Validierung erfolgt iterativ, sodass jedes Segment einmal als Testdatensatz verwendet wurde und in jeder Iterationsschleife ein anderer Trainingsdatensatz bestehend aus den anderen Datensegmenten verwendet wurde. Anschließend wird für die zuvor erläuterten Güte-Kennzahlen der Mittelwert über alle Iterationen ermittelt. (Ng et al. 2018, S. 16)

Die Validierung erfolgt anhand der in Kapitel 4.1.3 beschriebenen 42 Lastenheftvergleiche (Anhang E). Dabei werden die Ergebnisse der Zuordnung des Lastenheftvergleichs zweier Dokumente zeilenweise manuell geprüft. Jede Zeile besteht dabei aus einem Textfragment 1 und dem zugeordneten, semantisch ähnlichen Textfragment 2

oder ggf. einer Leerzelle, falls kein semantisch ähnliches Textfragment in dem anderen Dokument gefunden wurde. Der Aufbau der Auswertungstabelle entspricht dem in Kapitel 3 beschriebenen Aufbau. Jede Zeile dieser Auswertungstabelle wird im Folgenden als ein **Match** bezeichnet. Insgesamt sind durch die Vergleiche 13.546 Matches entstanden, die manuell auf ihre korrekte Zuordnung kontrolliert werden.

Sind in einer Zeile zwei Textfragmente als semantisch ähnlich einander zugeordnet, so handelt es sich um einen positiven Match. Wird bei der Kontrolle festgestellt, dass die inhaltliche Zuordnung dieser Textfragmente richtig ist, handelt es sich um ein **True Positive (TP)**. Handelt es sich aber um zwei inhaltlich verschiedene Textfragmente, wird der Match als **False Positive (FP)** gekennzeichnet.

Besteht ein Match aus einem Textfragment und einer leeren Zelle, so konnte das Sprachmodell keine passende, inhaltlich ähnliche Anforderung finden. Hierbei handelt es sich um ein negatives Match. Ist es tatsächlich richtig, dass kein inhaltlich ähnliches Textfragment in dem anderen Dokument bei der manuellen Überprüfung gefunden werden kann, wird das Match als **True Negative (TN)** markiert. Wird allerdings bei der manuellen Überprüfung in dem zweiten Dokument ein passendes Textfragment gefunden, handelt es sich bei dem Match um ein **False Negative (FN)**.

Wird bei der Überprüfung hingegen festgestellt, dass ein Textfragment mitten im Satz abgeschnitten wurde, das Textfragment nur aus einem Stichpunkt ohne Kontext besteht oder aus einem anderen Grund der Kontext des Textfragmentes fehlt, wird dieses Match mit einem **Zerlegungsfehler (Z)** markiert, da es sich hierbei nicht um ein Problem der semantischen Zuordnung, sondern um einen Fehler im Preprocessing der Textdokumente handelt.

4.2.2 Ergebnisse

Das Ergebnis dieser manuellen Kontrolle von 13.546 Matches setzt sich wie folgt zusammen (Abbildung 4-5): Insgesamt wurde bei 992 Matches ein Zerlegungsfehler gefunden, 6041 sind True Positives, 5537 True Negatives, 211 False Positives und 756 False Negatives.

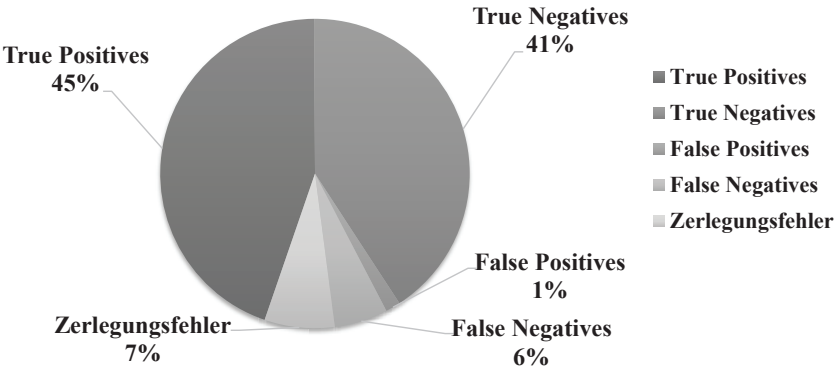


Abbildung 4-5: Auswertung der semantischen Zuordnung – Anteile der Matches

(3.1) Wie können durch definierte Zerlegungsregeln im Preprocessing einzelne Anforderungen voneinander aus einem Fließtext abgegrenzt werden?

Für die Bewertung der Güte des Preprocessing, der Textzerlegung, zeigt diese Auswertung, das von 13.546 Matches 7%, bzw. 992 der Matches, einen Zerlegungsfehler aufweisen und daher eine falsche Zuordnung zur Folge haben. Eine detailliertere Analyse dieser Zerlegungsfehler zeigt folgende Hauptgründe für ein fehlerhaftes Preprocessing des Fließtextes der Anforderungsdokumente:

Die größte Fehlerquelle mit 32% aller Zerlegungsfehler stellen Aufzählungen dar. Aufzählungen können in den Lastenheften unterschiedlich formatiert sein: In einigen Fällen sind sie durch Spiegelstriche, Aufzählungspunkte oder Nummerierungen gekennzeichnet, in anderen nicht. In manchen Fällen ist ein Stichpunkt ein vollständiger Satz, in anderen Fällen besteht er nur aus Zahlen mit Einheiten oder enthält kein Prädikat. In allen Fällen wird die Aufzählung allerdings mit einem einleitenden Satz begonnen. Als Signal für diesen einleitenden Satz kann an dessen Ende ein Doppelpunkt stehen oder mit der Konjunktion „dass“ und einer Absatzmarke enden. Alle auf diesen Einleitungssatz folgenden Stichpunkte müssen mit dem einleitenden Satz zu einer Einheit verknüpft werden, um den Kontext des Stichpunkts nicht zu verlieren. Schwieri-

ger ist das Problem der Aufzählungen zu lösen, wenn die einzelnen Stichpunkte vollständige Sätze darstellen, die nicht als Aufzählung durch einen Spiegelstrich oder andere Aufzählungszeichen gekennzeichnet sind. Hier ist noch zu untersuchen, wie ein Algorithmus erkennen kann, wann eine solche Aufzählung endet.

Die zweitgrößte Fehlerquelle in der Zerlegung ist mit 25% die Anapher. Anaphern sind Rückwärtsbezüge, die sich auf einen bereits beschriebenen Kontext beziehen. In der Auswertung der Zerlegungsfehler können einige Signalwörter für solche Anaphern identifiziert werden: dies, hierfür, außerdem, alternativ, optional, deshalb, zusätzlich, Anmerkung, wie oben (beschrieben/dargestellt/erläutert) sowie Personalpronomen (er, sie es, sie). Für eine vollständige Auflistung siehe Tabelle 3-4. Aber auch Satzzeichen wie Klammern zeigen eine Anapher. In den meisten Fällen bezieht sich die Anapher direkt auf den vorstehenden Satz, sodass die einfache Regel zur Verknüpfung mit dem voranstehenden Satz in den vorliegenden Anforderungsdokumenten korrekt ist. Allerdings gibt es auch Ausnahmen. Schwieriger sind Anaphern, bei denen die Subjekte wie „Die Messung“, „Die Mindestabstände“, „Die Berechnung“, „Die Grenzwerte“, „Die Originaldateien“, „Die Anforderungen“, „Die Konstruktion“, „Der Durchmesser“ Anaphern darstellen, da erst in ihrem jeweiligen vorangegangenen Kontext deutlich wird, was mit dem jeweiligen Subjekt gemeint ist. Diese können eher nicht als regelbasierter Schlüsselbegriff festgelegt werden.

Die Kataphern als Vorwärtsbezug sind die Ursache von 15% der Zerlegungsfehler. Auch hier lassen sich gewisse Signalwörter identifizieren: im Folgenden, folgende (-n, -s,-r), nachstehend (-e,-en), nachfolgend (-e,-er,-es,-en), folgendermaßen, lt. unteren (Werten/…), untenstehend (-e, -en), dass und das Satzzeichen Doppelpunkt (siehe Tabelle 3-4). Häufig leiten Kataphern auch eine Aufzählung ein. In solchen Fällen greift wieder die vereinfachte Regel, solche Fragmente mit den darauffolgenden Fragmenten zu verknüpfen. Kataphern, die allerdings auf eine ganz andere Stelle im Dokument selbst verweisen, lassen sich nicht so einfach regelbasiert verknüpfen. Nur bei expliziter Benennung einer Abbildungsnummer oder einer Kapitelnummer ließe sich eine solche Katapher mit dem jeweiligen Inhalt verknüpfen.

Weitere Gründe für Zerlegungsfehler stellen zerlegte Datentabellen (8%), Satzzeichen (3,6%), Beschriftungen/Überschriften (2,9%), bereits unvollständige Sätze bzw. fehlende Satzzeichen im Ursprungsdokument (1,9%), Kombinationen aus Anaphern und Kataphern innerhalb eines Fragmentes (0,7%) sowie andere, nicht klassifizierbare Ursachen (10,5%) dar.

Tabelle 4-5 zeigt ein Beispiel für den Zerlegungsfehler Satzzeichen. Wie bereits Carstensen et al. (2010, S. 264ff.) feststellt, ist nicht jeder Punkt gleichzusetzen mit dem Ende eines Satzes und somit das Ende eines Tokens. Im gezeigten Beispiel ist der Punkt mehrmals als Abkürzung verwendet worden, allerdings vom Preprocessing nicht immer als solcher erkannt worden, sodass ein zusammenhängender Satz in drei Fragmente zerteilt wurde.

Dokument A - Original	Dokument A - nach dem Preprocessing
Teilekennzeichnung Produktspezifische Daten (Sach-Nr., Änderungsindex, Herstelldatum, Laufende-Nr., etc.) müssen in einem Aufkleber mit Data Matrix Code nach [GS 91010-2] sowie in Klarschrift enthalten sein.	Teilekennzeichnung: Produktspezifische Daten (Sach-Nr. Teilekennzeichnung: , Änderungsindex, Herstelldatum, Laufende-Nr. Teilekennzeichnung: , etc.) müssen in einem Aufkleber mit Data Matrix Code nach [GS 91010-2] sowie in Klarschrift enthalten sein.

Tabelle 4-5: Beispiel für Zerlegungsfehler durch Satzzeichen

Insgesamt zeigt das Preprocessing eine Güte von 93% korrekt zerlegten Matches. Die festgestellten Zerlegungsfehler lassen sich größtenteils durch Erweiterungen des Regelwerkes bei der Zerlegung beheben. Daher kann man an dieser Stelle festhalten, dass es mit Hilfe der Zerlegungsregeln möglich ist, automatisiert einzelne Anforderungen aus Fließtexten der Anforderungsdokumente zu identifizieren und abzugrenzen.

(3.2) Kann eine Software inhaltlich ähnliche, aber unterschiedlich formulierte Anforderungen erkennen und einander zuordnen?

Unabhängig von der Güte des Preprocessing soll die Güte der semantischen Zuordnung der Textfragmente bewertet werden. Dazu wird der Anteil der Zerlegungsfehler an den Matches aus der weiteren Betrachtung ausgeschlossen und nur die korrekt zerlegten Matches betrachtet. Daher sinkt die Anzahl der betrachteten Matches für die Bewertung der semantischen Zuordnung auf 12.545. Die Güte der semantischen Zuordnung erfolgt durch die Berechnung der Kennzahlen precision, recall, accuracy und F1 anhand der Summen von TP, FP, TN und FN. Die ermittelten Werte sind komponentenweise in Tabelle 4-6 dargestellt. Diese Kennzahlen lassen sich durch die Festlegung des Grenzwertes der Überlappung (Kapitel 4.1.2) beeinflussen. Der auf die in dieser Validierung betrachtete Daten angewandte Grenzwert beträgt 35%. Anhand des

gesetzten Grenzwertes entscheidet das Sprachmodell, ob zwei Textfragmente als inhaltlich ähnlich zugeordnet werden sollen. Wird der Grenzwert herabgesetzt, so erhöht sich der Anteil der FP an den positiven Matches und der Anteil der FN sinkt. Dadurch würde sich der recall verbessern und die precision verschlechtern. Wird hingegen der Grenzwert sehr hoch angesetzt, sodass zwei Textfragmente eine sehr hohe inhaltliche Ähnlichkeit aufweisen müssen, um einander zugeordnet zu werden, so verringert sich der Anteil der FP und der Anteil der FN steigt. Die accuracy hingegen bildet das Verhältnis aller korrekt als ähnlich oder verschieden eingestuften Matches zu der Gesamtheit aller Matches ab. Dieser Wert liegt im beschriebenen Anwendungsfall über alle Komponenten bei 0,9233. Das bedeutet, dass 92% der Zuordnung durch das Sprachmodell korrekt sind.

Komponenten	∑ TP	∑ TN	∑ FP	∑ FN	∑ Z	Accuracy	Precision	Recall	F1
Hinterachsträger	1783	2813	106	449	378	0,892	0,944	0,799	0,865
Lenksäule	152	187	8	10	19	0,950	0,950	0,938	0,944
Lenkspindel	326	67	0	0	4	1,000	1,000	1,000	1,000
Radlager	826	617	14	73	158	0,943	0,983	0,919	0,950
Schwenklager	1116	905	38	116	246	0,929	0,967	0,906	0,935
Stoßdämpfer	250	36	1	2	9	0,990	0,996	0,992	0,994
Stützlager	1588	912	44	106	178	0,943	0,973	0,937	0,955
Gesamtergebnis	**6041**	**5537**	**211**	**756**	**992**	**0,923**	**0,966**	**0,889**	**0,926**

Tabelle 4-6: Auswertung des semantischen Vergleichs je Komponente

Eine Erkenntnis, die im Rahmen der semantischen Zuordnung erzielt wurde, ist die Zusammenlegung von inhaltlich ähnlichen Anforderungen innerhalb eines Dokuments zu einem Textfragment, dass mit den anderen Dokumenten verglichen wird. Dieses Vorgehen wurde im Rahmen dieser Arbeit in Zusammenarbeit mit cortical.io unter dem Stichwort **Semantic Merging** untersucht. Zum einen kann durch eine vorherige Untersuchung der semantischen Ähnlichkeit innerhalb eines Dokumentes Duplikate einzelner Anforderungen in diesem Dokument gefunden werden. Die Verknüpfungen dieser Duplikate zu einem zu vergleichenden Textfragment sind sinnvoll, um im Vergleich mit anderen Dokumenten direkt sehen zu können, an welchen Stellen die Duplikate sinnvoll oder überflüssig sind. Dieses Vorgehen lässt sich am besten an folgendem Beispiel verdeutlichen: In den Komponentenlastenheften wird für jede Schnittstelle zu einer benachbarten Komponente ein separates Kapitel angelegt, d.h. jedes

dieser Kapitel beschreibt eine andere Schnittstelle und gibt so den darinstehenden Anforderungen einen anderen Kontext. Jedes dieser Schnittstellenkapitel enthält die Anforderung:

„Die Schnittstelle ist gegen Korrosion zu schützen."

Es handelt sich hier um Duplikate derselben Anforderung, allerdings in unterschiedlichem Kontext. Um nun die Lastenhefte vergleichen zu können, ist es hilfreich, diese Anforderungen aus dem jeweiligen Kapitel mit Überschrift einander gegenüberstellen zu können. Dadurch lässt sich z.B. prüfen, ob die Schnittstellenkapitel vollständig sind.

Zusätzlich dient dieses Zusammenlegen von inhaltlich ähnlichen Textfragmenten innerhalb eines Dokumentes auch dazu, die unterschiedliche Granularität von Anforderungen je nach Autor anzugleichen. So kann in einem Dokument eine Anforderung recht grob für einen übergreifenden Anwendungsfall beschrieben sein und in einem anderen Dokument unterscheidet der Autor die inhaltlich ähnliche Anforderung in verschiedene Fälle oder beschreibt sie detaillierter durch einen zusätzlichen Satz als Information. Diese auch personenabhängige Granularität der Anforderungsbeschreibung muss für den anschließenden Vergleich mit anderen Anforderungsdokumenten angeglichen werden.

Nr.	Dokument A	Dokument B
1	Es gelten folgende Grenzwerte: • Grenzwert xx MnP weltweit: < xx Fälle / 100 Fahrzeuge - Grenzwert xx MnP weltweit: < xx Fälle / 100 Fahrzeuge	Folgende Grenzwerte werden festgelegt: Vorderachse: • Grenzwert xx MnP weltweit: < xx Fälle / 100 Fahrzeuge • Grenzwert xx MnP US: <xx Fälle / 100 Fahrzeuge --- Hinterachse: • Grenzwert xx MnP weltweit: < xx Fälle / 100 Fahrzeuge Grenzwert xx MnP US: <xx Fälle / 100 Fahrzeuge

Tabelle 4-7: Semantic Merging - Zusammenlegung semantisch ähnlicher Textfragmente aufgrund versch. Granularität

Das Beispiel in Tabelle 4-7 zeigt, dass in Dokument A keine Unterscheidung der Grenzwerte für die Gewährleistungsdauer in Vorder- und Hinterachse erfolgt ist. Dokument B weist hierbei eine feinere Granularität auf. Für den Vergleich ist an dieser

Stelle eine Zusammenlegung der unterteilten Grenzwerte sinnvoll, um sie den gröberen Grenzwerten in Dokument A gegenüberstellen zu können. Dennoch wird in der Auswertungstabelle durch das Trennzeichen „---“ deutlich, dass es sich hierbei um zwei unterschiedliche Textfragmente handelt, die auch an unterschiedlichen Stellen im Dokument auftreten können. Durch den Vergleich der Anforderungen basierend auf ihrer inhaltlichen Bedeutung ist eine Transparenz über die unterschiedlichen Formulierungen der gleichen Anforderungen entstanden. Tabelle 4-8 zeigt einige Beispiele, die durch das Sprachmodell korrekt einander zugeordnet wurden, obwohl sie sehr unterschiedlich formuliert wurden.

Nr.	Dokument A	Dokument B
1	3.3 Qualität: Die Entwicklungs- und Fertigungsprozesse bei allen beteiligten Unternehmen sind bezüglich Fehlervermeidung und Zuverlässigkeitsabsicherung so zu gestalten, dass bei Auslieferung und Nutzung des Gesamtfahrzeugs über eine durchschnittliche Lebensdauer (siehe Abschnitt 3.2) in Kundenhand keine Beanstandungen (Null Fehler) auftreten.	Allgemeines: Das Ziel der Qualitätsstrategie für BMW Fahrzeuge und alle das Gesamtfahrzeug betreffenden Komponenten und Systeme bei Auslieferung und Nutzung in Kundenhand ist Null Fehler (keine Beanstandungen in der durchschnittlichen Produktlebenszeit). --- Zur Erreichung dieses Ziels sind sowohl die Entwicklungs- wie auch die Fertigungsprozesse bei allen beteiligten Unternehmen bezüglich Fehlervermeidung und Zuverlässigkeitssicherung optimal zu gestalten.
2	3.6.5 Reparatur- und Montageanforderungen: Ein Falschverbau darf nicht möglich sein.	Reparatur- und Montageanforderungen: Die Möglichkeit eines Falschverbaus ist auszuschließen.
3	3.6.6.3 Schnittstelle verschraubte Lenker zum Achsträger: Für die Einstellmöglichkeit für Vorspur/Sturz muss ein Langloch mit Exzenterabstützung umgesetzt sein.	Schnittstelle Achsträger zu den Spurlenkern: Hierfür ist ein Langloch mit Exzenterabstützung zu realisieren. --- Schnittstelle Achsträger zu den Sturzlenkern: Hierfür ist ein Langloch mit Exzenterabstützung zu realisieren.

Tabelle 4-8: Anforderungsvergleich für korrekt zugeordnete, unterschiedliche Formulierungen derselben Anforderung

Auch hier sind einige Beispiele zu sehen, in denen das Semantic Merging dazu geführt hat, die Granularität zwischen inhaltlich ähnlichen Anforderungen in zwei verschiedenen Dokumenten anzugleichen. Diese sind erkennbar an den Trennzeichen „---“, die die zusammengelegten Textfragmente voneinander abgrenzen. Des Weiteren zeigen die Beispiele, wie wichtig ein Einbezug der jeweiligen Kapitelüberschrift für den Kontext der Anforderung ist. Die Kapitelüberschrift ist jeweils als erstes vor dem Textfragment zu sehen. Dadurch lassen sich unterschiedliche Lastenheftstrukturen, unterschiedliche Kapitelzuordnungen der Anforderungen und Duplikate der Anforderungen in den verschiedenen Kapiteln erkennen.

Diese Beispiele zeigen auch, dass noch Optimierungsbedarf im Preprocessing besteht. So ist im dritten Beispiel Langloch zur Exzenterabstützung im Dokument auf der rechten Seite der Kontext der Funktion Spur-/Sturzeinstellung verloren gegangen. Die Verknüpfung zu einem anderen Satz durch das Wort „hierfür“ wurde im Preprocessing nicht korrekt erkannt.

Insgesamt zeigt das Sprachmodell bei der Identifikation und Zuordnung von inhaltlich ähnlichen Textfragmenten sehr gute Ergebnisse. 92,3% aller Matches sind korrekt (accuracy). Der Anteil fehlerhafter Zuordnungen ist gering. Ein großes Fehlerpotential für fehlerhafte Zuordnungen liegt im Preprocessing. Die s.g. Zerlegungsfehler bei der Segmentierung des Fließtextes in einzelne Anforderungen wurden bei der Berechnung der Kennzahlen herausgerechnet, um den reinen semantischen Vergleich durch die Kennzahlen zu bewerten. Für das Preprocessing kann eine Güte von 7% fehlerhafter Zerlegungen ausgewiesen werden.

4.3 Klassifizierung der Anforderung zur Lastenheftkonfiguration

Als zweiten Baustein der Validierung wird die Klassifizierung der Anforderungen anhand der Ergebnisse des automatisierten Anforderungsvergleichs in die Gruppen projektübergreifend, ausgeprägt und spezifisch überprüft. Des Weiteren wird basierend auf der Auswertung einer Expertenbefragung untersucht, ob sich diese Klassifizierung der Anforderungen zur Erstellung einer Anforderungsbibliothek eignet mit dem Ziel einer Wiederverwendung von Anforderungen und projektspezifischen Konfiguration der vorhandenen Anforderungen. In diesem Kapitel sollen die folgenden beiden Forschungsfragen beantwortet werden:

(1.1) Können Anforderungen basierend auf einem automatisierten Vergleich in die Gruppen projektübergreifend, ausgeprägt oder spezifisch klassifiziert werden?

(1.2) Ist durch die Klassifizierung eine produktspezifische Konfiguration eines Lastenheftes aus einer Anforderungsbibliothek einer Komponente möglich?

4.3.1 Vorgehen

Der in Kapitel 4.2 erläuterte automatisierte Anforderungsvergleich ermöglicht eine Auswertung über das Vorkommen von Anforderungen in den Komponentenlastenheften der unterschiedlichen Entwicklungsprojekte. Anhand des Vorkommens und der Identifizierung bestimmter Merkmale einer Ausprägung können Anforderungen automatisiert in drei Gruppen klassifiziert werden: projektübergreifend, spezifisch und ausgeprägt (Abbildung 3-1). Im beschriebenen Anwendungsfall (Kapitel 4.1) werden jeweils zwei Komponentenlastenhefte derselben Komponente aus verschiedenen Entwicklungsprojekten miteinander verglichen. Anschließend erfolgt basierend auf dem Vergleich eine automatische Eingruppierung der Matches in eine der drei Gruppen. Dabei werden folgende Regeln angewandt: Wird eine Anforderung in beiden Lastenheften gefunden und sie weist kein Merkmal für eine Ausprägung auf, wird sie als projektübergreifend klassifiziert. Wenn keine ähnliche Anforderung in dem anderen Lastenheft gefunden wird und die Anforderung kein Merkmal für eine Ausprägung aufweist, gilt sie als spezifisch. Wird hingegen ein Merkmal für eine Ausprägung erkannt, wird die Anforderung direkt als ausgeprägt klassifiziert. Die zum Zeitpunkt der Validierung definierten Merkmale für eine Ausprägung sind: Zahlenwerte mit Einheit, Farben, Materialien und Antrieb. Die in Kapitel 3 benannten Merkmale wurden um die nach der Auswertung identifizierten Merkmale erweitert.

Abbildung 4-6 zeigt die Zusammensetzung der eingruppierten Matches nach dem automatisierten Vergleich. Hiernach sind die projektspezifischen Anforderungen mit 53% die größte Gruppe. Der Anteil der ausgeprägten Anforderungen ist von allen drei Gruppen mit 7% am geringsten. Die projektübergreifenden Anforderungen sind trotz bestehender Wiederverwendung durch das Copy-Paste-Vorgehen mit 40% die zweitgrößte Gruppe. Die Überprüfung dieser automatisierten Eingruppierung der Anforderungen auf korrekte Eingruppierung wird nun erläutert.

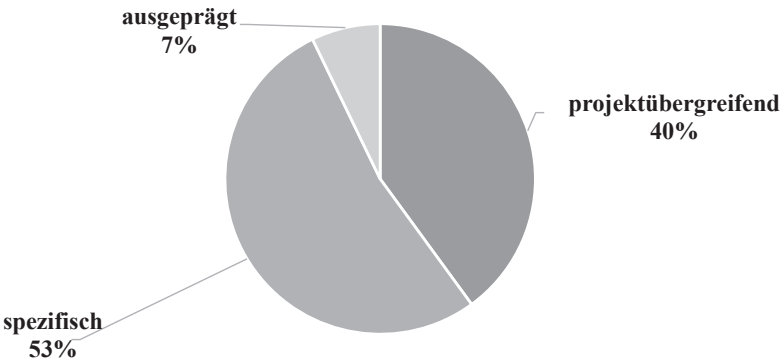


Abbildung 4-6: Zusammensetzung der Gruppen nach automatisiertem Vergleich

Jede Zeile setzt sich aus der möglichen Zuordnung von zwei Anforderungen zusammen (Tabelle 4-9) und wird im Folgenden als ein Match bezeichnet.

Gruppe	Dokument A	Dokument B	Zuordnung	Gruppierung
projektübergreifend	**3.6.6.1 Allgemeine Anforderungen:** Die Vorspannung der Radlagereinheit muss mit einer Vernietung realisiert werden.	**Vorgaben für die Auslegung:** Die Vorspannung der Radlagereinheit ist mit einer Vernietung zu realisieren.	richtig	falsch

Tabelle 4-9: Überprüfung der Eingruppierung

Im Beispiel in Tabelle 4-9 wurde das Match als projektübergreifend eingruppiert. Die inhaltliche, semantische Zuordnung beider Anforderungen aus der Überprüfung in Kapitel 4.2 ist korrekt. Allerdings weist die Anforderung das Ausprägungsmerkmal Vernietung auf, welches nicht erkannt wurde. Die richtige Gruppe in diesem Fall wäre

ausgeprägt. Die Liste der definierten Ausprägungsmerkmale wird anschließend hinsichtlich dieser Ausprägung geprüft und ggf. um diese erweitert.

Um die Güte der Klassifizierung nach dieser automatisierten Methode zu überprüfen, werden so insgesamt 13.546 Matches manuell auf ihre richtige Gruppe kontrolliert. Dabei wird geprüft, ob ein Ausprägungsmerkmal erkannt wird und ob die semantische Zuordnung korrekt war. Bei den Matches erfolgt je nach Eingruppierung die Überprüfung anhand der folgenden Fragen:

Automatische Klassifizierung als spezifisch:

1. Weist die Anforderung ein Ausprägungsmerkmal wie z.B. eine Zahl, Farbe oder z.B. eine Verbindungsart auf?
    - Ja →Klassifizierung falsch.
    - Nein → weiter mit 2. Frage.
2. Ist die semantische Zuordnung falsch? Konnte eine inhaltlich ähnliche Anforderungen nicht korrekt zugeordnet werden?
    - Ja → Klassifizierung falsch.
    - Nein → Klassifizierung richtig.

Automatische Klassifizierung als projektübergreifend:

1. Weist die Anforderung ein Ausprägungsmerkmal wie z.B. eine Zahl, Farbe oder z.B. eine Verbindungsart auf?
    - Ja → Klassifizierung falsch.
    - Nein → weiter mit 2. Frage.
2. Ist die semantische Zuordnung falsch? Ist die zugeordnete Anforderung nicht inhaltlich ähnlich?
    - Ja → Klassifizierung falsch.
    - Nein → Klassifizierung richtig.

Automatische Klassifizierung als ausgeprägt:

1. Kann das identifizierte Merkmal keine unterschiedlichen Ausprägungen aufweisen? Ist das identifizierte Ausprägungsmerkmal eine Dokument-Nr., CAD-Modell-Nr. oder ähnliches?
    - Ja → Klassifizierung falsch.
    - Nein → Klassifizierung richtig, ggf. in Liste der Ausprägungsmerkmale aufnehmen.

Um die zweite Forschungsfrage (2.2) zu beantworten, wurden 8 Experteninterviews mit den jeweiligen komponentenverantwortlichen Entwicklern der betrachteten mechanischen Komponenten durchgeführt (Tabelle 4-14). In den Interviews wurde den Entwicklern die Auswertungen der Anforderungsvergleiche mit ihren klassifizierten Anforderungen vorgelegt und besprochen. Anschließend sollten die Entwickler mit ja oder nein beantworten, für welchen der folgenden Anwendungsfälle sie die Auswertung in Zukunft nutzen werden:

1. Anzahl Anforderungen/ Umfang Lastenhefte reduzieren
2. Vereinheitlichung der Formulierungen
3. Erzeugung einer Anforderungsbibliothek zur projektspezifischen Lastenheftkonfiguration
4. Kosteneinsparung durch Anforderungsentfall (konstruktiv)
5. Kosteneinsparung durch Anforderungsentfall (Produktion)
6. Variantenreduzierung in der Komponente durch Harmonisierung von Anforderungen
7. Keine Optimierung/Anwendung

Des Weiteren hatten die Entwickler die Möglichkeit, in einem Freitextfeld noch eigene Anwendungsfälle für den Anforderungsvergleich zu benennen.

4.3.2 Ergebnisse

(1.1) Können Anforderungen basierend auf einem automatisierten Vergleich in die Gruppen projektübergreifend, ausgeprägt oder spezifisch klassifiziert werden?

Insgesamt sind 78,3% aller kontrollierten Matches der korrekten Gruppe zugeordnet worden, 21,7% der Zuordnungen sind falsch. Je nach Anforderungsgruppe ergibt sich ein anderer Anteil der falschen Klassifikationen. Folgende Abbildung 4-5 zeigt deren Anteile. Unter den ausgeprägten Anforderungen finden sich nur 0,15% falsch klassifizierter Matches, sodass dieser Anteil im Größenverhältnis der Abbildung nicht auftaucht.

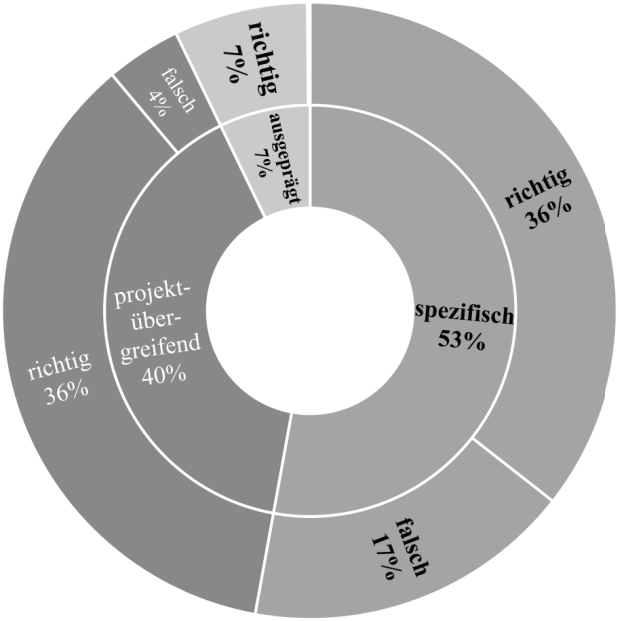


Abbildung 4-7: Auswertung der Klassifikation je Anforderungsgruppe

Bei der Auswertung der falsch klassifizierten Anforderungen fällt der hohe Anteil falscher Klassifizierungen innerhalb der spezifischen Anforderungen auf. Um die Ursachen für falsche Eingruppierungen zu identifizieren, werden alle Matches mit falscher Eingruppierung untersucht: Bei einem Anteil von 38,9% aller falschen Klassifizierungen wurde eine bereits definierte Ausprägung nicht erkannt. Ursachen hierfür waren,

dass die Regeln zur Erkennung einer Ausprägung zu eng definiert waren. Z.B. liegt in manchen Fällen zwischen Zahl und Einheit ein Leerzeichen, mal nicht. Des Weiteren wurde die Zahl, insbesondere wenn danach die Einheiten Zyklen, Ereignisse, Mal oder Schrauben folgte, ausgeschrieben. Zusätzlich wurde die Ausprägung der Streckgrenze oder cpk nicht erkannt.

Die zweitgrößte Ursache für falsche Klassifikationen ist mit 24% eine zuvor falsche semantische Zuordnung. Wurden zwei inhaltlich unterschiedliche Anforderungen einander zugeordnet, ergab die Klassifikation automatisch ein projektübergreifend. Genauso wurden aufgrund fehlender semantischer Zuordnung zweier Anforderungen diese fälschlicherweise als spezifisch klassifiziert. Hierbei handelt es sich um Folgefehler des semantischen Vergleichs und nicht der Klassifikation an sich.

Komponenten	∑ Projekt-übergreifend	∑ spezifisch	∑ ausgeprägt	∑ Match richtig	∑ Match falsch	∑ Gruppierung richtig	∑ Gruppierung falsch
Hinterachsträger	1648	3600	281	4596	555	4033	1496
Lenksäule	147	212	17	339	18	309	65
Lenkspindel	271	69	57	393	0	379	18
Radlager	774	835	79	1443	87	1331	357
Schwenklager	1055	1229	137	2021	154	1968	450
Stoßdämpfer	216	44	38	286	3	274	24
Stützlager	1303	1171	363	2509	150	2308	529
Gesamtergebnis	**5.414**	**7.160**	**972**	**11.587**	**967**	**10.602**	**2.939**

Tabelle 4-10: Auswertung der Eingruppierungen der Lastenhefte

Ein weiterer Folgefehler entsteht aus dem Preprocessing und der fehlerhaften Zerlegung des Fließtextes. Dieser Folgefehler ist die Ursache von 20,6% aller falschen Klassifizierungen. Mit 6% stellen auch neue Ausprägungsmerkmale Fehlerursachen in der Klassifikation dar, die bislang in der regelbasierten Erkennung anhand der Ausprägungsliste noch nicht definiert waren. Diese Liste wird basierend auf diesen Fehlern um Ausprägungen wie Schraube, Verbindungsart, Länder, Klasse, Note oder Freiheitsgrad erweitert. Die verbleibenden falschen Klassifizierungen weisen zu spezifische, einzelne Ursachen auf.

Der hohe Anteil falscher Klassifizierungen in den projektspezifischen Anforderungen (45%) im Gegensatz zu projektübergreifenden (14%) und ausgeprägten (2%) Anforderungen ergibt sich aus dem Zusammentreffen von mehreren fehlerverursachenden Faktoren wie Zerlegungsfehler, falscher semantischer Zuordnung und nicht erkannter Ausprägung. Hieraus lässt sich erkennen, dass sowohl die Qualität des Preprocessing als auch die Qualität der semantischen Zuordnung einen starken Einfluss auf die Korrektheit der Eingruppierung aufweisen.

Werden nur die Matches betrachtet, die keinen Zerlegungsfehler aufweisen und korrekt semantisch zugeordnet wurden als Summe der TP und TN (Kapitel 4.2.2), sind 11,7% davon falsch klassifiziert. Basierend auf der Anwendung bei zwei verglichenen Lastenheften ist die Einordnung in die Gruppen projektübergreifend, ausgeprägt und spezifisch mit einer Genauigkeit von 88,3% möglich. Durch eine Erweiterung der Liste für Ausprägungsmerkmale kann diese noch weiter verbessert werden. Werden mehr als zwei Lastenhefte miteinander verglichen, so müssen die Regeln zur Eingruppierung der Anforderungen erweitert werden, ab wie vielen Vorkommen einer Anforderung in den unterschiedlichen Lastenheften sie als projektübergreifend gilt.

(1.2) Ist durch die Klassifizierung eine produktspezifische Konfiguration eines Lastenheftes aus einer Anforderungsbibliothek einer Komponente möglich?
Zur Beantwortung der Frage werden 149 Matches mit den jeweils komponentenverantwortlichen Entwicklern besprochen. In den 8 Expertengesprächen sagen 88% der befragten Entwickler (Tabelle 4-11), dass sich die gezeigte Klassifizierung der Anforderungen dazu eignet, eine Anforderungsbibliothek zur Wiederverwendung von Anforderungen zu erzeugen, aus dem wiederum projektspezifische Lastenhefte konfiguriert werden.

Anwendungsfälle	Ja	Nein	Ja – relativ
Anzahl Anforderungen/ Umfang Lastenheft reduzieren.	5	3	63%
Vereinheitlichung der Formulierungen.	8	0	100%
Erzeugung einer Anforderungsbibliothek zur projektspezifischen Lastenheftkonfiguration.	7	1	88%
Kosteneinsparung durch Anforderungsentfall (konstruktiv).	1	6	14%
Kosteneinsparung durch Anforderungsentfall (Produktion).	2	6	25%
Variantenreduzierung in der Komponente durch Standardisierung/Harmonisierung von Anforderungen.	0	8	0%
Kontrollfrage: Keine Optimierung möglich.	0	8	0%

Tabelle 4-11: Auswertung Experteninterview zu Anwendung und Chancen des Anforderungsvergleichs

Am meisten stimmen die Entwickler dem Nutzen zur Vereinheitlichung der Formulierungen zu (100%). Dies geht einher mit einer einheitlichen, strukturierten Wiederverwendung in einer Anforderungsbibliothek. Zusätzlich sehen 63% der Entwickler eine Chance darin, mithilfe des Anforderungsvergleichs die Anzahl ihrer Anforderungen zu reduzieren. Dass man allerdings durch den Anforderungsvergleich Anforderungen entfallen lassen und dadurch tatsächlich Kosten in der Konstruktion oder Produktion von einzelnen Komponenten einsparen könnte, sehen 75% der Entwickler nicht. Ebenso sieht niemand eine Chance zur Reduzierung von Komponentenvarianten durch den Anforderungsvergleich und den Aufbau einer Anforderungsbibliothek. Dies wird insbesondere noch in Kapitel 4.4 diskutiert.

In den Gesprächen zum Aufbau einer Anforderungsbibliothek anhand der Vergleiche fällt auf, dass bislang das chronologische Vorkommen der Anforderungen ebenso wenig betrachtet wird wie das Vorkommen in unterschiedlichen Produktlinien bzw. technischen Konzepten. So werden Anforderungen als projektübergreifend zur Wiederverwendung in zukünftigen Lastenheften vorgeschlagen, die zwar in der Vergangenheit in mehreren Lastenheften festgelegt wurden, aber nach einer Bereinigung oder nach einem Technologiesprung entfallen sind. Da der Anteil der älteren Lastenhefte mit dieser Anforderung aber größer als 50% ist, wird diese als projektübergreifend zur Wiederverwendung eingeordnet. Für diesen Fall könnte z.B. ein Archiv-Attribut für Anforderungen in einer Anforderungsbibliothek zur Wiederverwendung manuell vergeben werden, sodass die Anforderung zwar als projektübergreifend eingeordnet, aber durch das Attribut Archiv nicht länger zur Wiederverwendung vorgeschlagen wird. Die Berücksichtigung des Vorkommens in unterschiedlichen Produktlinien und unterschiedlichen technischen Konzepten ist schwieriger. Hierbei könnten Stichwort-Tags für die Anforderungen eines Lastenheftes helfen. In jedem der betrachteten Komponentenlastenheften wird im Lastenheft-Kapitel 2: Entwicklungsumfang beschrieben, in welchem Achskonzept die Komponente verbaut wird (Fünf-Lenker-Hinterachse, Zwei-Gelenk-Federbein-Vorderachse, Doppel-Querlenker-Vorderachse etc.) und ob es sich um eine angetriebene oder nicht-angetriebene Achse handelt. Hierdurch können zwar keine Technologiesprünge direkt identifiziert werden, aber die Lastenhefte und ihre Anforderungen könnten in Technologiegruppen sortiert werden, sodass man z.B. projektübergreifende Anforderungen für einen jeweiligen Achstyp erhält. Solche Stichwort-Tags können durch Annotationen mittels einer der vielen verfügbaren Annotationstools automatisiert erzeugt werden. Dies wird bereits in vielen Systemen mit Stichwort-Verlinkung wie z.B. in Wikipedia angewendet.

Darüber hinaus wurden von den Entwicklern noch weitere Anwendungsgebiete der Ergebnisse aufgrund des automatisierten Anforderungsvergleichs aufgezeigt. So könne durch die Methode eine Qualitätsanalyse der Anforderungsformulierung durchgeführt werden, Duplikate eine Anforderung innerhalb eines Lastenheftes erkannt und durch die Vereinheitlichung der Formulierung durch die Verwendung als Anforderungsbibliothek der Aufwand zur Abstimmung mit den Lieferanten reduziert werden. Durch den automatisierten Vergleich könnten auch Bauraumanforderungen zur geometrischen Integration in ein Teilsystem ausgewertet sowie komponentenübergreifende Prüffälle und ihre Bedingungen, die in den Lastenheften beschrieben werden, verglichen werden. Durch die Möglichkeiten des semantischen Vergleichs wurde auch der Bedarf zum Abgleich von gewonnenen Erkenntnissen aus Vorgängerprojekten, s.g. Lessons Learned, mit den im Lastenheft beschriebenen Anforderungen genannt.

Zusammenfassung

Insgesamt ist die Güte der automatisierten Klassifizierung der Anforderungen in hohem Maß davon abhängig, wie gut das Preprocessing und die semantische Zuordnung inhaltlich ähnlicher Anforderungen im vorangegangenen Schritt ist. Betrachtet man nur die Beispiele mit korrektem Preprocessing und richtiger semantischer Zuordnung, werden 88,3% der Anforderung richtig klassifiziert. Trotz korrekter semantischer Zuordnung und richtiger Klassifizierung können einander zugeordnete Anforderungen dennoch einen inhaltlichen Unterschied aufweisen. Diese Beobachtung wird anhand von Zahlen und Beispielen in Kapitel 4.4.1 unter Abschnitt (e) erläutert. Um Aspekte wie Chronologie oder unterschiedliche technische Konzepte in einer Anforderungsbibliothek zur Wiederverwendung der Anforderungen zu berücksichtigen, können durch eine Erweiterung der Methode mittels Annotations-Verfahren Stichwort-Tags bzw. weitere Attribute zur Charakterisierung einer Anforderung vergeben werden. Bei der Konfiguration eines projektspezifischen Lastenheftes aus einer Anforderungsbibliothek könnten anhand dieser Attribute geeignete Anforderungen für die Konfiguration gefiltert werden. Allerdings ist der hohe Anteil an spezifischen Anforderungen von 45% (nur Summe der korrekten Klassifizierungen betrachtet) eine große Herausforderung für die Konfiguration, da diese trotz möglicher Zusatzattribute bislang einzeln und manuell auf ihre Wiederverwendung geprüft werden müssten. Hingegen hilft nach Aussage der Entwickler aus den Expertengesprächen insbesondere die Eingruppierung der ausgeprägten Anforderungen sowie die farbliche Hervorhebung der Ausprägungen, Schwankungen in den Werten zu identifizieren und zu hinterfragen. So könnten z.B. schnell Auswertungen von Bauraum- und Freigangs-Anforderungen für die geometrische Gestaltung erzeugt werden.

4.4 Anforderungen als Variantenverursacher

Durch den automatisierten Vergleich der Anforderungen verschiedener Komponentenlastenhefte wird eine Klassifikation dieser in die Gruppen projektübergreifend, ausgeprägt und spezifisch möglich. Gemäß der in Kapitel 3.1 beschriebenen Hypothese können Anforderungen der Gruppen ausgeprägt und spezifisch potenziell zu Varianten der Komponenten führen, da sie durch unterschiedliche Merkmale umgesetzt werden. Um diese Theorie zu überprüfen, sollen Anforderungen der Gruppen ausgeprägt und spezifisch in diesem Kapitel auf ihre potenziell variantenverursachende Wirkung untersucht werden und dadurch die letzten beiden Forschungsfragen beantwortet werden:

(2.1) Sind spezifische oder ausgeprägte Anforderungen variantenverursachend?

(2.2) Können durch einen automatisierten Anforderungsvergleich mit anschließender Klassifizierung variantenverursachende Anforderungen identifiziert werden?

4.4.1 Auswahl geeigneter Fallbeispiele

Um Einflüsse durch eine fehlerhafte Textzerlegung, falscher semantischer Zuordnung oder Klassifizierung auszuschließen, werden für diesen Schritt ausschließlich Anforderungen betrachtet, die die folgenden Kriterien erfüllen:

(a) Es handelt sich um eine Anforderung und nicht um eine Information, Verweis oder eine Vereinbarung zur Zusammenarbeit zwischen Auftraggeber und Auftragnehmer.
→ Objekttyp: Anforderung

(b) Die Zuordnung im automatisierten Vergleich ist für die betrachtete Anforderung korrekt.
→ semantische Zuordnung: richtig

(c) Die Klassifizierung der Anforderung in eine der drei Gruppen ist korrekt.
→ Klassifizierung: richtig

(d) Die Anforderung wurde korrekt in die Gruppe ausgeprägt oder spezifisch klassifiziert.
→ Gruppe: Spezifisch oder ausgeprägt

(e) Die Anforderung muss einen inhaltlichen Unterschied zum verglichenen Lastenheft aufweisen.
→ Inhaltlicher Unterschied: ja

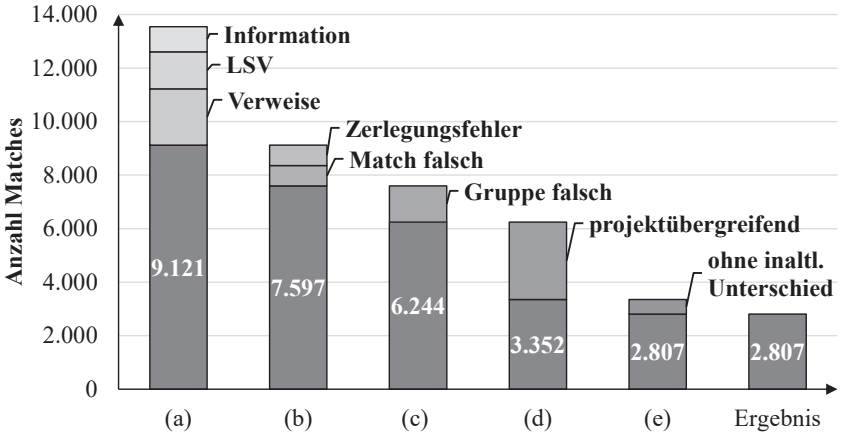


Abbildung 4-8: Diagramm zur Auswahl geeigneter Validierungsbeispiele

Diese Kriterien werden nun detailliert erläutert:

(a) Es handelt sich um eine Anforderung und nicht um eine Information, Verweis oder eine Vereinbarung zur Zusammenarbeit zwischen Auftraggeber und Auftragnehmer.

Dieses Kriterium ist erforderlich, da bei der Überprüfung der 13.546 Matches aufgefallen ist, dass viele verglichene Textfragmente in den Lastenheften keine technische Anforderung (s. Kapitel 2.1.2 Anforderungsarten) an die Komponente darstellen. Stattdessen können neben technischen Anforderungen an die Komponente folgende drei Objekttypen in den Komponentenlastenheften beschrieben werden:

Information: Eine Information dient zum besseren Verständnis des Kontextes einer Anforderung. Sie kann die Eigenschaften des Produktes beschreiben, benachbarte Komponenten benennen oder den Bauraum der betrachteten Komponente definieren. Sie stellt aber keine Anforderung dar, die durch den Auftragnehmer des Lastenheftes zu erfüllen ist.

„Das im CAD-Modell XY dargestellte Ausrichtsystem ist vorbehaltlich und entspricht dem aktuellen Konzeptstand."

„Die Erfassung der HC-Emissionen erfolgt im Rahmen der für die Typzulassung erforderlichen Gesamtfahrzeug-Messungen."

Verweis: Durch die bereits beschriebene Vielzahl an Begleitdokumenten bei der Vergabe einer Komponente stehen im Komponentenlastenheft Verweise auf die jeweils mitgeltenden Unterlagen. Des Weiteren gibt es Verweise auf andere Abschnitte des Lastenheftes oder auf Abbildungen und Tabellen.

„Die auftretenden Lasten sind im Bauteil-Datenblatt definiert."

„Die Deformationscharakteristik ist Abschnitt 5.3. zu entnehmen."

Leistungsvereinbarung: Eine Leistungsvereinbarung ähnelt einer Projektanforderung. Sie beschreibt eine zu erbringende Leistung des Auftragnehmers, die nicht auf die Komponente bezogen ist. So können diese Vereinbarungen zur Zusammenarbeit darstellen, wann der Auftraggeber zu informieren ist oder welche Dokumentationen zu erstellen sind.

„Abweichungen dazu sind nach detaillierter Prüfung möglich, müssen aber vom Auftraggeber genehmigt werden."

„Die genaue Position der Aufnahmepunkte ist im Laufe der Entwicklung mit dem Auftraggeber abzustimmen."

Es kommt auch vor, dass sich zwei Objekttypen in einem Textfragment vermischen. So ist folgendes Beispiel als Verweis oder als Leistungsvereinbarung einzustufen.

„Daraus abzuleitende Einpresskräfte für das Lager und Gelenk sind in einer Prozessvorschrift zu dokumentieren und gelten dann in Verbindung mit der Korrelation zur Auspresskraft als relevante Merkmale nach [ISO TS 16949]."

Die Kategorisierung der Textfragmente in den Lastenheften in die Objekttypen Anforderung, Information, Verweis und Leistungsvereinbarung zeigt über die betrachteten Matches die in Tabelle 4-12 dargestellte Verteilung.

% Anforderungen	%Verweise	%Leistungs-vereinbarung	% Informationen
67,3%	15,5%	10,3%	6,9%

Tabelle 4-12: Auswertung der Anteile Objekttypen je Komponente

Die Auswertung der Objekttypen in Tabelle 4-12 zeigt, dass der Anteil der technischen Anforderungen bei 67,3% liegt. Die zweitgrößte Gruppe der Objekttypen in den betrachteten Komponentenlastenheften sind Verweise mit einem Anteil von 15,5%.

Diese hohen Anteile der Verweise lässt sich durch die bereits erwähnte hohe Anzahl an verschiedenen Begleitdokumenten in der Vergabe einer Komponente erklären. Danach folgen Leistungsvereinbarungen mit einem Anteil von 10,3%. Die kleinste Gruppe sind Informationen mit 6,9%. Die Klassifizierung der Matches in diese vier Objekttypen führt dazu, dass nur 67,3% der Matches für die Überprüfung als Variantenverursacher in Frage kommen.

(b) Die Zuordnung im automatisierten Vergleich ist für die betrachtete Anforderung korrekt.
Es sollen nur die im semantischen Vergleich korrekt zugeordneten Anforderungen betrachtet werden. Die Anzahl der korrekt semantisch zugeordneten Anforderungen ergibt sich aus der Anzahl korrekter Matches aus Kapitel 4.2.2.

(c) Die Klassifizierung der Anforderung in eine der drei Gruppen ist korrekt.
Um nicht fälschlicherweise als spezifisch oder ausgeprägt eingestufte Anforderungen hinsichtlich ihrer variantenverursachenden Wirkung zu untersuchen, werden nur korrekte Klassifizierungen betrachtet. Die Anzahl der hier relevanten Matches ergibt sich aus der Auswertung von Kapitel 4.3.2.

(d) Die Anforderung wurde korrekt in die Gruppe ausgeprägt oder spezifisch klassifiziert.
Nach Bereinigung der Anforderungen mit falscher Klassifizierung werden nun gemäß der grundlegenden Idee nur die als ausgeprägt oder projektspezifisch auf deren variantenverursachenden Wirkung untersucht. Deren Anzahl ergibt sich ebenfalls anhand der Auswertung in Kapitel 4.3.2.

(e) Die Anforderung muss einen inhaltlichen Unterschied zum verglichenen Lastenheft aufweisen.
Des Weiteren müssen basierend auf der Theorie, dass Unterschiede in den Anforderungen zu Unterschieden in den Merkmalen und somit Varianten der Komponente führen, die Validierungsbeispiele einen inhaltlichen Unterschied aufweisen. Im Fall der spezifischen Anforderungen ist dies sichergestellt, da sie nur als spezifisch eingestuft werden, wenn sie in dem verglichenen Dokument nicht gefunden werden und somit durch ihr Vorkommen einen Unterschied aufweisen. Im Fall der ausgeprägten Anforderungen kann es jedoch sein, dass die Anforderungen zwar korrekt als ausgeprägt gruppiert wurden, da sie z.B. einen Zahlenwert aufweisen. Allerdings ist die Ausprägung in beiden verglichenen Dokumenten gleich. In diesem Fall liegt somit kein inhaltlicher Unterschied einer ausgeprägten Anforderung vor.

Abbildung 4-9 zeigt die Anzahl der identifizierten Anforderungen mit inhaltlichem Unterschied je Gruppe.

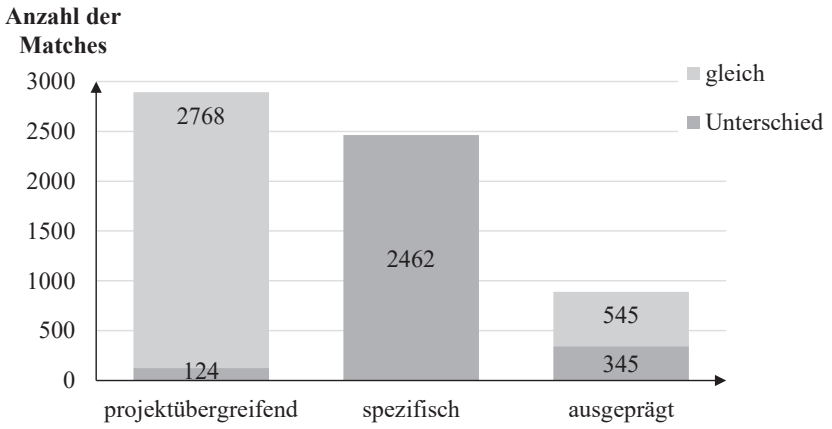


Abbildung 4-9: Anzahl der Matches je Gruppe mit inhaltlichem Unterschied

An dieser Stelle soll auch nochmal anhand von Beispielen erläutert werden, dass die Zuordnung anhand des Vergleichs auf einer semantischen Ähnlichkeit beruht und nicht bedeutet, dass der Inhalt identisch ist. Gemäß der Definition einer projektübergreifenden Anforderung als inhaltlich ähnliche Anforderung über verschiedene Projekte sollte die Gesamtzahl der inhaltlichen Unterschiede hier bei null liegen. Dennoch

Nr.	Dokument A	Dokument B
1	**3.6.6.3 Schnittstelle verschraubte Lenker zum Achsträger:** Für die Einstellmöglichkeit für Vorspur/Sturz muss ein Langloch mit Exzenterabstützung umgesetzt sein.	**Schnittstelle Achsträger zu den Spurlenkern:** Hierfür ist ein Langloch mit Exzenterabstützung zu realisieren. --- **Schnittstelle Achsträger zu den Sturzlenkern:** Hierfür ist ein Langloch mit Exzenterabstützung zu realisieren.
2	**4.2.2.2 Lebensdauerauslegung Einstufenversuche:** Als Zielgröße sind die im Bauteil-Datenblatt angegebenen Lastwechselzahlen bis Anriss (Schädigung 1) zu erreichen.	**Lebensdauerauslegung Einstufenversuche:** Als Zielgröße sind die im Bauteil-Datenblatt angegebenen Lastwechselzahlen bis Anriss zu erreichen.

Tabelle 4-13: projektübergreifende Anforderungen mit inhaltlichem Unterschied

wird bei 4,3% der projektübergreifenden Anforderungen ein Unterschied festgestellt. Um diesen Anteil projektübergreifender Anforderungen mit inhaltlichem Unterschied zu hinterfragen, werden in Tabelle 4-13 Beispiele für diesen Fall dargestellt.

In Beispiel Nr. 1 ist die Anforderung aus Dokument A korrekt den beiden Anforderungen aus Dokument B als inhaltlich ähnlich zugeordnet. Durch das Semantic Merging wurden in Dokument B zwei Anforderungen zusammengelegt. Dieses Match wurde korrekt als projektübergreifend eingestuft, da in beiden Dokumenten eine inhaltlich ähnliche Anforderung vorliegt. Der inhaltliche Unterschied liegt hier allerdings darin, dass in Dokument B explizit ein Langloch für die Schnittstelle Spurlenker als auch Sturzlenker gefordert wird. In Dokument A ist lediglich die Rede von Lenkern. Somit sind die Anforderungen aus Dokument B detaillierter und spezifischer, aber dennoch inhaltlich ähnlich und zugehörig zu der aus Dokument A. Ein weiteres Beispiel für korrekt als übergreifend klassifizierte Anforderungen mit inhaltlichem Unterschied ist Nr. 2: Die Anforderungen sind in beiden Dokumenten auch von der Wortwahl identisch. Allerdings gibt es in Dokument A die Ergänzung „Schädigung 1“, die es in Dokument B nicht gibt.

Insgesamt wurde durch die in diesem Abschnitt beschriebenen Kriterien aus 13.546 Matches 2.807 Anforderungen für eine Überprüfung der variantenverursachenden Wirkung ausgewählt. Diese Anforderungen erfüllen alle der oben beschriebenen Auswahlkriterien.

4.4.2 Vorgehen

Um eine potenziell variantenverursachende Wirkung dieser ausgewählten 2.807 Anforderungen zu überprüfen, werden in Expertengesprächen mit den zuständigen Komponentenentwicklern bekannte Variantentreiber der Komponente erhoben und anschließend die zuvor ausgewählten Anforderungen in zu überprüfende Merkmale übersetzt. Diese Merkmale sollen anschließend an der realisierten Komponente auf Unterschiede geprüft werden. Für die folgenden vier Komponenten werden die jeweiligen Komponentenentwickler befragt und die ausgewählten Anforderungen als Match aus dem Lastenheftvergleich besprochen (Tabelle 4-14).

Komponente	Anzahl Komponentenentwickler	Anzahl besprochener Anforderungen
Radlager	1	22
Schwenklager	2	36
Stützlager	3	18
Hinterachsträger	2	73
Summe	**8**	**149**

Tabelle 4-14: Anzahl befragter Experten und überprüfter Anforderungen

Die Expertengespräche sind in zwei Phasen gegliedert: Zunächst werden die Entwickler zu bereits bekannten Ursachen für Varianten ihrer Komponente befragt:

Phase 1:

1. Wodurch werden neue Varianten der Komponente verursacht?
2. Wann (in welchem Fall) wird eine neue Variante der Komponente erzeugt?
3. Durch welche Merkmale unterscheiden sich die Varianten der Komponente?

Dabei werden einerseits die bekannten Ursachen ermittelt. Andererseits wird durch die dritte Frage ein Transfer in die daraus resultierenden Unterschiede bei den Komponentenvarianten geleistet.

In der zweiten Phase werden konkret die ausgewählten Anforderungen besprochen. Hierzu wird für jede Anforderung gefragt, durch welche Merkmale an der Komponente die Anforderung erfüllt werden kann. Konnten Merkmale definiert werden, wird anschließend gefragt, zu welchen unterschiedlichen Merkmalsausprägungen der Unterschied in den Anforderungen führen kann.

Phase 2: Je Anforderung

a. Durch welche Merkmale wird diese Anforderung an der Komponente erfüllt?
b. Zu welchen unterschiedlichen Merkmalsausprägungen kann der Unterschied in der Anforderung führen?

Anschließend werden die aus den Anforderungen abgeleiteten Merkmale auf ihren Unterschied an der Komponente geprüft. Parallel wird zu den Komponenten im Produktdatenmanagement erhoben, ob die Komponenten als Gleichteil (Kapitel 2.2.1) oder Neuteil angelaufen sind.

4.4.3 Ergebnisse

Phase 1:

In den Expertengesprächen werden für die betrachteten Komponenten Hinterachsträger, Radlager, Schwenklager und Stützlager auf die erste Frage die in Tabelle 4-15 aufgelisteten Variantentreiber genannt.

Radlager	Schwenklager	Stützlager	Hinterachsträger
- angetriebene oder nicht angetriebene Achse - Achslast - Rdyn = Dynamischer Rollradius - Felgenmaterial - Radgröße	- Anpassbauteil für benachbarte Baukastenbauteile - angetriebene oder nicht angetriebene Achse - Zugstrebe oder Querlenker - Achskonzept - Sturzkorrektur - Radgröße - Kosten (Business Case) - Vergabeszenario / Märkte - Gewichtsziel - Kinematik - Achslast - SA Radbeschleunigungssensor	- Gesamtfahrzeug -steifigkeit - Bauraum Gepäckraum Durchladebreite - Funktion Fahrdynamik - Montagekonzept Hochzeit - Links/Rechts Variante - Dämpfungskonzept	- Achskonzept - Schnittstellen zur Karosserie - Kinematik - Lager Antriebsstrang - Antriebsart ICE/BEV/PHEV - Gesamtfahrzeugarchitektur - Bauraum - Betriebsfestigkeit - Kosten (Business Case) - Gummilager- und Tilgervarianten - Crash: ertragbare Lasten und Knicklasten

Tabelle 4-15: Variantentreiber ausgewählter Fahrdynamikkomponenten

Dabei haben die Experten die aus den Variantentreibern resultierenden Merkmale wie im Folgenden erläutert. Das **Radlager**, welches die Radkräfte und die Momente aus Antrieb und Bremsen zwischen dem Rad und dem Radträger/Schwenklager überträgt, wird in Abhängigkeit von dem Antrieb der jeweiligen Achse mit einer Stirnverzahnung für den Abtrieb oder einem Deckel ausgestattet. Das Felgenmaterial des angebundenen Rades bestimmt die Flanschart des Radlagers von Leichtbau über Vollflansch oder Stahlradflansch. Die Radgröße wirkt sich auf den Bohrkreisdurchmesser des Radlagers aus. Ebenfalls Einfluss auf die Radlagervarianten hat laut Experten der dynamische Rollradius Rdyn, der den wirksamen (schlupffreie) Abrollradius des Rades beschreibt (Heißing et al. 2013, S. 398).

Das **Schwenklager** ist ein Radträger an einer gelenkten Achse und nimmt das Radlager auf (Heißing et al. 2013, S. 366). Die Varianten des Schwenklagers werden z.B. dadurch verursacht, dass das Schwenklager zur Anpassung der benachbarten Baukastenbauteile wie dem Radlager dient. Je nach Konzept zur Sturzkorrektur kann es auch eine Minus-, Plus- oder 0-Variante des Schwenklagers geben. Das Achskonzept (McPherson, Verbundlenker, Mehrlenker, Doppelquerlenkerachse etc.) beeinflusst durch die unterschiedlichen Komponenten die erforderlichen Anbindungspunkte am Schwenklager. So ergeben sich geometrische Varianten aus unterschiedlichen räumlichen Anbindungen von Spurstange, Zug-/Druckstrebe und Querlenker. Ebenfalls ergeben sich die geometrischen Varianten aus angetriebener oder nicht-angetriebener Achse (Heißing et al. 2013, S. 368ff.)

Das **Stützlager** verbindet das Federbein oder den Dämpfer mit der Karosserie. Dabei werden bei einem Dämpfer ausschließlich die Dämpferkräfte über das Stützlager übertragen, die dynamischen Fahrwerkskräfte und das Fahrzeuggewicht über die getrennte Tragfeder. Liegt ein Federbein vor, so werden neben der Dämpferkräfte auch Tragfederkräfte über eine Elastomerfeder in die Karosserie eingeleitet. Daher wird es je nach übertragenen Kräften Federbeinstützlager oder auch Dämpferlager genannt. (Heißing et al. 2013, S. 496) Die Ursachen für die Varianten des Stützlagers als Komponente des Teilsystems Vertikaldynamik sind sowohl die Funktion Fahrdynamik, welche sich auf die Wahl des Elastomers am Stützlager auswirkt, als auch das Federungskonzept Luftfederbein, konventionelle Federbein oder aufgelöstes Dämpfer- und Tragfedersystem. Darin eingeschlossen ist beispielsweise an der Hinterachse unter anderem auch die Anforderung Durchladebreite an den Gepäckraum, die darüber entscheidet, ob das Dämpfungskonzept als Federbein oder aufgelöster Dämpfer und separate Tragfeder ausgeführt wird. Ein weiterer Variantentreiber stellt das Montagekonzept der Hochzeit dar, bei der die Achsen einschließlich des Antriebsaggregats in die Karosserie verschraubt werden. Die Federbeine inklusive des Stützlagers werden entweder stehend in die Karosserie eingefahren oder an der Karosserie hängend mit der Achse verbunden. Hierfür werden Pins am Werkstückträger für den Aggregateeinbau zu Zentrierung und Führung als auch entsprechende Laschen mit Bohrungen am Federbeinhalter benötigt, die je nach Fahrzeug und Fügekonzept unterschiedlich in Größe, Anzahl und Position ausfallen können. Einfluss hierauf hat insbesondere auch die Karosserieform z.B. Limousine oder SUV und somit das Derivat, aber auch das jeweilige Achskonzept.

Der **Achsträger** nimmt Fahrwerkskomponenten wie z.B. Lenk-, Brems- und Federungssystem als auch je nach Antriebskonzept Komponenten des Antriebs auf. Er leitet deren äußeren Kräfte und Momente in die Karosserie ein und stütz die inneren Kräfte

ab. Daher stellt er ein Bauteil mit sehr vielen Schnittstellen dar. Diese Schnittstellen sind ein großer Faktor für die Varianz des Achsträgers. So werden die Schnittstellen zur Karosserie, die Lagerung des Antriebsstrangs als auch die Antriebsart, die Gummilager- und Tilgervarianten und das Achskonzept als Variantentreiber des Achsträgers genannt, die sich auf die geometrische Gestaltung des Achsträgers auswirken. Tilger bezeichnen hier zur Körperschalldämmung eingesetzte Massen, um Schwingungs- und Störgeräuschquellen durch Resonanzen im Fahrwerk zu vermindern (Heißing et al. 2013, S. 477ff.) Des Weiteren werden aus dem Gesamtsystem Fahrzeug Anforderungen an den Bauraum, die Fahrzeugarchitektur, die Betriebsfestigkeit und die im Crash ertragbare Lasten als Ursachen für die Varianz in der Gestaltung des Achsträgers durch unterschiedliche Geometrie, Materialdicken und -güte als auch das Herstellverfahren genannt.

In der Betrachtung der genannten Variantentreiber fällt auf, dass die Treiber keine Ziele oder Anforderungen an die Komponente selbst, sondern an das zugehörige System bzw. Teilsystem oder an das Produkt Fahrzeug als Ganzes darstellen. So sind die Variantentreiber angetriebene Achse, Achskonzept, Achslast, Kinematik etc. als Anforderungen an das System Fahrdynamik zu betrachten, die als solche nicht in einem Komponentenlastenheft zu finden sind. Ebenfalls sind die Treiber Business Case oder Vergabeszenario keine technischen Anforderungen an die Komponente, sondern wirtschaftliche Anforderungen an das gesamte Entwicklungsprojekt. Ebenfalls sind einige der genannten Variantentreiber die benachbarten Komponenten oder Systeme des Produktes. Das Felgenmaterial wird als Anforderung im Lastenheft des Rades festgelegt und nicht im Radlager-Lastenheft. Das Schwenklager ist als Anpassbauteil zwischen mehreren benachbarten Baukastenbauteilen in hohem Maß abhängig von den benachbarten Komponenten, sodass diese Anforderungen wiederum nicht im Komponentenlastenheft des Schwenklagers zu finden sind. Die Varianten des Achsträgers werden auch von den Anforderungen an das benachbarte System Antrieb verursacht. Aufgrund der unterschiedlichen Antriebsarten ICE, BEV und PHEV sowie unterschiedlichen Lagerungen des Antriebsstrang werden verschiedene Schnittstellen am Achsträger und dadurch wiederum unterschiedliche Varianten verursacht.

Beim Transfer der genannten Variantentreiber (Tabelle 4-15) in daraus beeinflusste Merkmale an den Komponenten werden bei den Komponenten Radlager, Schwenklager und Stützlager sehr spezifische Merkmale genannt. So führt unterschiedliches Felgenmaterial zu verschiedenen Flanscharten und die angetriebene oder nicht-angetriebene Achse führt zu unterschiedlichen Positionen der Lenkeranbindungen oder eine Stirnverzahnung für den Abtrieb. Ebenfalls können wirtschaftliche Anforderungen

wie der Business Case oder das Vergabeszenario in konkrete Merkmale (Guss- oder Schmiedeteil; unterschiedliche Sachnummern aufgrund unterschiedlicher Lieferanten) übersetzt werden. Die Auswirkung einiger Variantentreiber, insbesondere bei der Komponente Hinterachsträger, kann nur sehr allgemein als unterschiedliche Geometrie übersetzt werden, ohne diese auf ein bestimmtes Merkmal zu konkretisieren. Eine mögliche Erklärung hierfür liegt im Entwicklungsprozess des Achsträgers, der nicht als vorgegebenes CAD-Modell an den Lieferanten übergeben wird, sondern durch FEM-Simulationen unter Einfluss verschiedener Belastungs- und Beanspruchungsgrößen aus dem Produktdatenblatt beim Lieferanten seine endgültige Geometrie erhält.

(2.1) Sind spezifische oder ausgeprägte Anforderungen variantenverursachend?
Anschließend werden die ausgewählten Beispiele aus den Anforderungsvergleichen mit den Experten einzeln besprochen und diese Anforderungen in zu überprüfende Merkmale übersetzt. Insgesamt werden 142 Fallbeispiele aus verglichenen Anforderungen mit potenziell variantenverursachender Wirkung (s. Auswahl Kapitel 4.4.1) diskutiert. In der Diskussion wird festgestellt, dass nur ein Teil der Anforderungen Auswirkungen auf die Komponente hat, da sie in ein durch die Anforderung beeinflusstes Merkmal übersetzt werden können. Ein anderer Teil der Anforderungen mit inhaltlichem Unterschied aus dem Vergleich hat nach Aussage der Experten keine Auswirkungen in Sinne einer physikalischen Änderung eines Merkmals auf die Komponente. Die Begründung der Entwickler hierfür lässt sich in diese vier Fälle gliedern:

1. Die Anforderung ist zu allgemein formuliert und nicht auf die Komponente des jeweiligen Entwicklungsprojektes spezifiziert.
2. Die Anforderung wurde zum besseren Verständnis oder als Hilfestellung für den Lieferanten eingefügt.
3. Die Anforderung dient zur rechtlichen Absicherung z.B., dass die Gesetze und Vorschriften der definieren Absatzmärkte einzuhalten sind.
4. Der Objekttyp wurde fälschlicherweise als Anforderung definiert, ist aber laut Experten eher eine Information.

Im folgenden Diagramm wird gezeigt, wie sich die besprochenen Anforderungen hinsichtlich ihrer Auswirkung auf die Komponente zusammensetzen. Abbildung 4-10 zeigt, dass der Anteil der Anforderungen mit einem durch den Entwickler identifizierten Einfluss auf die Merkmale der Komponente in einem Bereich zwischen 17% und 41% in Abhängigkeit der Komponente liegen. Dabei wurden z.B. spezifische Anforderungen eines Entwicklungsprojektes mit elektrischem Antrieb mit Auswirkungen auf die Merkmale der Komponente korrekt als variantenverursachend identifiziert, da

das verglichene Komponentenlastenheft eine Komponente für ein Fahrzeug mit Verbrennungsmotor diese Schnittstellen entsprechend nicht aufweist.

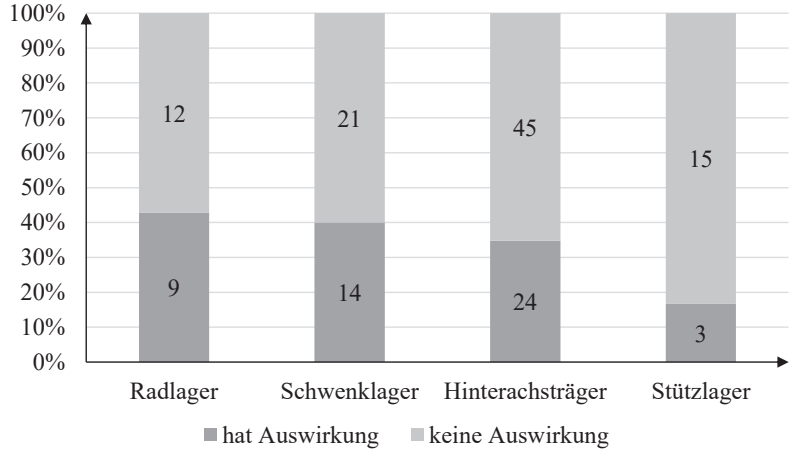


Abbildung 4-10: Anforderungsauswirkung auf ein physikalisches Merkmal

Weitere Beispiele für spezifischen Anforderungen mit Einfluss auf die Merkmale der Komponente sind bedingt durch das unterschiedliche Achskonzept der verglichenen Entwicklungsprojekte, sodass die Komponente entsprechende Anforderungen an die Schnittstelle bestimmter Lenker aufwies. Allerdings sind diese durch die Methode als variantenverursachend identifizierten Anforderungen als solche zu hinterfragen, denn in beiden genannten Fällen war nicht die spezifische Komponentenanforderung ursprünglich die Ursache für die Entstehung der Komponentenvariante, sondern die Entscheidung auf Produkt- oder Systemebene über die Antriebsform oder das Achskonzept.

Andere mit den Entwicklern besprochene Matches sind eindeutig als variantenverursachend zu bezeichnen, da sie in ein sehr konkretes spezifisches Merkmal der Komponente übersetzt werden konnten z.B. Anforderungen zu Korrosionsschutzmaßnahmen, die zu zusätzlichen oder anderen Positionen von Wasserabläufen in der Komponente geführt haben. Die Anzahl dieser Beispiele ist in den besprochenen Matches allerdings sehr gering (6 von 149 besprochenen Matches). Weitere als variantenverursachend identifizierten Anforderungen konnten nur in Einfluss auf generische Merkmale wie Material, Geometrie oder Wanddicke übersetzt werden. Eine konkret beeinflusste Geometriedimension oder Materialauswahl konnte nicht definiert werden.

Die Auswertung in Abbildung 4-10 zeigt gleichzeitig, dass zwischen 55%-83% der Anforderungen in kein Merkmal der Komponenten übersetzt werden konnten und somit in der Mehrheit der Fälle auch keine physikalischen Merkmale an den realisierten Komponentenvarianten geprüft werden können. Aus diesem Grund ist der Unterschied dieser Anforderungen in den verglichenen Lastenheften ohne Auswirkung auf die Komponente und ihren Varianten. Solche Anforderungen sind hinsichtlich ihrer Notwendigkeit in einem Komponentenlastenheft zu hinterfragen. Die Gründe für die mangelnde Übersetzbarkeit der Anforderungen in Merkmale sind vielfältig: zu allgemeine Formulierung der Anforderung ohne Konkretisierung für die Komponente, rechtliche Absicherung gegenüber Vertragspartner oder zum besseren Verständnis und Reduzierung von Rückfragen. Häufig wurden zu allgemeine Formulierungen aus unternehmensweiten Anforderungstemplates für das Komponentenlastenheft übernommen, die aber in ihrem jeweiligen Anwendungsfall nicht auf die Komponente zutreffen oder die nicht durch die Komponente allein, sondern nur durch das Teilsystem oder System erfüllt werden können. Dies weist auf ein Defizit in der Detaillierung und Spezifizierung der Anforderungen auf Systemebene zur Komponentenebene hin. Gleichzeitig zeigt es ein weiteres notwendiges Kennzeichen für Anforderungen in einer Anforderungsbibliothek: die Ebene der Anforderung als Produkt-, System-, Teilsystem- oder Komponentenanforderung.

Viele der in den Komponentenlastenheften genannten Anforderungen, die nicht in ein Merkmal übersetzt werden konnten, beziehen sich außerdem auf das Vorgehen zur Auslegung der Komponente, bestimmte Berechnungsvorschriften und Vorgaben zur Ausrichtung des Simulationsmodells in der Entwicklung der Komponente. Andere Anforderungen beschreiben Schritte im Herstellverfahren oder Bauteilprüfungen, die die Komponente bestehen muss und die durch den Auftragnehmer des Lastenheftes durchzuführen sind. Hierbei stellt sich die Frage, ob solche Anforderungen tatsächlich nicht in Merkmale der Komponente übersetzt werden können oder ob es an einem methodischen Vorgehen mangelt, das auch Anforderungen im Sinne von Verfahrensvorschriften in der Komponentenentwicklung in Merkmale übersetzt. Des Weiteren ist zu untersuchen, ob dies ein rein mechanisches Problem darstellt oder ob auch große Umfänge von Software-Lastenheften nicht in Merkmale der Software übersetzt werden können?

Die Antwort auf die Frage (2.1) ist basierend auf diesen Ergebnissen, dass spezifische und ausgeprägte Anforderungen variantenverursachend sein können, es aber nicht

zwingend sind. Aus der Gesamtheit aller ausgeprägten und spezifischen Anforderungen ist weniger als die Hälfte tatsächlich variantenverursachend bzw. kann überhaupt in ein Merkmal der Komponente übersetzt werden.

(2.2) Können durch einen automatisierten Anforderungsvergleich mit anschließender Klassifizierung variantenverursachende Anforderungen identifiziert werden?

Der automatisierte Anforderungsvergleich gibt durch die Auswertung Transparenz über die Unterschiede der Anforderungen. Zum einen unterscheiden sich die als spezifisch klassifizierten Anforderungen in ihrem Vorkommen, da sie nur in vereinzelten Lastenheften zutreffen. Zum anderen unterscheiden sich ausgeprägte Anforderungen hinsichtlich ihres Ausprägungsmerkmals. Die zugrundeliegende Idee dieser Methode ist, dass Unterschiede in den Anforderungen auch Unterschiede in den Merkmalen der Komponente verursachen. Daher werden in dieser Methode die ausgeprägten und spezifischen Anforderungen als variantenverursachende Anforderungen betrachtet und können mit dieser Klassifikation mit Hilfe der vorgestellten Methode identifiziert werden. Die Anwendung der vorgestellten Methode erfolgt anhand von Anforderungen in Komponentenlastenheften. Bereits im vorherigen Absatz wurde dargelegt, dass komponentenabhängig bis zu 83% der als ausgeprägt oder spezifisch klassifizierten Anforderungen keinen Einfluss auf ein Merkmal der Komponente haben. Stattdessen wurden in den Experteninterviews zahlreiche Variantentreiber auf Teilsystem-, System- oder Produktebene genannt. Es kann somit nur ein geringer Zusammenhang zwischen den Unterschieden in den Komponentenanforderungen und den Komponentenvarianten aufgezeigt werden.

Aufgrund dieses Ergebnisses der Anwendung und der Expertengespräche im spezifischen Umfeld mechanischer Fahrwerkskomponenten, eignet sich die Methode bedingt, um variantenverursachende Anforderungen in Komponentenlastenheften mechanischer Komponenten zu identifizieren. Die Methode liefert die Transparenz über die Unterschiede in den Anforderungen sowie eine Gruppierung. Allerdings müssen die als ausgeprägt und spezifisch eingestuften Anforderungen anschließend manuell durch den Komponentenentwickler auf ihre variantenverursachende Wirkung geprüft werden. Durch diesen zusätzlich notwendigen manuellen Arbeitsschritt kann die Methode nicht als automatisiertes Verfahren zur Identifizierung von variantenverursachenden Komponentenanforderungen bezeichnet werden. In Folge der Erkenntnis, dass unterschiedliche Anforderungen in den Komponentenlastenheften nicht zwingend zu Varianten der Komponente führen, aber die Konzeptentscheidungen als Anforderungen auf

Teilsystem-, System- oder Produktebene spezifische Lösungen und dadurch Komponentenvarianten definiert haben, ist die Eignung der vorgestellten Methode auf die Dokumentation der Anforderungen auf diesen übergeordneten Ebenen zu überprüfen. Dabei ist die Herausforderung die Dokumentationsform der Anforderungen auf diesen Ebenen: Während das Komponentenlastenheft als Vertragsunterlage sehr detailliert und strukturiert Anforderungen in natürlicher Sprache als zusammenhängenden Text dokumentiert, sind die Dokumentationsformen auf Produkt- oder Systemebene vielfältiger und verteilter sowie seltener als Fließtext in einem Dokument analog einem Lastenheft zu finden. Z.B. werden Anforderungen auf diesen Ebenen in Zielspinnen als Netzdiagramm, Use-Case-Diagramm oder anderen grafischen Dokumentationsformen dargestellt. Die Voraussetzung zur Anwendung der vorgestellten Methode auf die der Komponente übergeordneten Ebenen ist, dass auch auf diesen Ebenen Anforderungen in natürlicher Sprache in Textform als Dokumente oder in Datenbanken vorliegen.

5. Zusammenfassung und Ausblick

Trotz zahlreicher in der Literatur beschriebenen Ansätze zur strukturierten Wiederverwendung von Anforderungen in der Produktentwicklung herrscht in der Praxis laut Studien ein Copy-Paste-Vorgehen vor. Doch dadurch wächst die unstrukturierte Anforderungsmenge weiter und es mangelt gleichzeitig an Transparenz über die Unterschiede und Gleichheit zwischen den Anforderungen in den Entwicklungsprojekten. Um bereits in der frühen Phase der Produktentwicklung während der Definition von Anforderungen Varianten zu vermeiden, ist bei einer Wiederverwendung von Anforderungen aus Vorgängerprojekten eine Transparenz über variantenverursachende Anforderungen erforderlich. Werden Anforderungen in Merkmale des Produktes übersetzt, so führen Unterschiede in den Anforderungen zu verschiedenen Produktmerkmalen und somit neuen Produktvarianten. Doch aufgrund der großen Anforderungsmenge ist der manuelle Vergleich von Anforderungen und somit die manuelle Identifikation der Unterscheide in der Praxis zu aufwendig. Daher ist eine automatisierte Methode erforderlich.

Aus dieser Motivation heraus wurden zwei Lösungshypothesen für diese Arbeit formuliert. Die erste Lösungshypothese besagt, dass anhand eines automatisierten, semantischen Anforderungsvergleichs Unterschiede in einer großen, bestehenden und unstrukturierten Anforderungsmenge identifiziert werden können. Die zweite darauf aufbauende Lösungshypothese besagt, dass die anhand des Vergleichs festgestellten Unterschiede in den Anforderungen zu Varianten des Produktes führen. Daher werden die Anforderungen basierend auf den Ergebnissen des automatisierten Vergleichs in die drei Gruppen projektübergreifend, spezifisch und ausgeprägt klassifiziert.

Zur Untersuchung der Lösungshypothesen wurde die entwickelte Methode auf Komponentenlastenhefte der Entwicklung Fahrdynamik der BMW AG angewendet. Die Anwendung soll drei Aspekte der Methode überprüfen: Die Güte des automatisierten Anforderungsvergleichs, die Einordnung der Anforderungen in die drei Gruppen projektübergreifend, spezifisch und ausgeprägt und abschließend die Identifizierung variantenverursachender Anforderungen. Hierzu wurden zu Beginn in Kapitel 2.4 insgesamt sechs Fragen formuliert.

Das Ziel des automatisierten Anforderungsvergleichs ist, eine große Anzahl Anforderungen in Komponentenlastenheften automatisiert inhaltlich miteinander zu vergleichen, um eine Auswertung des Vorkommens der Anforderungen in den Lastenheften zu erhalten. Die Methode verwendet Technologien aus dem Bereich NLP und ist daher begrenzt auf textlich definierte Anforderungen. Es können keine Anforderungen aus

technischen Zeichnungen, Abbildungen oder Diagrammen mithilfe dieser Methode interpretiert oder verglichen werden. Des Weiteren liegt der Fokus der untersuchten Methode auf deutschsprachigen Anforderungen in Lastenheften für mechanische Komponenten. Um einzelne Anforderungen miteinander vergleichen zu können, muss zunächst definiert werden, wo eine Anforderung beginnt und endet. Hierzu werden Regeln für die automatisierte Textzerlegung und Methoden des Preprocessing aus dem NLP angewendet. Da die Anforderungen in Lastenheften in Fließtext vorliegen, ist zunächst ein Preprocessing erforderlich. Dieses Preprocessing soll den Fließtext in einzelne, abgrenzbare Anforderungen zerlegen.

(3.1) Wie können durch definierte Zerlegungsregeln im Preprocessing einzelne Anforderungen voneinander aus einem Fließtext abgegrenzt werden?

Hierzu wurden in Kapitel 3.2.1 die angewandten Zerlegungsregeln vorgestellt. Die Auswertung der Güte dieser Zerlegungsregeln ergibt 93% korrekt zerlegte Textfragmente. In 992 Matches wurden Textfragmente mit Zerlegungsfehler identifiziert (7%). Die größte Herausforderung bei der automatisierten Textzerlegung stellen Anaphorik und Kataphorik dar, die in den natürlichsprachigen Anforderungen eine große Rolle spielen. Für die Verknüpfung der Anaphern und Kataphern mit dem von ihnen referenzierten Ausdruck bestehen keine automatisierbaren Verfahren für den deutschsprachigen Bereich. Ebenso herausfordernd ist der Umgang in der Textzerlegung mit Spiegelstrichten, Aufzählungen, Nummerierungen, Textbezüge und inhaltliche Verweisen. Wie bereits Almefelt et al. (2006, S. 122) in ihrer Studie bei einem schwedischen Automobilhersteller feststellen, erfolgen innerhalb der Anforderungsdokumente viele Vorwärts- und Rückwärtsbezüge zu begleitenden Unterlagen und Dokumenten, die teils nicht für alle am Entwicklungsprojekt Beteiligten zugänglich sind. Diese hohe Anzahl an Verweisen auf begleitende Unterlagen kann im Rahmen dieser Untersuchung ebenfalls bestätigt werden. So entsprechen von den 13.546 Matches 15,5% dem Objekttyp Verweis (s. Kapitel 4.4.1 Abschnitt (a)).

Nach erfolgter Zerlegung des Fließtextes in einzelne Anforderungen erfolgt der semantische Vergleich anhand eines zuvor antrainierten Sprachmodels. Bestehende Ansätze unterteilen sich in wissensbasierte und korpusbasierte Ansätze. Wissensbasierte Ansätze erfordern eine linguistische Analyse des entsprechenden Sprachraums und eine Definition der Wortbeziehungen untereinander. Daher ist ihr Aufbau entsprechend aufwendig. Korpusbasierte Ansätze hingegen erfassen die semantische Bedeutung von Wörtern anhand der statistischen Verteilung bzw. mit welchen anderen Begriffen die Wörter häufig zusammen auftreten. Solche Ansätze wiederum benötigen

eine große Menge an Trainingsmaterial. Im spezifischen Fall der technischen Anforderungen ist der Zugriff auf Trainingsmaterial allerdings stark eingeschränkt, sodass ein Ansatz benötigt wird, der zum einen keine aufwendige linguistische Analyse des verwendeten Sprachraums benötigt und zum anderen auch bei einer kleinen Anzahl an zur Verfügung stehenden Trainingsdokumenten funktioniert. Daher wird der Ansatz Semantic Folding Theory von Webber (2016) für die Methode angewendet. Anhand des Semantic Folding sollen die zuvor zerlegten Anforderungen semantisch miteinander verglichen werden können.

(3.2) Kann eine Software inhaltlich ähnliche, aber unterschiedlich formulierte Anforderungen erkennen und einander zuordnen?

Nachdem der Fließtext der Lastenhefte im Preprocessing in einzelne Textfragmente zerlegt wurde, wurden diese Fragmente durch die Software anhand des trainierten Sprachmodells miteinander verglichen. Wird durch die Software ein Ähnlichkeitswert von über 35% ermittelt, werden die Textfragmente als inhaltlich ähnlich einander zugeordnet. Die Auswertung der Güte dieser Zuordnung erfolgt anhand der Kennzahlen precision, recall, accuracy und F1. Die precision des untersuchten Sprachmodells liegt insgesamt über alle Komponenten bei 96,6% und der recall bei 88,9%. Beide Werte liegen gemäß der Einstufung nach Hayes et al. (2005) im exzellenten Bereich. Zum Vergleich konnte Boutkova (2014, S. 59) in ihrem semiautomatischen Ansatz MIA eine precision zwischen 3% und 22% sowie einem recall zwischen 31% und 89% erreichen. Das im Rahmen dieser Arbeit angewandte Sprachmodell erzielte in der accuracy für alle Datensätze einen Wert von 92,3%. Das bedeutet, dass 92,3% der Zuordnungen korrekt sind. Auch der F1-Score als Kombinationswert aus precision und recall liegt bei 92,6%. Insgesamt kann diese semantische Zuordnung anhand des Semantic Folding basierend auf den betrachteten Umfängen als erfolgreich betrachtet werden. Wichtige Maßnahmen zur Verbesserung des semantischen Vergleichs liegen in der Einbeziehung der Kapitelüberschriften in jede einzelne Anforderung, da diese der Anforderung häufig ihren Kontext gibt. Zusätzlich werden auch Duplikate und inhaltlich ähnliche Textfragmente innerhalb eines Dokumentes zu einem zu vergleichenden Textfragment verknüpft. Dieses als Semantic Merging bezeichnete Vorgehen unterstützt den Ausgleich der unterschiedlichen Granularität, in der Anforderungen durch verschiedene Autoren formuliert werden. Grundsätzlich ist für den automatisierten Anforderungsvergleich zu betonen, dass es sich immer um eine semantische Ähnlichkeit und nicht um eine semantische Gleichheit handelt. Ab welchem Grad der Ähn-

lichkeit zwei Anforderungen als projektübergreifend einander zugeordnet werden sollen, ist abhängig vom Anwendungsfall für die Methode anhand von Musterbeispielen zu bestimmen.

Als nächsten Schritt der Methode werden basierend auf den Ergebnissen des automatisierten Anforderungsvergleichs die Anforderungen ebenfalls automatisiert in die Gruppen spezifisch, ausgeprägt und projektübergreifend klassifiziert. Diese Klassifikation erfolgt regelbasiert. Enthält die Anforderung eines der in Tabelle 3-1 genannten Ausprägungsmerkmale, wird sie als ausgeprägt klassifiziert. Enthält sie keines der definierten Ausprägungsmerkmale und kommt sie in mehr als 50% der verglichenen Dokumente vor, gilt sie als projektübergreifend, in allen anderen Fällen als spezifisch. Zu dieser regelbasierten Klassifikation von Anforderungen soll folgende Forschungsfrage beantwortet werden:

(1.1) Können Anforderungen basierend auf einem automatisierten Vergleich in die Gruppen projektübergreifend, ausgeprägt oder spezifisch klassifiziert werden?

Die Auswertung anhand von 13.546 Matches hat ergeben, dass 78,3% der Klassifikationen basierend auf den Regeln korrekt war, 21,7% wurden in die falsche Gruppe klassifiziert. Betrachtet man nur die Klassifikation der Anforderungen basierend auf den zuvor definierten Regeln und lässt die Matches mit fehlerhafter Zerlegung und falscher semantischen Zuordnung außen vor, werden 88,3% der Anforderungen korrekt klassifiziert. Allerdings ist die Auswertung nur an dem Vergleich von jeweils zwei Lastenheften validiert worden. Werden mehr als zwei Lastenhefte verglichen, ist noch zu diskutieren, ob die Regel „>50% Vorkommen = projektübergreifend“ im Sinne einer allgemeingültigen Anforderung zutrifft. Insbesondere bei einem Technologiesprung kann eine bislang allgemeingültige Anforderung für neuere Entwicklungsprojekte ungültig werden. Bilden aber der Großteil der verglichenen Lastenhefte noch den alten Technologiestand ab, so würde die Auswertung weiterhin ein projektübergreifend ergeben. Das führt direkt zur Diskussion der nächsten Frage:

(1.2) Ist durch die Klassifizierung eine produktspezifische Konfiguration eines Lastenheftes aus einer Anforderungsbibliothek einer Komponente möglich?

Zur Beantwortung dieser Frage wurden die Ergebnisse des Anforderungsvergleichs von vier verschiedenen Komponenten mit den jeweils komponentenverantwortlichen Entwicklern in Rahmen von 8 Experteninterviews diskutiert. Das Ergebnis dieser Experteninterviews ist, dass 88% der Entwickler den Anforderungsvergleich dafür nut-

zen wollen, um für ihre Komponente eine Anforderungsbibliothek aufzubauen. Allerdings fehlte den Entwicklern zum Aufbau einer Anforderungsbibliothek in der Auswertung des Anforderungsvergleichs die Chronologie der Anforderungen als auch zusätzliche Attribute zur Technologie oder der Produktvariante als auch ein Archiv-Attribut. Technologie-Attributen wie z.B. „angetriebene Achse“, „Doppel-Querlenker-Achse“ als Achskonzept, Antriebsform ICE/BEV/PHEV oder auch die jeweilige Produktlinie des Komponentenlastenheftes stellen sich in den Gesprächen als nützliche Attribute bei der späteren projektspezifischen Konfiguration eines Komponentenlastenheftes aus der Anforderungsbibliothek heraus. Diese zusätzlichen Informationen sind bereits in dem Großteil der Lastenhefte enthalten und könnten entsprechend automatisiert durch ein Annotations-Tool aus dem Bereich NLP ermittelt werden. 100% der befragten Entwickler wollen durch die Auswertung die Formulierungen vereinheitlichen und 63% können sich vorstellen, durch den Anforderungsvergleich den Umfang ihrer Lastenhefte reduzieren zu können.

Ein weiteres Ziel der entwickelten Methode ist die Identifizierung variantenverursachender Anforderungen. In der Literatur zeigt nur der Ansatz MIA von Boutkova (2014) eine Möglichkeit zur Identifizierung von Variantentreibern über die Modellierung der Variabilität von Anforderungen in Merkmalsbäumen auf. Allerdings liegt dessen precision zwischen 0,03 und 0,22 und recall zwischen 0,31 und 0,89. Boutkova (2014, S. 59) und Mamrot et al. (2013, S. 351) stellen fest, dass im Variantenmanagement bislang noch kein Vergleich von Anforderungen durchgeführt wurde. Daher wird zur Untersuchung der zweiten Lösungshypothese zur Identifizierung variabler Anforderungen als Variantentreiber folgende Forschungsfragen diskutiert:

(2.1) Sind spezifische oder ausgeprägte Anforderungen variantenverursachend?

(2.2) Können durch einen automatisierten Anforderungsvergleich mit anschließender Klassifizierung variantenverursachende Anforderungen identifiziert werden?

In Experteninterviews sollten Entwickler für ihre Komponente zunächst bereits bekannten Variantentreiber nennen, aufgrund dessen eine neue Variante ihrer Komponente erforderlich wird. Die genannten Variantentreiber liegen nicht auf der Komponentenebene. Alle der genannten Variantentreiber sind Entscheidungen, die auf Produkt-, System- oder Teilsystemebene getroffen wurden. Anschließend sollten 142 als spezifisch oder ausgeprägte Anforderungen aus dem Anforderungsvergleich im Rahmen der Experteninterviews in Merkmale der Fahrwerkskomponente übersetzt werden.

Alle ausgewählten Anforderungen weisen einen inhaltlichen Unterschied in den verglichenen Lastenheften auf. Gemäß der Annahme, dass Unterschiede in den Anforderungen zu unterschiedlichen Merkmalen führen, müsste ein Einfluss der besprochenen Anforderungen in Merkmale oder Funktionen der Komponente übersetzt werden können. Anhand dieser Merkmale sollte der variantenverursachende Charakter von spezifischen und ausgeprägten Anforderungen nachgewiesen werden. Allerdings war dies für 55%-83% der besprochenen Anforderungen für die zuständigen Entwickler aus unterschiedlichen Gründen nicht möglich (s. Kapitel 4.4.3). Im Hinblick auf die genannten Variantentreiber der Komponenten kann die untersuchte Methode nur wenige variantenverursachenden Anforderungen in den Komponentenlastenheften finden, da die Entscheidung über die Varianz der Komponente auf einer anderen Ebene z.B. dem System Fahrdynamik bei der Entscheidung über das Achskonzept getroffen wurde. In anderen Fällen war die Entscheidung über die Variantenvielfalt bereits auf Produktebene getroffen, als entschieden wurde, ob das Fahrzeug als Front-, Heck- oder Allradantrieb oder mit mehreren Antriebsvarianten angeboten werden soll. Um tatsächlich die maßgebenden Variantentreiber zu identifizieren, sind basierend auf dieser Erkenntnis bereits die Anforderungen bei der Auslegung des Systems oder Teilsystems zu untersuchen.

Zusätzlich ist auch die Lokalisierung der Komponente als die Vergabe an einen Lieferanten in Ländern niedrigerer Herstellkosten als Variantentreiber zu diskutieren, die sich nicht in den Komponentenlastenheften als Anforderung zeigt. Bei der Vergabe einer Komponente als Black-Box-Teil (s. Kapitel 2.1.1) erfolgt zunächst eine Ausschreibung basierend auf dem Lastenheft, auf die sich mehrere Lieferanten bewerben. Jeder Lieferant interpretiert lösungsoffene Anforderungen innerhalb des verbleibenden Lösungsraums. Dadurch entstehen unterschiedliche Lösungen bei gleicher Anforderung. Soll die Variantenentstehung durch solche Interpretationen des Lösungsraumes verhindert werden, müssten alle Komponenten als Build-to-Print ausgeschrieben werden, die keinen Entwicklungsumfang mehr darstellen, sondern nur die reine Produktion durch einen Lieferanten beauftragen. Dies ist gerade zur Entlastung der eigenen Entwicklung und in Anbetracht der Vorteile von Entwicklungskooperationen nicht sinnvoll. Vielmehr ist es für Unternehmen zielführender, im Sinne ihres jeweiligen Baukasten-, Plattform- oder Architekturkonzepts eine Komponente für möglichst viele ihrer Endprodukte zu vergeben und dadurch Skaleneffekte zu erzielen. An dieser Stelle ist die Transparenz über die Unterschiede als auch die Ähnlichkeit zwischen den Anforderungen an eine Komponente wichtig, um die Komponente für den Einsatz in den unterschiedlichen Endprodukten auf möglichst wenige Varianten zu skalieren.

Des Weiteren wurde aufgrund der Untersuchung der Variantentreiber auch eine manuelle Auswertung der Objekttypen der Textfragmente in den Komponentenlastenheften durchgeführt, bei dem festgestellt wurde, dass in den untersuchten Fallbeispielen nur 55-73% der Textfragmente Anforderungen sind. Weitere Bestandteile der Lastenhefte sind Verweise auf mitgeltende Unterlagen, Informationen zum Kontext der Komponente oder Leistungsvereinbarungen als Projektanforderungen zur Zusammenarbeit zwischen Auftragnehmer und Auftraggeber.

Auch wenn durch die entwickelte Methode nur in geringen Umfang variantenverursachenden Anforderungen in den Komponentenlastenheften identifiziert werden konnten, ist die automatisierte Strukturierung der Anforderungen durch den Anforderungsvergleich aufgrund der gezeigten Güte vielversprechend. Die Automatisierung des Anforderungsvergleichs unterstützt die Entwickler in der Strukturierung und Wiederverwendung ihrer Anforderungen bei gleichzeitig geringem Aufwand. Dieser Bedarf nach automatisierbaren Methoden zur Unterstützung des Anforderungsmanagements wurde bei der BMW AG durch diese Arbeit erkannt, die Möglichkeiten über Ansätze des NLP aufgezeigt und wird nun durch das Projekt RELYZE fortgesetzt.

Ausblick

Zur Wiederverwendung von Anforderungen und dem Aufbau einer Anforderungsbibliothek ergeben sich Möglichkeiten zur Erweiterung der Methode. Durch eine automatisierte Qualitätsanalyse der Anforderungsformulierung gemäß den Kriterien des IEEE Std 830, S. 4 könnte sichergestellt werden, dass ausschließlich qualitativ gut formulierte Anforderungen in die Anforderungsbibliothek aufgenommen werden. Hier besteht allerdings noch weiterer Forschungsbedarf, wie eine Methode des NLP erkennen kann, ob eine Anforderung korrekt, eindeutig, vollständig, konsistent, nachverfolgbar oder prüfbar ist. Weiterer im Rahmen der Arbeit erkannte Bedarf besteht in der Erweiterung der Methode um eine Attributierung der Anforderungen nach Technologien, Produktlinien oder Konzepten. Dies könnte durch ein Annotationstool automatisiert erfolgen. Zusätzlich sollte basierend auf der Erkenntnis, dass nur 55-73% der Textfragmente in den Komponentenlastenheften Anforderungen sind, eine Möglichkeit zur Erkennung der unterschiedlichen Objekttypen Information, Verweis, Anforderung oder Leistungsvereinbarungen in einem Lastenheft untersucht werden. Damit könnten ausschließlich bestimmte Objekttypen in die Anforderungsbibliothek übernommen werden.

Im Rahmen der Arbeit haben sich in der automatisierten Textverarbeitung Defizite in der automatisierten Verknüpfung von Anaphern und Kataphern mit ihrem Referenzausdruck ergeben. Mannion et al. (1999, S. 454) beschreiben den Idealfall, dass wiederverwendbare Anforderungen eigenständig für sich stehen können und dadurch unabhängig voneinander konfiguriert werden können. Wie sich aber in der Untersuchung dieser Arbeit gezeigt hat, spielen Anaphorik und Kataphorik in den natürlichsprachigen Anforderungen im Fließtextformat eine große Rolle. Aber auch Aufzählungen stellen in der Textzerlegung eine Herausforderung dar, da hier häufig das Ende der Aufzählung nicht durch klar definierbare Regeln erkannt werden kann z.B., wenn die Aufzählungen nicht mit Spiegelstrichen, Nummerierungen oder anderen Aufzählungszeichen formatiert sind. Hier besteht noch weiterer Forschungsbedarf, um Anaphorik und Kataphorik in deutschsprachigen Anforderungen korrekt dem richtigen Textbezug zuordnen zu können.

Des Weiteren hat die Überprüfung der variantenverursachenden Anforderungen gezeigt, dass die Übersetzung von Anforderungen in Produktmerkmale nicht trivial ist. Viele der besprochenen Anforderungen aus den Komponentenlastenheften konnte durch die komponentenverantwortlichen Entwickler nicht in Merkmale übersetzt werden. Dies zeigt den Bedarf zur Untersuchung möglicher Ansätze für den Transfer von

Anforderungen in Merkmale, um die Entwickler in der Praxis methodisch zu unterstützen, sowie den Forschungsbedarf zur Untersuchung der Eignung solcher Anforderungen, die nicht in Merkmale übersetzt werden können.

Letztlich hat diese Arbeit auch gezeigt, dass Komponentenvarianten bereits durch die Anforderungen und Entscheidungen auf einer höher gelegenen Produktebene definiert wurden. Die Fallbeispiele der vorgestellten Komponenten haben gezeigt, dass bereits bei der Definition der Antriebsform oder des Achskonzepts die Komponentenvarianten indirekt definiert wurden. Somit sollte die Methode zur Identifizierung variantenverursachender Anforderungen auf die Anforderungen des Produktes, Systems oder Teilsystems ausgeweitet werden. An dieser Stelle könnten zukünftige Forschungsvorhaben Möglichkeiten zur automatisierten Interpretation von Abbildungen und des Vergleichs der in ihnen enthaltenen Anforderungen untersuchen, da die Anforderungen auf diesen Ebenen häufig in visuellen Dokumentationsformen wie Netzdiagrammen, Abbildungen, Use-Case-Diagrammen etc. definiert sind.

Anhang

Anhang A: Bewertung Literatur - Wiederverwendung von Anforderungen

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	technische Anforderungen	mechanische Komponenten	Wiederverwendung
Mamrot et al. (2013)	DeCoDe	Unterscheidung changeable und unchangeable requirements, Verknüpfung Anforderung mit Funktionen, Komponenten und Prozessen durch DeCoDe und anschließender Übersetzung in Design Variablen	Wiederverwendung Anforderungen, Darstellung Varianz Anforderungen	●	●	●	●
Boutkova (2014)	MIA, R2F, KDVM	Merkmalsmodellierung; Variabilitätsmodell, Zuordnung der Anforderungen zu Variabilitätskriterien; Lastenheftkonfiguration anhand Variabilitätsmodell	Variantenmanagement in Anforderungsdokumenten bei Mercedes-Benz AG	●	●	●	●
Tavakoli Kolagari et al. (2007)	-	Anforderungsbibliothek, Variabilitätsmodellierung von funktionalen Anforderungen	Wiederverwendung von Softwareanforderungen aus Anforderungsbibliotheken bei Daimler Chrysler	◕	●	◑	●
Mannion et al. (2008)	MRAM	Unterscheidung in common und variable requirements (verifiziert anhand Spacecraft control operating system)	Wiederverwendung von Anforderungen in Produktlinien anhand von Parametern und Diskriminanten	◕	●	◑	●
Hauksdottir et al. (2012)	Adjustable requirement reuse	Aufbau eines company repository (Anforderungsverzeichnis), Identifizierung der Variationsstellen	Wiederverwendung von Anforderungen bei Danfoss GmbH	◕	●	◑	●

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	technische Anforderungen	mechanische Komponenten	Wiederverwendung
Knauss et al. (2014)	Socio-technical system	Preprocessing (k-gram, indexing), classification (support vector machine), post-processing (topic generalization)	Automatisierte Kategorisierung von Anforderungen in Dokumenten bei Mercedes-Benz AG	◑	●	●	○
Gülke (2014)	RMnet	semantisches Netz (Ontologie) als Wissensbasis für Anforderungsmanagement	Verknüpfung Anforderungsartefakte und deren Auswirkungen aufeinander bei Volkswagen AG	◑	●	●	○
Cybulski et al. (1998 & 2000)	RARE IDIOM	Vergleich neuer Anforderungen mit existierenden Anforderungsdokumenten	Wiederverwendung von Softwareanforderungen	◑	●	○	●
Konrad et al. (2002)	-	Requirements Pattern, Softwareanforderungen	Wiederverwendung von Anforderungen anhand von UML Repräsentation	◑	●	○	●
Heumesser et al. (2003)	-	Unterscheidung von modellabhängigen und modellunabhängigen Anforderungen	Wiederverwendung von Systemanforderungen bei ECUs der Mercedes-Benz AG	◑	●	○	●
Mei et al. (2003)	FODM	feature oriented domain modeling method, feature model	Modellierung und Wiederverwendung von Anforderungen in Softwareproduktlinien	◑	●	○	●
Dehlinger et al. (2008)	DECIMAL	Commonality and Variability Analysis, Variation Point Schema	Wiederverwendung und Verifizierung von multi-agent System Produktlinien Anforderungen	◑	●	○	●

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	technische Anforderungen	mechanische Komponenten	Wiederverwendung
Monzon (2008)	MIA Add In	common requirement from parental document, strong and weak reuse, specific project requirements, derived and non-derived requirements	Wiederverwendung von Anforderungen in Produktfamilien von On-Board Systemen in Militär-Transportflugzeugen	◑	●	○	●
Renault et al. (2009)	PABRE	Pattern	Auswahl und Wiederverwendung von Anforderungen durch Pattern	◑	●	○	●
Chernak (2012)	-	-	Studie zu Wiederverwendungsmethoden in der Praxis	◑	●	○	●
Franch (2013)	-	Pattern	Wiederverwendung von Softwareanforderungen durch Pattern	◑	●	○	●
Naz et al. (2013)	CB-RCM model	Initialisierungsphase (Änderungsinitialisierung einer Anforderung, Vorprüfung und Vergleich mit vorhandenen Problemen/Fällen), Zwischenphase (Anforderungsnachverfolgung, Änderungseinflussanalyse, Zielkonfliktmanagement), Endphase (Entscheidung)	Einflussanalyse bei Anforderungsänderung	◑	●	○	●
Karatas et al. (2014)	OntSRDT	Ontologie zur Konfiguration, Repository (Anforderungsverzeichnis), konstante und variable Anforderungen	Wiederverwendung Anforderungen von Softwareproduktlinien	◑	●	○	●
Niu et al. (2014)	-	automatisierte Identifizierung Funktionaler Anforderungsprofile durch NLP	Wiederverwendung von Softwareanforderungen in Produktlinien	◑	●	○	●

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	technische Anforderungen	mechanische Komponenten	Wiederverwendung
Palomares et al. (2014)	-	Pattern	Studie Praxis: Wiederverwendung durch Pattern in Softwareentwicklung	◑	●	○	●
Garcia et al. (2016)	RRXRC	Anforderungskatalog	Wiederverwendung von Anforderungen in Rational Requisite Pro	◑	●	○	●
Darimont et al. (2017)	-	Template and Pattern Library	Entwicklung einer Pattern Anforderungsbibliothek zur Wiederverwendung von Anforderungen	◑	●	○	●
Kim et al. (2004)	DREAM	Unternehmensziel- und Szenariomodellierug zur Identifizierung von Anforderungsvarianten	Identifizierung von Anforderungsvarianten	◔	○	◑	◑

Anhang B: Bewertung Literatur - Varianten im AM

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	Variantentreiber	Variabilität von Anforderungen	technische Anforderungen	mechanische Komponenten
Renner (2007)	-	Anforderungsoptimierung und -harmonisierung	Baukastenentwicklung zur Variantenbeherrschung	●	●	●	●	●
Stechert (2010)	-	Partialmodelle verschiedener Abstraktionsgrade in SysML, Darstellung Anforderungsabhängigkeiten in Modell, Anwendungsfall-Anforderungs-Matrix zur Identifizierung Anforderungsspreizung	komplexe Anforderungen an mechatronische Produkte systematisch erfassen und aufbereiten	●	●	●	●	●
Boutkova (2014)	MIA, R2F, KDVM	Merkmalsmodellierung; Variabilitätsmodell, Zuordnung der Anforderungen zu Variabilitätskriterien; Lastenheftkonfiguration Variabilitätsmodell	Variantenmanagement in Anforderungsdokumenten bei Mercedes-Benz AG	◕	○	●	●	●
Mamrot et al. (2013)	DeCoDe	Unterscheidung änderbare und unveränderliche Anforderungen, Verknüpfung Anforderung mit Funktionen, Komponenten und Prozessen durch DeCoDe und anschließender Übersetzung in Design Variablen	Wiederverwendung Anforderungen, Darstellung Varianz Anforderungen	◕	○	●	●	●
Rohleder (2008)	-	Variabilitätsmodelle als Graphen	Einfluss nichtfunktionaler Anforderungen auf Varianten	◕	●	●	●	○
Stoiber et al. (2010)	feature unweaving	manuelle Identifizierung variabler Elemente, Übersetzungen dieser in Funktionen, Erstellung eines grafischen Anforderungsmodells, Einführung von Variabilitäts-Beschränkungen	Erstellen und Spezifizieren von Kommunalität und Variabilität in Softwareproduktlinien	◕	●	●	●	○

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	Variantentreiber	Variabilität von Anforderungen	technische Anforderungen	mechanische Komponenten
Tavakoli Kolagari et al. (2007)	-	Anforderungsbibliothek, Variabilitätsmodellierung von funktionalen Anforderungen	Wiederverwendung von Softwareanforderungen aus Anforderungsbibliotheken bei Daimler Chrysler	◕	○	●	●	◑
Almefelt et al. (2006)	-	Studie bei schwedischen Automobilhersteller und Lieferanten, Kommunalität von Anforderungen und Komponenten	AM in der Praxis am Beispiel Cockpit (Automobilindustrie)	◕	○	◑	●	●
Alders (2006)	VAMOS – Variantenmanagement und Optimierungssystem	Transparenz und Analyse in Variantenplanung, Zuordnung der Variantenkosten auf Merkmalvarianten in Merkmalsbaum	Optimierung Variantenzahl von Bauteilen durch Analyse Verkaufsdaten; Verbauwahrscheinlichkeit AUDI AG	◑	●	○	○	●
Bühne et al. (2004, 2005)	-	Modellierung Variabilität der Anforderungsartefakte über alle Abstraktionsstufen in einem Modell, orthogonales Variabilitätsmodell	Wiederverwendung und durchgängige Dokumentation von Anforderungen von komplexen Systemen	◑	○	●	●	○
Caesar (1991)	VMEA	Beherrschung technischer und wirtschaftlicher Auswirkungen der anzubietenden Produktvielfalt	Variantenvermeidung für Kunden nicht erkennbarer Varianten	◑	●	○	○	●
Das et al. (2021)	wissensbasierte Variantenentwicklung		Wissensbasis (Ontologie)zur Variantenentwicklung	◑	○	◑	○	●
Huysegoms et al. (2013)	-	formal concept analysis (FCA), Harmonisierung und Variabilisierung von Stakeholder Anforderungen für Software	Visualisierung von Variabilität in Anforderungsdokumenten	◑	◑	●	◑	○

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	Variantentreiber	Variabilität von Anforderungen	technische Anforderungen	mechanische Komponenten
Knauss et al. (2014)	Socio-technical system	NLP: Preprocessing k-gram, indexing; Klassifizierung durch Support Vector Machines, Post-processing topic generalization	Automatisierte Kategorisierung von Anforderungen in Dokumenten bei Mercedes-Benz AG	◑	○	○	●	●
Schuh et al. (2016)	-	Studie Fragebogen an Unternehmen zu Variantentreibern bei Dienstleistungen	Identifizierung variant-creating factors variety drivers; Fokus auf PSS (Dienstleistungen)	◑	●	●	○	○
Tseng et al. (1997)	-	Identifizierung funktionaler Anforderung Patterns in vorhandenen Anforderungsdokumenten	Anpassungsentwicklung durch Modifikation vorhandener funktionaler Anforderungspattern	◑	○	●	●	○
Manicke (2012)	VIS - Varianten Informations-System	Abbildung prognostizierte Ve3rbauhäufigkeiten auf Varianten, Ähnlichkeitsanalyse elektronischer Funktionen	Zerlegung Produkt in Einzelteile in Tableau zur Identifizierung Varianten und ihrer Ursachen	◑	●	○	◑	◑
Robinson-Mallett et al. (2009)	-	Zustandsautomaten, Selektionsverfahren, Parametrisierung	Variantenmanagement für funktionale Anforderungen bei Mercedes Benz AG	◑	○	●	◑	◑
Schuh et al. (2014)	-	Visualisierung von Ähnlichkeiten zwischen Produktvarianten	Identifizierung von bevorzugten Produktvarianten anhand Kundenanforderungen	◑	○	◑	○	●
Franke et al. (2002)	EVA-PRO	Fokus Einzel und Kleinserienfertigung, Methodendatenbank	Zusammenstellung einer Methodendatenbank im Variantenmanagement	◑	◑	○	○	●
Kesper (2012)	LOO-MEO	Einschränkung zulässiger Eigenschafts- und Komponentenkombinationen, Kosten Nutzen Bewertung von Produktvarianten	Vermeidung und Reduzierung der Variantenvielfalt: Festlegung von Produktvariantenspektren	◑	◑	○	○	●

Autor (Jahr)	Methode	Vorgehen	Ziel	Summe	Variantentreiber	Variabilität von Anforderungen	technische Anforderungen	mechanische Komponenten
AlGeddawy et al. (2013) ElMaraghy et al. (2012)	Reactive Platform Design Model	Identifizierung von Kommunalität und Variabilität mithilfe von Cladistics (Methode aus der Biologie) anhand Produktmerkmalen, Ergebnis ist ein grafisches Clusterdiagramm (cladogram), benachbarte Komponenten im Graphen werden zu Modulen, Verwendung von Liaison-Graphen aus der Montageprozessplanung zur Identifizierung benachbarter Komponenten	Identifizierung beste Produktfamilie anhand Abhängigkeitsmatrix und Liaison Grafen, Wiederholteilsuchsystem	◑	◑	○	○	●
Dey et al. (2015)		Repertory Grid Interviewtechnik, Stakeholder Erwartungen und Intensionen, self-adaptive systems	Variabilitätsanalyse	◔	○	●	○	○
Wang et al. (2015)	Naive Bayes		Zuordnung Kundenanforderungen zu Produktvarianten	◔	○	●	○	○
Kubica (2007)	AMoS	Merkmalsmodellierung mit FODA	Wiederverwendung von Softwarekomponenten in Software Produktlinien Automobil AUDI AG	◔	◑	○	◑	○

Anhang C: Bewertung Literatur - NLP Ansätze im Anforderungsmanagement

Autor (Jahr)	Methode	NLP	Ziel	Summe	semantische Ähnlichkeit	technische Anforderungen	deutschsprachig	mechanische Komponenten
Jörg (2005)	ReMaS	Preprocessing (splitting, tagging, stemming, parsing); Semantik: GermaNet	Anforderungsmodellierung, Verknüpfung zu CAD/ PDM, Abhängigkeiten durch semantisches Anforderungsnetz, Disambiguierung	●	◑	●	●	●
Körner (2014)	RECAA, RESI	Ontologie, WordNet, POS, Stemming, Parsing	automatisierte Anforderungsverarbeitung Softwareentwicklung	◕	●	●	○	○
Arora et al. (2015)	CIA	text chunking, tokenization, SEMILAR (path)	Einflussanalyse von Anforderungsänderungen	◕	●	●	○	○
Cybulski et al. (2000)	RARE IDIOM	parsing, reference identification, semantic network, thesaurus	Wiederverwendung von Softwareanforderungen	◕	●	●	○	○
Kläger (1993)	DIICAD	Constraint Netz, semantisches Netz der Anforderungsstruktur	Anforderungsmodellierung, Verknüpfung mit Konstruktionswissen	◕	○	●	●	●
Kickermann (1995)	ALL-TOOL	LEX, YACC zur Anforderungssyntax, KNN	Formalisierung von Anforderungen (Anforderungssyntax), Ableitung Zielfunktion, Verknüpfung Anforderung zu Konstruktionsdaten	◕	○	●	●	●
Rzehorz (1998)	-	Ontologie	Anforderungsbibliothek, Zerlegung und Strukturierung von Anforderungen, wissensbasierte Anforderungsentwicklung	◕	○	●	●	●
Gebauer (2001)	-	Ontologie	semantisches Anforderungsnetz, Abhängigkeiten zwischen Anforderungen	◕	○	●	●	●

Autor (Jahr)	Methode	NLP	Ziel	Summe	semantische Ähnlichkeit	technische Anforderungen	deutschsprachig	mechanische Komponenten
Mayer-Bachmann (2008)	-	-	Modell zur Verbesserung Produktdatendurchgängigkeit, Produkt-DNA	◕	○	●	●	●
Kolbe et al. (2011) Kolbe (2014)	-	Ontologie	Ontologie für Anforderung an komplexe Systeme, Beziehung zw. Anf.	◕	○	●	●	●
Gülke (2014)	RMnet	Ontologie, semantische Netze	Verknüpfung Anforderungsartefakte und deren Auswirkungen aufeinander	◕	○	●	●	●
Boutkova (2014)	MIA, R2F	POS, Lemmatisierung, Stop word-Removal,	Variantenmanagement in Anforderungsdokumenten	◕	○	●	●	●
Knauss et al. (2014)	Socio-technical system	k-gram, indexing, support vector machines, topic generalization	Automatisierte Kategorisierung von Anforderungen in Dokumenten	◕	○	●	●	●
Alabdulkareem et al. (2015)	Goal and Preference Identification	UMBC STS (LSA kombiniert mit WordNet)	Zielsystem	◑	●	◑	○	○
Cyriaks et al. (2007)	Soft-Wiki-Projekt	POS, Lemmatisierung, Nachbarschafts- und Satzkookurenzen	Anforderungserhebung aus vorhandenen Dokumenten (z.B. Kundenemails)	◑	◔	◑	●	○
Ahrens (2000)	SPEC-tool	-	Unterstützung Erfassung und Handhabung Anforderungen	◑	○	●	●	○
Gacitua et al. (2010)	RAI	POS, Stop-word-Removal, Lemmatization, corpus-based frequency, syntactic pattern, lexical similarity	Identifizierung von Abstraktionen in Anforderungs-dokumenten	◑	○	●	○	●

Autor (Jahr)	Methode	NLP	Ziel	Summe	semantische Ähnlichkeit	technische Anforderungen	deutschsprachig	mechanische Komponenten
Ferrari et al. (2014)	CAR	POS, linguistic filter, term hood metric contrastive analysis	Vollständigkeit überprüfen	◑	○	●	○	●
Ferrari et al. (2018)	GATE	regelbasierter Ansatz; Tokenisierung, POS, Shallow Parsing, Gazetteer, JAPE Rules	Identifizierung von Fehlern in Anforderungen (pattern)	◑	○	●	○	●
Gelhausen (2010)	SENSE DMX	Omnigraph anhand Wortklassen, Bedeutungsgruppen/Rollen --> Verbindung mit Wörterbuch	natürlichsprachige Software-Spezifikation In UML-Modell übersetzen anhand Semantik (Fokus funktionale Anforderungen)	◑	○	●	●	○
Wang et al. (2015)	Naive Bayes	Naive Bayes classifier	Zuordnung Kundenanforderungen zu Produktvarianten	◑	◑	○	○	●
Riechert et al. (2007)	SWORE	SoftWiki Ontologie	semantisch unterstütztes Software RE	◔	○	○	●	○
Dwarakanath et al. (2013)	-	Segmentation, Dependency Parsing, term hood determination	Extraktion von Glossarbegriffen/ Glossareinträgen	◔	○	●	○	○
Naz et al. (2013)	CB-RCM model	AI (CBR case-based reasoning)	Einflussanalyse bei Anforderungsänderung	◔	○	●	○	○
Karatas et al. (2014)	OntSRDT	Ontologie	Wiederverwendung Anforderungen von Softwareproduktlinien	◔	○	●	○	○

Anhang D: öffentlich zugängliche Literatur zum Antrainieren des Sprachmodells

Allgemeine Standards	Fachliteratur
DIN EN ISO. **20567-1**:2017-07: Beschichtungsstoffe – Prüfung der Steinschlagfestigkeit von Beschichtungen – Teil 1: Multischlagprüfung. **DIN EN ISO**. **8044**:2015-12: Korrosion von Metallen und Legierungen – Grundbegriffe. **DIN EN ISO**. **9227**:2017-07: Korrosionsprüfungen in künstlichen Atmosphären – Salzsprühnebelprüfungen. **DIN ISO**. **8855**:2013-11: Straßenfahrzeuge – Fahrzeugdynamik und Fahrverhalten – Begriffe. **VDA (2015)**: Prüfung der Technischen Sauberkeit. 2. Aufl. (Qualitätsmanagement in der Automobilindustrie, 19) (Teil 1).. **VDA (2010)**: Technische Sauberkeit in der Montage - Umgebung, Logistik. Personal und Montageeinrichtungen. 1. Aufl. (Qualitätsmanagement in der Automobilindustrie, 19) (Teil 2). **VDI**. **2230 Blatt 1**:2015: Systematische Berechnung hochbeanspruchter Schraubenverbindungen Zylindrische Einschraubenverbindungen. **VDI**. **2230 Blatt 2**:2014: Systematische Berechnung hochbeanspruchter	**Isermann**, R. (Hg.) (**2006)**: Fahrdynamik-Regelung - Modellbildung, Fahrerassistenzsysteme, Mechatronik. 1. Aufl. Wiesbaden: Vieweg (ATZ/MTZ-Fachbuch). **Heißing**, B. (Hg.) (**2007)**: Fahrwerkhandbuch - Grundlagen, Fahrdynamik, Komponenten, Systeme, Mechatronik, Perspektiven. 1. Aufl. Wiesbaden: Vieweg (Aus dem Programm Kraftfahrzeugtechnik). **Theumert**, H.; **Fleischer**, B. (**2007)**: Entwickeln - Konstruieren - Berechnen - Komplexe praxisnahe Beispiele mit Lösungsvarianten. 1. Aufl. Wiesbaden: Vieweg (Viewegs Fachbücher der Technik). **Czichos**, H.; **Hennecke**, M. (Hg.) (**2007)**: HÜTTE - Das Ingenieurwissen. Akademischer Verein Hütte e.V. 33. Aufl. Berlin, Heidelberg: Springer. **Ruge**, J.; **Wohlfahrt**, H. (**2007)**: Technologie der Werkstoffe - Herstellung, Verarbeitung, Einsatz. 8. Aufl. Wiesbaden: Vieweg (Vieweg Studium Technik). **Steinhilper**, W.; **Sauer**, B. (**2008)**: Konstruktionselemente des Maschinenbaus 1 - Grundlagen der Berechnung und Gestaltung von Maschinenelementen. Berlin, Heidelberg: Springer. **Steinhilper**, W.; **Sauer**, B. (**2008)**: Konstruktionselemente des Maschinenbaus 2 - Grundlagen von Maschinenelementen für Antriebsaufgaben. Berlin, Heidelberg: Springer. **Pahl**, G.; **Beitz**, W.; **Feldhusen**, J.; **Grote**, K.-H. (**2007)**: Konstruktionslehre - Grundlagen erfolgreicher Produktentwicklung ; Methoden und Anwendung. 7. Aufl. Berlin, Heidelberg: Springer. **Muhs**, D.; **Wittel**, H.; **Jannasch**, D.; **Voßiek**, J. (**2007)**: Rooloff/Matek Maschinenelemente - Normung, Berechnung, Gestaltung. 18. Aufl. Wiesbaden: Vieweg (Studium). **Niemann**, G.; **Winter**, H.; **Höhn**, B.-R. (**2005)**: Konstruktion und Berechnung von Verbindungen, Lagern, Wellen. 4. Aufl. Berlin: Springer (Maschinenelemente / G. Niemann, H. Winter, B.-R. Höhn, Bd. 1).

Allgemeine Standards	Fachliteratur
Schraubenverbindungen Mehrschraubenverbindungen.	**Haberhauer**, H.; **Bodenstein**, F. (**2007**): Maschinenelemente - Gestaltung, Berechnung, Anwendung. 14. Aufl. Berlin, Heidelberg: Springer (Springer-Lehrbuch). **Künne**, B. (Hg.) (**2007**): Köhler/Rögnitz Maschinenteile 1. 10. Aufl. Wiesbaden: Teubner. **Rösler**, J.; **Harders**, H.; **Bäker**, M. (**2006**): Mechanisches Verhalten der Werkstoffe. 2. Aufl. Wiesbaden: Teubner (Lehrbuch Maschinenbau). **Meissner**, M.; **Schorcht**, H.-J. (**2007**): Metallfedern - Grundlagen, Werkstoffe, Berechnung, Gestaltung und Rechnereinsatz. 2. Aufl. Berlin, Heidelberg: Springer (VDI-/Buch]). **Matschinsky**, W. (**2007**): Radführungen der Straßenfahrzeuge - Kinematik, Elasto-Kinematik und Konstruktion. 3. Aufl. Berlin, Heidelberg: Springer. **Kloos**, K.-H.; **Thomala**, W. (**2007**): Schraubenverbindungen - Grundlagen, Berechnung, Eigenschaften, Handhabung. 5. Aufl. Berlin, Heidelberg: Springer.

Anhang E: Übersicht der verglichenen Komponentenlastenhefte

Versuch	Komponente	Dokument 1	Dokument 2	# Match
Versuch01	Hinterachsträger	11093034_02	10928726_00	362
Versuch01	Radlager	11340304_02	10260021_01	331
Versuch01	Schwenklager	11178653_05	10945704_04	372
Versuch01	Stützlager	11136245_03	10260072_02	230
Versuch01	Stützlager	11136245_03	10260080_02	254
Versuch02	Hinterachsträger	11024363_02	10900480_01	349
Versuch04	Hinterachsträger	11024363_02	10398997_05	517
Versuch04	Hinterachsträger	11024363_02	10569597_02	485
Versuch08	Hinterachsträger	10398997_05	10928726_00	473
Versuch08	Schwenklager	10945704_04	10335475_03	287
Versuch16	Hinterachsträger	10494733_05	10928726_00	438
Versuch16	Schwenklager	10528393_02	10945704_04	255
Versuch17	Hinterachsträger	11093034_02	10569597_02	476
Versuch17	Radlager	11340304_02	10335470_03	302
Versuch17	Schwenklager	10284749_04	10600463_04	312
Versuch17	Schwenklager	11178653_05	10618098_02	388
Versuch18	Hinterachsträger	10569597_02	10900480_03	456
Versuch18	Lenksäule	10666121_04	10350585_04	376
Versuch18	Schwenklager	10335475_03	10284749_04	289
Versuch18	Stützlager	10907381_02	10396872_05	319
Versuch18	Stützlager	10907382_02	10396875_03	257
Versuch20	Hinterachsträger	10398997_05	10494733_05	418
Versuch20	Schwenklager	10335475_03	10618098_02	285
Versuch20	Schwenklager	10528393_02	10335475_03	233
Versuch20	Stützlager	10337337_02	10396872_05	240
Versuch20	Stützlager	10337339_02	10396875_03	231

Versuch	Komponente	Dokument 1	Dokument 2	# Match
Versuch21	Hinterachsträger	11093034_02	10521287_05	574
Versuch21	Radlager	11340304_02	10335470_03	302
Versuch29	Hinterachsträger	11093034_02	10398997_05	494
Versuch29	Hinterachsträger	11093034_02	10001023_02	487
Versuch29	Radlager	11340304_02	10335470_03	302
Versuch29	Stützlager	10337337_02	10001041_02	293
Versuch29	Stützlager	10337337_02	11136245_03	240
Versuch32	Lenkspindel	10365946_06	10403829_05	203
Versuch32	Lenkspindel	10403834_05	10403836_05	194
Versuch32	Radlager	10001093_04	10001185_03	121
Versuch32	Radlager	10284827_05	10313543_05	165
Versuch32	Radlager	10313541_06	10284748_04	165
Versuch32	Stoßdämpfer	10337344_03	10337345_03	298
Versuch32	Stützlager	10396872_05	10396875_03	245
Versuch32	Stützlager	10907382_02	10907381_02	262
Versuch32	Stützlager	11304245_01	11304246_01	266
SUMME	**7**			**13546**

Literaturverzeichnis

Ahrens, G. (**2000)**: Das Erfassen und Handhaben von Produktanforderungen - Methodische Voraussetzungen und Anwendung in der Praxis. Dissertation. Technische Universität Berlin, Berlin.

Alabdulkareem, F.; **Cercone**, N.; **Liaskos**, S. (**2015)**: Goal and Preference Identification through natural language. In: Proceedings 2015 IEEE 23rd International Requirements Engineering Conference (RE). Ottawa, ON, Canada, 24.08.2015 - 28.08.2015. Piscataway, NJ: IEEE Computer Society Press, S. 56–65.

Alders, K. (**2006)**: Komplexitäts- und Variantenmanagement der AUDI AG. In: Lindemann, U., Reichwald, R. und Zäh, M. F. (Hg.): Individualisierte Produkte — Komplexität beherrschen in Entwicklung und Produktion. Berlin, Heidelberg: Springer (VDI-Buch), S. 221–237.

AlGeddawy, T.; **ElMaraghy**, H. (**2013)**: Reactive design methodology for product family platforms, modularity and parts integration. In: *CIRP Journal of Manufacturing Science and Technology* 6 (1), S. 34–43. DOI: 10.1016/j.cirpj.2012.08.001.

Almefelt, L.; **Berglund**, F.; **Nilsson**, P.; **Malmqvist**, J. (**2006)**: Requirements management in practice - Findings from an empirical study in the automotive industry. In: *Res Eng Design* 17 (3), S. 113–134. DOI: 10.1007/s00163-006-0023-5.

Arora, C.; **Sabetzadeh**, M.; **Goknil**, A.; **Briand**, L. C.; **Zimmer**, F. (**2015)**: Change impact analysis for Natural Language requirements: An NLP approach. In: Proceedings 2015 IEEE 23rd International Requirements Engineering Conference (RE). Ottawa, ON, Canada, 24.08.2015 - 28.08.2015. Piscataway, NJ: IEEE Computer Society Press, S. 6–15.

Arweiler, S. (**2008)**: Berechnung semantischer Ähnlichkeit: Ein Vergleich zwischen WordNet und FrameNet. Bachelor Thesis. Universität des Saarlandes, Saarbrücken.

Atkinson, R. (**1999)**: Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. In: *International Journal of Project Management* 17 (6), S. 337–342.

Aust, H. (**2021)**: Das Zeitalter der Daten. Berlin, Heidelberg: Springer.

Baker, C. F.; **Fillmore**, C. J.; **Lowe**, J. B. (**1998)**: The Berkeley FrameNet Project. In: ACL'98/COLING'98: Proceedings of 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1. Montreal, Quebec, Canada, 10.08.1998-14.08.1998. Stroudsburg, PA, USA: Association for Computational Linguistics, S. 86–90.

Barr, A. (**2015)**: Google Mistakenly Tags Black People as 'Gorillas,' Showing Limits of Algorithms. In: *The Wall Street Journal*:01.07.2015.

Bauer, W. (**2016)**: Planung und Entwicklung änderungsrobuster Plattformarchitekturen. München: Dr. Hut (Produktentwicklung) zugl. Dissertation Technische Universität München 2016.

Baumgart, I. (**2016)**: Requirements Engineering. In: Lindemann, U. (Hg.): Handbuch Produktentwicklung. München: Carl Hanser Verlag, S. 425–454.

Bender, B.; **Gericke**, K. (**2016)**: Entwicklungsprozesse. In: Lindemann, U. (Hg.): Handbuch Produktentwicklung. München: Carl Hanser Verlag, S. 401–424.

Berry, D. M.; **Kamsties**, E.; **Krieger**, M. M. (**2003)**: From Contract Drafting to Software Specification: Linguistic Sources of Ambiguity - A Handbook. Online verfügbar unter https://cs.uwaterloo.ca/~dberry/handbook/ambiguityHandbook.pdf, zuletzt geprüft am 10.01.2022.

Blanche, E. E. (**1957)**: Applying New Electronic Computers To Traffic and Highway Problems. In: *Traffic Quarterly* (11), S. 406–416.

Blees, C. (**2011)**: Eine Methode zur Entwicklung modularer Produktfamilien. Dissertation. Technische Universität Hamburg-Harburg, Hamburg.

BMW AG (Hg.) (**2021)**: BMW iX - Fahrwerk (Juni 2021) - ID: P90423664. Online verfügbar unter https://www.press.bmwgroup.com/deutschland/photo/detail/P90423664/bmw-ix-fahrwerk-juni-2021.

Boehm, O. (**o.J.)**: Anforderungsmanagement für Geschäftsanwendungen - Ein integriertes Analyse-Synthese-Konzept. White Paper. Hg. v. Fraunhofer-Institut ISST. Berlin (COMPARC insight). Online verfügbar unter http://www.isst.fraunhofer.de/content/dam/isst/de/documents/Publikationen/StudienundWhitePaper/Whitepaper_Anforderungsmanagement-fuer-Geschaeftsanwendungen.pdf, zuletzt geprüft am 29.09.2016.

Böttrich, M. (**2015**): Entwicklung von Produktfamilien in den frühen Phasen des Produktentstehungsprozesses - Methode zur effizienten Konfigurierung, Konstruktion und Analyse. Dissertation. Technische Universität Dresden, Dresden.

Boutkova, E. (**2014**): Variantenmanagement in Anforderungsdokumenten für hochkomplexe und variantenreiche Produkte. Dissertation. Universität Ulm, Ulm.

Brodersen, J.; **Siersma**, V. D. (**2013**): Long-term psychosocial consequences of false-positive screening mammography. In: *Annals of family medicine* 11 (2), S. 106–115. DOI: 10.1370/afm.1466.

Bühne, S.; **Lauenroth**, K.; **Pohl**, K. (**2004**): Anforderungsmanagement in der Automobilindustrie: Variabilität in Zielen, Szenarien und Anforderungen. In: Dadam, P. und Reichert, M. (Hg.): INFORMATIK 2004 - Informatik verbindet. 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI). Ulm, 20.09.2004-24.09.2004. Bonn: Ges. für Informatik (GI) (Lecture notes in informatics. Proceedings, 51), S. 23–27.

Bühne, S.; **Lauenroth**, K.; **Pohl**, K. (**2005**): Modelling requirements variability across product lines. In: Proceedings 13th IEEE International Requirements Engineering Conference RE2005. La Sorbonne, France, 29.08 2005- 02.09.2005. Los Alamitos, California: IEEE Computer Society Press, S. 41–50.

Burghardt, M. (**2013**): Einführung in Projektmanagement - Definition, Planung, Kontrolle, Abschluss. Erlangen: Publicis Publishing.

Buxmann, P.; **Schmidt**, H. (**2019**): Künstliche Intelligenz. Berlin, Heidelberg: Springer Gabler.

Caesar, C. (**1991**): Kostenorientierte Gestaltungsmethodik für variantenreiche Serienprodukte - Variant Mode and Effects Analysis (VMEA). Aachen: VDI-Verlag zugl. Dissertation RWTH Achen 1991.

Carstensen, K.-U.; **Ebert**, C.; **Ebert**, C. et al. (Hg.) (**2010**): Computerlinguistik und Sprachtechnologie - Eine Einführung. 3., überarb. und erw. Aufl. Heidelberg: Spektrum Akad. Verl. (Spektrum Lehrbuch). Online verfügbar unter http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10351853.

Chen, D.; **Manning**, C. D. (**2014**): A Fast and Accurate Dependency Parser using Neural Networks. In: Moschitti, A., Pang, B. und Daelemans, W. (Hg.): Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar, 25.10.2014-29.10.2014. Stroudsburg, PA, USA: Association for Computational Linguistics, S. 740–750.

Cheng, B. H. C.; **Atlee**, J. M. (**2007**): Research Directions in Requirements Engineering. In: Briand, L. C. und Wolf, A. L. (Hg.): FoSE 2007: Future of Software Engineering. Minneapolis, 23.05.2007-25.05.2007. Los Alamitos, Calif.: IEEE Computer Society Press, S. 285–303.

Chernak, Y. (**2012**): Requirements Reuse: The State of the Practice. In: 2012 IEEE International Conference on Software Science, Technology and Engineering. Herzlia, Israel, 12.06.2012 - 13.06.2012: IEEE Computer Society Press, S. 46–53.

Chollet, F. (**2017**): Deep learning with Python. Shelter Island, NY: Manning (Safari Tech Books Online).

Chunks (**2012**). In: Seel, N. M. (Hg.): Encyclopedia of the Sciences of Learning. Boston, MA: Springer, S. 540–541.

Cilibrasi, R. L.; **Vitanyi**, P. M. (**2007**): The Google Similarity Distance. In: *IEEE Trans. Knowl. Data Eng.* 19 (3), S. 370–383. DOI: 10.1109/TKDE.2007.48.

Clark, K. B.; **Fujimoto**, T. (**1992**): Automobilentwicklung mit System - Strategie, Organisation und Management in Europa, Japan und USA. Frankfurt/Main, New York: Campus Verlag.

Collier, D. A. (**1981**): The Measurement and Operating Benefits of Component Part Commonality. In: *Decision Sciences* 12 (1), S. 85–96. DOI: 10.1111/j.1540-5915.1981.tb00063.x.

Cortical.io (**2017**): Semantic Folding: a new model for natural language understanding. YouTube. Online verfügbar unter https://youtu.be/HLuRQKzYbb8.

Covington, M. A. (**2001**): A Fundamental Algorithm for Dependency Parsing. In: ACM (Hg.): 39th Annual ACM Southeast Conference. Athens, Georgia, USA, 16.03.2001-17.03.2001.

Cybulski, J. L.; **Reed**, K. (**1998**): Computer-assisted analysis and refinement of informal software requirements documents. In: Proceedings 1998 Asia Pacific Software Engineering Conference (Cat. No.98EX240). Taipei, Taiwan, 02.12.1998-04.12.1998: IEEE Computer Society Press, S. 128–135.

Cybulski, J. L.; **Reed**, K. (**2000**): Requirements Classification and Reuse: Crossing Domain Boundaries. In: Frakes, W. B. (Hg.): Software Reuse: Advances in Software Reusability. 6th International Conference, ICSR-6. Vienna, Austria,

27.06.2000-29.06.2000. Berlin, Heidelberg: Springer (Lecture Notes in Computer Science, 1844), S. 190–210.

Cyriaks, H.; **Lohmann**, S.; **Stolz**, H.; **Velioglu**, V.; **Ziegler**, J. (**2007**): Semantische Aufbereitung von Dokumentenbeständen zur Gewinnung anforderungsrelevanter Informationen. In: Auer, S., Bizer, C. et al. (Hg.): The social semantic web. Proceedings of the 1st Conference on Social Semantic Web (CSSW). Leipzig, 26.09.2007-28.09.2007. Bonn: Ges. für Informatik (GI) (GI-Edition : Proceedings, 113), S. 139–146.

Czichos, H.; **Hennecke**, M. (Hg.) (**2007**): HÜTTE - Das Ingenieurwissen. Akademischer Verein Hütte e.V. 33. Aufl. Berlin, Heidelberg: Springer.

Darimont, R.; **Zhao**, W.; **Ponsard**, C.; **Michot**, A. (**2017**): Deploying a Template and Pattern Library for Improved Reuse of Requirements Across Projects. In: Proceedings 2017 IEEE 25th International Requirements Engineering Conference. Lisbon, Portugal, 04.09.2017-08.09.2017. Piscataway, NJ: IEEE Computer Society Press, S. 456–457.

Das, S. K.; **Swain**, A. K. (**2021**): An Ontology-Based Framework for Decision Support in Assembly Variant Design. In: *Journal of Computing and Information Science in Engineering* 21 (2). DOI: 10.1115/1.4048127.

Dehlinger, J.; **Lutz**, R. R. (**2008**): Supporting requirements reuse in multi-agent system product line design and evolution. In: IEEE International Conference on Software Maintenance (ICSM). Beijing, China, 28.09.2008 - 04.10.2008: IEEE Computer Society Press, S. 207–216.

Dellanoi, R. (**2006**): Kommunalitäten bei der Entwicklung variantenreicher Produktfamilien. Dissertation. Universität St. Gallen, St. Gallen.

Dey, S.; **Leey**, S.-W. (**2015**): From Requirements Elicitation to Variability Analysis using Repertory Grid: A Cognitive Approach. In: Proceedings 2015 IEEE 23rd International Requirements Engineering Conference (RE). Ottawa, ON, Canada, 24.08.2015 - 28.08.2015. Piscataway, NJ: IEEE Computer Society Press, S. 46–55.

Dölle, C. (**2018**): Projektsteuerung in der Produktentwicklung mittels Predictive Analytics - Development project management by aid of predictive analytics. 1. Auflage. Aachen: Apprimus Verlag (Produktionssystematik, Band 2/2018).

Dwarakanath, A.; **Ramnani**, R. R.; **Sengupta**, S. (**2013**): Automatic Extraction of Glossary Terms from Natural Language Requirements. In: Proceedings 21st

IEEE International Requirements Engineering Conference (RE). Rio de Janeiro, Brazil, 15.07.2013 - 19.07.2013. Piscataway, NJ: IEEE Computer Society Press, S. 314–319.

Ebert, C. (**2010**): Systematisches Requirements Engineering - Anforderungen ermitteln, spezifizieren, analysieren und verwalten. 3., aktualisierte und erw. Aufl. Heidelberg: Dpunkt-Verl.

Ehrlenspiel, K.; **Kiewert**, A.; **Lindemann**, U. (**2007**): Kostengünstig entwickeln und konstruieren - Kostenmanagement bei der integrierten Produktentwicklung ; mit 143 Tabellen. 6., überarb. und korrigierte Aufl. Berlin, Heidelberg: Springer (VDI-/Buch]).

Ehrlenspiel, K.; **Kiewert**, A.; **Lindemann**, U.; **Mörtl**, M. (**2014**): Kostengünstig entwickeln und konstruieren - Kostenmanagement bei der integrierten Produktentwicklungen. 7. Aufl. Berlin [u.a.]: Springer (VDI).

Ehrlenspiel, K.; **Meerkamm**, H. (**2013**): Integrierte Produktentwicklung - Denkabläufe, Methodeneinsatz, Zusammenarbeit. 5. Auflage. München: Carl Hanser Verlag.

Eigner, M.; **Stelzer**, R. (**2013**): Product-Lifecycle-Management - Ein Leitfaden für Product-Development und Life-Cycle-Management. 2., neu bearb. Aufl. Berlin, Heidelberg: Springer (VDI).

ElMaraghy, H.; **AlGeddawy**, T. (**2012**): New dependency model and biological analogy for integrating product design for variety with market requirements. In: *Journal of Engineering Design* 23 (10-11), S. 722–745. DOI: 10.1080/09544828.2012.709607.

Elser, H.; **Wangerow**, K.; **Ngo**, Q. H.; **Schmitt**, R. H. (**2021**): Six Sigma. In: Pfeifer, T. und Schmitt, R. (Hg.): Handbuch Qualitätsmanagement. 7., überarbeitete Auflage. München: Hanser, S. 262–291.

Fawcett, T. (**2006**): An introduction to ROC analysis. In: *Pattern Recognition Letters* 27 (8), S. 861–874. DOI: 10.1016/j.patrec.2005.10.010.

Fellbaum, C. (**1998**): A Semantic Network of English: The Mother of All WordNets. In: *Computers and the Humanities* 32 (2/3), S. 209–220.

Ferrari, A.; **dell'Orletta**, F.; **Spagnolo**, G. O.; **Gnesi**, S. (**2014**): Measuring and Improving the Completeness of Natural Language Requirements. In: Hutchison, D., Kanade, T. et al. (Hg.): Requirements Engineering: Foundation for Software

Quality. 20th International Working Conference, REFSQ 2014. Essen, Germany, 07.04.2014-10.04.2014. Cham: Springer (Lecture Notes in Computer Science, 8396), S. 23–38.

Ferrari, A.; **Gori**, G.; **Rosadini**, B.; **Trotta**, I.; **Bacherini**, S.; **Fantechi**, A.; **Gnesi**, S. (**2018)**: Detecting requirements defects with NLP patterns: an industrial experience in the railway domain. In: *Empir Software Eng* 23 (6), S. 3684–3733. DOI: 10.1007/s10664-018-9596-7.

Firchau, N. L. (**2003)**: Variantenoptimierende Produktgestaltung. 1. Aufl. Braunschweig: Cuvillier zugl. Dissertation Technische Universität Carolo-Wilhelmina zu Braunschweig 2003.

Franch, X. (**2013)**: Software requirement patterns. In: 35th International Conference on Software Engineering (ICSE). San Francisco, CA, USA, 18.05.2013 - 26.05.2013: IEEE Computer Society Press, S. 1499–1501.

Franke, H.-J.; **Hesselbach**, J.; **Huch**, B.; **Firchau**, N. L. (**2002)**: Variantenmanagement in der Einzel- und Kleinserienfertigung. München [u.a.]: Carl Hanser Verlag.

Gabrilovich, E.; **Markovitch**, S. (**2007)**: Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI 2007. Hyderabad; India, 06.01.2007 - 12.01.2007, S. 1606–1611.

Gacitua, R.; **Sawyer**, P.; **Gervasi**, V. (**2010)**: On the Effectiveness of Abstraction Identification in Requirements Engineering. In: Proceedings of the 18th IEEE International Requirements Engineering Conference RE2010. Sydney, New South Wales, Australia, 27.09.2010 - 01.10. 2010. Los Alamitos, California: IEEE Computer Society Press, S. 5–14.

Garcia, I.; **Pacheco**, C. (**2016)**: RRXRC: a functional extension for incorporating the reuse of requirements in Rational RequisitePro®. In: *IEEE Latin Am. Trans.* 14 (9), S. 4181–4186. DOI: 10.1109/TLA.2016.7785950.

Gebauer, M. (**2001)**: Kooperative Produktentwicklung auf der Basis verteilter Anforderungen. Aachen: Shaker (Forschungsberichte aus dem Institut für Rechneranwendung in Planung und Konstruktion der Universität Karlsruhe, Bd. 2001,3) zugl. Dissertation Karlsruhe, Univ. 2000.

Gebhardt, N. (**2020**): Methodische Entwicklung von Visualisierungen als Arbeitswerkzeuge in der Produktentwicklung. Hamburg: Springer zugl. Dissertation Technischen Universität Hamburg 2019.

Gebhardt, N.; **Kruse**, M.; **Krause**, D. (**2016**): Gleichteile-, Modul- und Plattformstrategie. In: Lindemann, U. (Hg.): Handbuch Produktentwicklung. München: Carl Hanser Verlag, S. 111–149.

Gelhausen, T. (**2010**): Modellextraktion aus natürlichen Sprachen - Eine Methode zur systematischen Erstellung von Domänenmodellen: KIT scientific publishing zugl. Dissertation Karlsruher Institut für Technologie KIT 2010.

Gembrys, S.-N. (**1998**): Ein Modell zur Reduzierung der Variantenvielfalt in Produktionsunternehmen. Berlin: IPK (Berichte aus dem Produktionstechnischen Zentrum Berlin) zugl. Dissertation Technische Universität Berlin 1998.

Gipp, B. (**2014**): Citation-based Plagiarism Detection - Detecting Disguised and Cross-language Plagiarism using Citation Pattern Analysis. Wiesbaden: Springer Fachmedien Wiesbaden.

Gleich, B.; **Creighton**, O.; **Kof**, L. (**2010**): Ambiguity Detection: Towards a Tool Explaining Ambiguity Sources. In: Hutchison, D., Kanade, T. et al. (Hg.): Requirements Engineering: Foundation for Software Quality. 16th International Working Conference, REFSQ 2010. Essen, 30.06.2010–02.07.2010. Berlin, Heidelberg: Springer (Lecture Notes in Computer Science, 6182), S. 218–232.

Goldin, L.; **Berry**, D. M. (**2015**): Reuse of requirements reduced time to market at one industrial shop - A case study. In: *Requirements Eng* 20 (1), S. 23–44. DOI: 10.1007/s00766-013-0182-7.

Gomaa, W. H.; **Fahmy**, A. A. (**2013**): A survey of text similarity approaches. In: *International Journal of Computer Applications* 68 (13), S. 13–18.

Goodfellow, I.; **Bengio**, Y.; **Courville**, A. (**2018**): Deep Learning - Das umfassende Handbuch : Grundlagen, aktuelle Verfahren und Algorithmen, neue Forschungsansätze. 1. Auflage. Frechen: mitp.

Grande, M. (**2014**): 100 Minuten für Anforderungsmanagement - Kompaktes Wissen nicht nur für Projektleiter und Entwickler. 2., aktual. Aufl. Wiesbaden: Springer Vieweg.

Grimm, A. (**2011**): Prozessorientierter Umgang mit Anforderungen für die kundenspezifische Auftragsabwicklung. München: Gabler Verlag zugl. Dissertation Ludwig-Maximillians-Universität München 2010.

Grosz, B. J.; **Joshi**, A. K.; **Weinstein**, S. (**1995**): Centering: A Framework for Modelling the Local Coherence of Discourse. In: *Comput. Linguistics* 21 (2), S. 203–225. DOI: 10.21236/ada324949.

Grotkamp, S.: Bewertung von Produktstrukturkonzepten im Variantenmanagement. 1. Auflage. Braunschweig (Bericht / Institut für Konstruktionstechnik, Technische Universität Braunschweig, Nr. 76) zugl. Dissertation Technische Universität Braunschweig 2009.

Gülke, T. (**2014**): Erweiterung des Anforderungsmanagement-Fokus - Von Produkten zu Prozessen. 1. Aufl. Aachen: Shaker (Aachener Informatik Berichte Software Engineering, 18) zugl. Dissertation RWTH Aachen 2014.

Guodong, Y.; **Yu**, Y.; **Aijun**, L. (**2015**): Joint optimization of complex product variant design responding to customer requirement changes. In: *IFS* 30 (1), S. 397–408. DOI: 10.3233/IFS-151764.

Gussen, L. C.; **Kukulies**, J.; **Sohnius**, F.; **Schmitt**, R. H. (**2021**): Customer Insights in der Produktentwicklung. In: Pfeifer, T. und Schmitt, R. (Hg.): Handbuch Qualitätsmanagement. 7., überarbeitete Auflage. München: Hanser, S. 514–533.

Haberhauer, H.; **Bodenstein**, F. (**2007**): Maschinenelemente - Gestaltung, Berechnung, Anwendung. 14. Aufl. Berlin, Heidelberg: Springer (Springer-Lehrbuch).

Hamp, B.; **Feldweg**, H. (**1997**): GermaNet - a Lexical-Semantic Net for German. In: Proceedings of the ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications. Madrid, S. 9–15.

Harms, E. (**2009**): Änderungs- und Konfigurationsmanagement unter Berücksichtigung von Verwendungsinstanzen: Arbeitsmethoden für integrierte Produktmodelle im Rahmen des Produkt-Lebenszyklus-Managements der Automobilindustrie. Dissertation. Universität Karlsruhe, Karlsruhe.

Hasenpusch, J. (**2018**): Methodik zur Beurteilung eigenschaftsoptimierter Karosseriekonzepte in Mischbauweise. Wiesbaden: Springer Fachmedien Wiesbaden zugl. Dissertation Technische Universität Carolo-Wilhelmina zu Braunschweig 2018.

Hauksdottir, D.; **Vermehren**, A.; **Savolainen**, J. (**2012)**: Requirements reuse at Danfoss. In: Proceedings 20th IEEE International Requirements Engineering Conference (RE). Chicago, Illinois, USA, 24.09.2012-28.09.2012. Piscataway, NJ: IEEE Computer Society Press, S. 309–314.

Hawkins, J. (**2006)**: Die Zukunft der Intelligenz - Wie das Gehirn funktioniert, und was Computer davon lernen können. Unter Mitarbeit von Blakeslee, S. Deutsche Erstausgabe. Reinbek bei Hamburg: Rowohlt Taschenbuch Verlag (rororo Science, 62167).

Hawkins, J.; **Ahmad**, S.; **Dubinsky**, D. (**2011)**: Hierarchical Temporal Memory - including HTM Cortical Learning Algorithms. Version 0.2.1. Hg. v. Numenta.

Hawkins, J.; **Blakeslee**, S. (**2005)**: On intelligence. 1. Aufl. New York, NY: Owl Books.

Hayes, J. H.; **Dekhtyar**, A.; **Sundaram**, S. K. (**2005)**: Improving After-the-Fact Tracing and Mapping: Supporting Software Quality Predictions. In: *IEEE Softw.* 22 (6), S. 30–37. DOI: 10.1109/MS.2005.156.

Hecker, D.; **Döbel**, I.; **Petersen**, U.; **Rauschert**, A.; **Schmitz**, V.; **Voss**, A. (**2017)**: Zukunftsmarkt Künstliche Intelligenz - Potenziale und Anwendungen. Hg. v. Fraunhofer-Allianz Big Data. Sankt Augustin.

Heina, J. (**1999)**: Variantenmanagement - Kosten-Nutzen-Bewertung zur Optimierung der Variantenvielfalt. Gabler Edition Wissenschaft. Wiesbaden, s.l.: Deutscher Universitätsverlag (Springer eBook Collection Humanities, Social Science).

Heißing, B. (Hg.) (**2007)**: Fahrwerkhandbuch - Grundlagen, Fahrdynamik, Komponenten, Systeme, Mechatronik, Perspektiven. 1. Aufl. Wiesbaden: Vieweg (Aus dem Programm Kraftfahrzeugtechnik).

Heißing, B.; **Ersoy**, M.; **Gies**, S. (**2013)**: Fahrwerkhandbuch - Grundlagen, Fahrdynamik, Komponenten, Systeme. 4., überarbeitete und ergänzte Aufl. Wiesbaden: Springer Vieweg.

Heumesser, N.; **Houdek**, F. (**2003)**: Towards systematic recycling of systems requirements. In: Proceedings of the 25th International Conference on Software Engineering ICSE '03. Portland, OR, USA, 3.05.2003-10.05.2003, S. 512–519.

Heumesser, N.; **Houdek**, F. (**2004**): Experiences in managing an automotive requirements engineering process. In: Proceedings 12th IEEE International Requirements Engineering Conference RE2004. Kyoto, Japan, 06.09.2004-11.09.2004. Piscataway, NJ: IEEE Computer Society Press, S. 302–307.

Hichert, R. (**1985**): Probleme der Vielfalt Teil 1: Soll man auf Exoten verzichten? In: *Werkstattstechnik: wt ; Produktion und Management* 75, S. 235–237.

Hobbs, J. R. (**1978**): Resolving pronoun references. In: *Lingua* 44 (4), S. 311–338. DOI: 10.1016/0024-3841(78)90006-2.

Hochreiter, S.; **Schmidhuber**, J. (**1997**): Long short-term memory. In: *Neural computation* 9 (8), S. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

Hood, C.; **Mühlbauer**, S.; **Rupp**, C.; **Versteegen**, G. (**2007**): iX-Studie Anforderungsmanagement. 2. völlig überarbeitete Aufl. Hannover: Heise Zeitschriften Verlag.

Hood, C.; **Wiebel**, R. (**2005**): Optimieren von Requirements Management & Engineering. Berlin, Heidelberg, New York: Springer (Xpert.press).

Hoppe, T. (**2020**): Semantische Suche - Grundlagen und Methoden semantischer Suche von Textdokumenten. [S.l.]: Morgan Kaufmann.

Hunt, E. (**2016**): Tay, Microsoft's AI chatbot, gets a crash course in racism from Twitter. In: *The Guardian*:24.03.2016.

Hüttenrauch, M.; **Baum**, M. (**2008**): Effiziente Vielfalt - Die dritte Revolution in der Automobilindustrie. Berlin u.a.: Springer.

Huysegoms, T.; **Snoeck**, M.; **Dedene**, G.; **Goderis**, A.; **Stumpe**, F. (**2013**): Visualizing Variability Management in Requirements Engineering through Formal Concept Analysis. In: *Procedia Technology* 9, S. 189–199. DOI: 10.1016/j.protcy.2013.12.021.

Ilie, D. (**2013**): Systematisiertes Ziele- und Anforderungsmanagement in der Fahrzeugentwicklung. 1. Aufl. München: Dr. Hut (Produktentwicklung) zugl. Dissertation Technische Universität München 2013.

Isermann, R. (Hg.) (**2006**): Fahrdynamik-Regelung - Modellbildung, Fahrerassistenzsysteme, Mechatronik. 1. Aufl. Wiesbaden: Vieweg (ATZ/MTZ-Fachbuch).

Jeschke, A. (**1997**): Beitrag zur wirtschaftlichen Bewertung von Standardisierungs-Maßnahmen in der Einzel- und Kleinserienfertigung durch die Konstruktion.

Dissertation. Technische Universität Carolo-Wilhelmina zu Braunschweig, Braunschweig.

Jochem, R. (**2010**): Was kostet Qualität? - Wirtschaftlichkeit von Qualität ermitteln. Unter Mitarbeit von Dietmüller, T., Geers, D., Giebel, M. und Landgraf, K. München: Carl Hanser Verlag.

Jörg, M. A.J. (**2005**): Ein Beitrag zur ganzheitlichen Erfassung und Integration von Produktanforderungen mit Hilfe linguistischer Methoden. Karlsruhe: Shaker (Forschungsberichte aus dem Institut für Rechneranwendung in Planung und Konstruktion der Universität Karlsruhe, Bd. 2005,3) zugl. Dissertation Universität Karlsruhe 2005.

Kamiske, G. F. (**2008**): Qualitätsmanagement - Eine multimediale Einführung. 4., aktualisierte Aufl. München: Fachbuchverl. Leipzig.

Kao, A.; **Poteet**, S. R. (**2007**): Natural Language Processing and Text Mining. London: Springer.

Karatas, E. K.; **Iyidir**, B.; **Birturk**, A. (**2014**): Ontology-Based Software Requirements Reuse: Case Study in Fire Control Software Product Line Domain. In: 2014 IEEE International Conference on Data Mining Workshop (ICDMW). Shenzhen, China, 14.12.2014 - 14.12.2014: IEEE Computer Society Press, S. 832–839.

Kersten, W. (**2002**): Vielfaltsmanagement - Integrative Lösungsansätze zur Optimierung und Beherrschung der Produkte und Teilevielfalt. München: TCW Transfer-Centrum (TCW-Report, 31).

Kesper, H. (**2012**): Gestaltung von Produktvariantenspektren mittels matrixbasierter Methoden. Dissertation. Technische Universität München, München.

Kickermann, H. (**1995**): Rechnerunterstützte Verarbeitung von Anforderungen im methodischen Konstruktionsprozeß. Dissertation. Technische Universität Carolo-Wilhelmina zu Braunschweig, Braunschweig.

Kim, J.; **Kim**, M.; **Yang**, H.; **Park**, S. (**2004**): A Method and Tool Support for Variant Requirements Analysis: Goal and Scenario Based Approach. In: 11th Asia-Pacific Software Engineering Conference. Busan, Korea, 30.10.2004-03.11.2004: IEEE Computer Society Press, S. 168–175.

Kirner, K. G.M. (**2014**): Zusammenhang zwischen Leistung in der Produktentwicklung und Variantenmanagement - Einflussmodell und Analysemethode. 1. Aufl.

München: Dr. Hut (Produktentwicklung) zugl. Dissertation Technische Universität München 2014.

Kläger, R. (**1993)**: Modellierung von Produktanforderungen als Basis für Problemlösungsprozesse in intelligenten Konstruktionssystemen. Als Ms. gedr. Karlsruhe: Shaker (Reihe Konstruktionstechnik) zugl. Dissertation Universität Karlsruhe 1993.

Kloos, K.-H.; **Thomala**, W. (**2007)**: Schraubenverbindungen - Grundlagen, Berechnung, Eigenschaften, Handhabung. 5. Aufl. Berlin, Heidelberg: Springer.

Klute, S.; **Kolbe**, C.; **Refflinghaus**, R. (**2011)**: Requirements Management – a Premise for adequate Life Cycle Design. In: Hesselbach, J. und Herrmann, C. (Hg.): Glocalized Solutions for Sustainability in Manufacturing. Berlin, Heidelberg: Springer, S. 537–542.

Knauss, E.; **Ott**, D. (**2014)**: (Semi-) automatic Categorization of Natural Language Requirements. In: Hutchison, D., Kanade, T. et al. (Hg.): Requirements Engineering: Foundation for Software Quality. 20th International Working Conference, REFSQ 2014. Essen, Germany, 07.04.2014-10.04.2014. Cham: Springer (Lecture Notes in Computer Science, 8396), S. 39–54.

Kolb, P. (**2009)**: Experiments on the difference between semantic similarity and relatedness. In: Jokinen, K. und Bick, E. (Hg.): Proceedings of the 17th Nordic Conference of Computational Linguistics (NODALIDA 2009). Odense, Denmark, 14.05.2009-16.05.2009: Northern European Association for Language Technology (NEALT) (NEALT Proceedings Series, 4), S. 81–88.

Kolbe, C. (**2014)**: Wissensbasiertes System zur Anforderungsanalyse bei der Anlagenplanung am Beispiel intralogistischer Anlagen. Kassel: kassel university press (Kasseler Schriftenreihe Qualitätsmanagement, Band 5) zugl. Dissertation Technische Universität Dortmund 2013.

Kolbe, C.; **Refflinghaus**, R. (**2011)**: Knowledge-based Concretization of Requirements in Preparation for AHP. In: Berenbach, B., Daneva, M. et al. (Hg.): 17th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2011). Proceedings of the REFSQ 2011 Workshops REEW, EPICAL and RePriCo, the REFSQ 2011 Empirical Track (Empirical Live Experiment and Empirical Research Fair), and the REFSQ 2011 Doctoral Symposium. Essen, Germany, 28.03.2011-30.03.2011: ICB (ICB-Research Report, 44), 97-109.

Konrad, S.; **Cheng**, B. (**2002**): Requirements patterns for embedded systems. In: Proceedings IEEE Joint International Conference on Requirements Engineering. Essen, Germany, 09.09.2002-13.09.2002: IEEE Computer Society Press, S. 127–136.

Körner, S. J. (**2014**): RECAA - Werkzeugunterstützung in der Anforderungserhebung. Dissertation. Karlsruher Institut für Technologie KIT, Karlsruhe.

Kotler, P.; **Bliemel**, F. (**2001**): Marketing-Management - Analyse, Planung und Verwirklichung. 10., überarb. und aktualisierte Aufl. Stuttgart: Schäffer-Poeschel.

Kubica, S. (**2007**): Variantenmanagement modellbasierter Funktionssoftware mit Software-Produktlinien. 40 Bände. Erlangen (Arbeitsberichte des Instituts für Informatik, 4) zugl. Dissertation Friedrich-Alexander-Universität Erlangen-Nürnberg 2007.

Künne, B. (Hg.) (**2007**): Köhler/Rögnitz Maschinenteile 1. 10. Aufl. Wiesbaden: Teubner.

Landauer, T. K.; **Dumais**, S. T. (**1997**): A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. In: *Psychological Review* 104 (2), S. 211–240. DOI: 10.1037/0033-295X.104.2.211.

Landauer, T. K.; **Foltz**, P. W.; **Laham**, D. (**1998**): An introduction to latent semantic analysis. In: *Discourse Processes* 25 (2-3), S. 259–284. DOI: 10.1080/01638539809545028.

Landgraf, K. (**2011**): Requirements Management & Engineering - ein Überblick. In: Jochem, R. und Landgraf, K. (Hg.): Anforderungsmanagement in der Produktentwicklung. Komplexität reduzieren, Prozesse optimieren, Qualität sichern. 1. Aufl. Düsseldorf: Symposion Publ, S. 15–33.

Leuser, J.; **Ott**, D. (**2010**): Tackling Semi-automatic Trace Recovery for Large Specifications. In: Hutchison, D., Kanade, T. et al. (Hg.): Requirements Engineering: Foundation for Software Quality. 16th International Working Conference, REFSQ 2010. Essen, 30.06.2010–02.07.2010. Berlin, Heidelberg: Springer (Lecture Notes in Computer Science, 6182), S. 203–217.

Li, Y.; **McLean**, D.; **Bandar**, Z. A.; **O'Shea**, J. D.; **Crockett**, K. (**2006**): Sentence similarity based on semantic nets and corpus statistics. In: *IEEE Trans. Knowl. Data Eng.* 18 (8), S. 1138–1150. DOI: 10.1109/TKDE.2006.130.

Lindemann, U. (Hg.) (**2016)**: Handbuch Produktentwicklung. München: Carl Hanser Verlag.

Lingnau, V. (**1994)**: Variantenmanagement - Produktionsplanung im Rahmen einer Produktdifferenzierungsstrategie. Berlin: Schmidt (Betriebswirtschaftliche Studien, 58) zugl. Dissertation Techn. Univ. Berlin 1994.

Lund, K.; **Burgess**, C. (**1996)**: Producing high-dimensional semantic spaces from lexical co-occurrence. In: *Behavior Research Methods, Instruments, & Computers* 28 (2), S. 203–208. DOI: 10.3758/BF03204766.

Mahmoud, A.; **Niu**, N. (**2010)**: An experimental investigation of reusable requirements retrieval. In: IEEE International Conference on Information Reuse & Integration (2010 IRI). Las Vegas, NV, USA, 04.08.2010 - 06.08.2010: IEEE Computer Society Press, S. 330–335.

Mamrot, M.; **Marchlewitz**, S.; **Winzer**, P. (**2013)**: Changeability of requirements — Usable indicators for product development. In: 2013 IEEE International Conference on Mechatronics (ICM). Vicenza, 27.02.2013 - 01.03.2013: IEEE Computer Society Press, S. 350–355.

Manicke, O. (**2012)**: Durchgängiges Variantenmanagement zur Komplexitätsbeherrschung im Entwicklungsprozess mechatronischer Fahrzeugfunktionen. Renningen: Expert-Verl. (Schriftenreihe des Lehrstuhls Fahrzeugmechatronik der TU Dresden, 5).

Manning, C. D.; **Raghavan**, P.; **Schütze**, H. (**2009)**: Introduction to information retrieval. Reprinted. Cambridge: Cambridge Univ. Press.

Mannion, M.; **Kaindl**, H. (**2008)**: Using parameters and discriminants for product line requirements. In: *Syst. Engin.* 11 (1), S. 61–80. DOI: 10.1002/sys.20086.

Mannion, M.; **Kaindl**, H.; **Wheadon**, J.; **Keepence**, B. (**1999)**: Reusing single system requirements from application family requirements. In: Boehm, B., Garlan, D. und Kramer, J. (Hg.): Proceedings of the 21st international conference on Software engineering - ICSE '99. Los Angeles, California, United States, 16.05.1999 - 22.05.1999. New York, New York, USA: ACM Press, S. 453–462.

Matschinsky, W. (**2007)**: Radführungen der Straßenfahrzeuge - Kinematik, Elasto-Kinematik und Konstruktion. 3. Aufl. Berlin, Heidelberg: Springer.

Matys, E. (**2018)**: Praxishandbuch Produktmanagement - Grundlagen und Instrumente. 7., aktualisierte und erweiterte Auflage. Frankfurt, New York: Campus Verlag.

Mayer-Bachmann, R. (**2008)**: Integratives Anforderungsmanagement - Konzept und Anforderungsmodell am Beispiel der Fahrzeugentwicklung. Karlsruhe: Universitätsverlag Karlsruhe (Reihe Informationsmanagement im Engineering Karlsruhe, Bd. 2, 2007) zugl. Dissertation Universität Karlsruhe 2007.

McCarthy, J.; **Minsky**, M. L.; **Rochester**, N.; **Shannon**, C. E. (**1955)**: A Proposal for the Dartmouth Summer Research Project On Artificial Intelligence. Hanover, New Hampshire, USA.

McClelland, J. L.; **Kawamoto**, A. H. (**1986)**: Mechanisms of Sentence Processing: Assigning Roles to Constituents. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 2: Psychological and Biological Models. Cambridge, MA, USA: MIT Press, S. 272–325.

Mei, H.; **Zhang**, W.; **Gu**, F. (**2003)**: A feature oriented approach to modeling and reusing requirements of software product lines. In: Proceedings 27th Annual International Computer Software and Applications Conference. COMPAC 2003. Dallas, TX, USA, 03.11.2003-06.11.2003: IEEE Computer Society Press, S. 250–256.

Meissner, M.; **Schorcht**, H.-J. (**2007)**: Metallfedern - Grundlagen, Werkstoffe, Berechnung, Gestaltung und Rechnereinsatz. 2. Aufl. Berlin, Heidelberg: Springer (VDI-/Buch]).

Meyer, M. H.; **Lehnerd**, A. P. (**1997)**: The power of product platforms - Building value and cost leadership. New York: Free Press.

Mich, L.; **Mariangela**, F.; **Pierluigi**, N. I. (**2004)**: Market research for requirements analysis using linguistic tools. In: *Requirements Eng* 9 (1), S. 40–56. DOI: 10.1007/s00766-003-0179-8.

Mikolov, T.; **Sutskever**, I.; **Chen**, K.; **Corrado**, G.; **Dean**, J. (**2013)**: Distributed representations of words and phrases and their compositionality. In: Burges, C., Bottou, L. et al. (Hg.): NIPS'13: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2. Lake Tahoe, Nevada, USA, 05.12.2013-10.12.2013: Curran Associates Inc., S. 3111–3119.

Monzon, A. (**2008**): A Practical Approach to Requirements Reuse in Product Families of On-Board Systems. In: Proceedings of the 16th IEEE International Requirements Engineering Conference RE2008. Barcelona, Spain, 8.09.2008 - 12.09.2008. Los Alamitos, California: IEEE Computer Society Press, S. 223–228.

Moore, G. E. (**1965**): Cramming more components onto integrated circuits. In: *Electronics* 38 (8), S. 114–117.

Muhs, D.; **Wittel**, H.; **Jannasch**, D.; **Voßiek**, J. (**2007**): Rooloff/Matek Maschinenelemente - Normung, Berechnung, Gestaltung. 18. Aufl. Wiesbaden: Vieweg (Studium).

Naz, H.; **Motla**, Y. H.; **Asghar**, S.; **Abbas**, M. A.; **Khatoon**, A. (**2013**): Effective usage of AI technique for requirement change management practices. In: 5th International Conference on Computer Science and Information Technology (CSIT). Amman, Jordan, 27.03.2013 - 28.03.2013: IEEE Computer Society Press, S. 121–125.

Ng, A.; **Soo**, K. (**2018**): Data Science - was ist das eigentlich?! - Algorithmen des maschinellen Lernens verständlich erklärt. Berlin, Heidelberg: Springer.

Niemann, G.; **Winter**, H.; **Höhn**, B.-R. (**2005**): Konstruktion und Berechnung von Verbindungen, Lagern, Wellen. 4. Aufl. Berlin: Springer (Maschinenelemente / G. Niemann, H. Winter, B.-R. Höhn, Bd. 1).

Niu, N.; **Savolainen**, J.; **Niu**, Z.; **Jin**, M.; **Cheng**, J.-R. C. (**2014**): A Systems Approach to Product Line Requirements Reuse. In: *IEEE Systems Journal* 8 (3), S. 827–836. DOI: 10.1109/JSYST.2013.2260092.

Ott, D. (**2012**): Defects in Natural Language Requirement Specifications at Mercedes-Benz: An Investigation Using a Combination of Legacy Data and Expert Opinion. In: Proceedings 20th IEEE International Requirements Engineering Conference (RE). Chicago, Illinois, USA, 24.09.2012-28.09.2012. Piscataway, NJ: IEEE Computer Society Press, S. 291–296.

Pahl, G.; **Beitz**, W.; **Feldhusen**, J.; **Grote**, K.-H. (**2007**): Konstruktionslehre - Grundlagen erfolgreicher Produktentwicklung ; Methoden und Anwendung. 7. Aufl. Berlin, Heidelberg: Springer.

Palomares, C.; **Franch**, X.; **Quer**, C. (**2014**): Requirements Reuse and Patterns: A Survey. In: Hutchison, D., Kanade, T. et al. (Hg.): Requirements Engineering: Foundation for Software Quality. 20th International Working Conference,

REFSQ 2014. Essen, Germany, 07.04.2014-10.04.2014. Cham: Springer (Lecture Notes in Computer Science, 8396), S. 301–308.

Paschek, D.; **Luminosu**, C. T.; **Draghici**, A. (**2017)**: Automated business process management – in times of digital transformation using machine learning or artificial intelligence. In: *MATEC Web Conf.* 121, S. 4007. DOI: 10.1051/matecconf/201712104007.

Patzak, G. (**1982)**: Systemtechnik - Planung komplexer innovativer Systeme - Grundlagen, Methoden, Techniken. Berlin: Springer.

Paulukuhn, L. (**2005)**: Typologisierung von Entwicklungsprojekten im Maschinenbau. Aachen: Shaker (Berichte aus der Produktionstechnik, Bd. 2005,1).

Pohl, K. (**2007)**: Requirements Engineering - Grundlagen, Prinzipien, Techniken. 1. Aufl. Heidelberg: Dpunkt-Verl.

Ponn, J.; **Lindemann**, U. (**2011)**: Konzeptentwicklung und Gestaltung technischer Produkte - Systematisch von Anforderungen zu Konzepten und Gestaltlösungen. 2. Aufl. Berlin Heidelberg: Springer (VDI-Buch).

Porter, M. F.; **Boulton**, R. (**2002)**: Stemming algorithms for various European languages. Online verfügbar unter https://snowballstem.org/algorithms/, zuletzt geprüft am 09.01.2022.

Refflinghaus, R. (**2000)**: Anbahnung von Forschungs- und Entwicklungskooperationen mit Methoden des Qualitätsmanagements. Dissertation. Universität Dortmund, Dortmund.

Regnell, B.; **Berntsson Svensson**, R.; **Wnuk**, K. (**2008)**: Can We Beat the Complexity of Very Large-Scale Requirements Engineering? In: Hutchison, D., Kanade, T. et al. (Hg.): Requirements Engineering: Foundation for Software Quality. 14th International Working Conference, REFSQ 2008. Montpellier, France, 16.06.2008-17.06.2008. Berlin, Heidelberg: Springer (Lecture Notes in Computer Science, 5025), S. 123–128.

Reindl, J. M. (**2013)**: Horizontale Einkaufs- und Entwicklungskooperationen in der deutschen Automobilindustrie: Universitätsverlag Chemnitz zugl. Dissertation Techn. Univ. Chemnitz 2013.

Reinsel, D.; **Gantz**, J.; **Rydning**, J. (**2018)**: The Digitization of the World - From Edge to Core. Hg. v. IDC - International Data Corporation.

Renault, S.; **Mendez-Bonilla**, O.; **Franch**, X.; **Quer**, C. (**2009)**: PABRE: Pattern-based Requirements Elicitation. In: Third International Conference on Research Challenges in Information Science (RCIS). Fez, Morocco, 22.04.2009 - 24.04.2009: IEEE Computer Society Press, S. 81–92.

Renner, I. (**2007)**: Methodische Unterstützung funktionsorientierter Baukastenentwicklung am Beispiel Automobil. München: Dr. Hut (Produktentwicklung) zugl. Dissertation Technische Universität München 2007.

Riechert, T.; **Lauenroth**, K.; **Lehmann**, J. (**2007)**: Semantisch unterstütztes Requirements Engineering. In: Auer, S., Bizer, C. et al. (Hg.): The social semantic web. Proceedings of the 1st Conference on Social Semantic Web (CSSW). Leipzig, 26.09.2007-28.09.2007. Bonn: Ges. für Informatik (GI) (GI-Edition : Proceedings, 113), S. 111–118.

Ristad, E. S. (**1993)**: The Anaphora Problem. In: *Information and Computation* 105 (1), S. 105–131. DOI: 10.1006/inco.1993.1042.

Ritter, L.; **Refflinghaus**, R. (**2020)**: Untersuchung variantenverursachender technischer Komponentenanforderungen und Bewertung der Folgen bei Entfall. In: Schmitt, R. H. (Hg.): Potenziale Künstlicher Intelligenz für die Qualitätswissenschaft. Berlin, Heidelberg: Springer, S. 109–126.

Robinson-Mallett, C.; **Köhnlein**, J.; **Wegener**, J.; **Grochtmann**, M. (**2009)**: Modellbasierte Anforderungsanalyse für die Entwicklungen variantenreicher Systeme - Erfahrungen mit modellbasierten Spezifikationen unter Variantenmanagement für Fahrzeugfunktionen. In: 14. Internationaler Kongress Elektronik im Kraftfahrzeug, Electronic Systems for Vehicles. Baden-Baden, 07.10.2009-08.10.2009. VDI. Düsseldorf: VDI-Verlag (VDI-Berichte, 2075), S. 277–290.

Rohleder, C. (**2008)**: Visualizing the Impact of 4on-Functional Requirements on Variants : A Case Study. In: 2008 Requirements Engineering Visualization. 2008 Requirements Engineering Visualization (REV). Barcelona, Spain, 08.09.2008 - 08.09.2008: IEEE Computer Society Press, S. 11–20.

Rösler, J.; **Harders**, H.; **Bäker**, M. (**2006)**: Mechanisches Verhalten der Werkstoffe. 2. Aufl. Wiesbaden: Teubner (Lehrbuch Maschinenbau).

Rothe, S. (**2016)**: Supervised and Unsupervised Methods for Learning Representations of Linguistic Units. Dissertation. Ludwig-Maximillians-Universität, München.

Ruge, J.; **Wohlfahrt**, H. (**2007**): Technologie der Werkstoffe - Herstellung, Verarbeitung, Einsatz. 8. Aufl. Wiesbaden: Vieweg (Vieweg Studium Technik).

Rupp, C. (**2013**): Systemanalyse kompakt. 3. Aufl. 2013. Berlin, Heidelberg: Springer (IT kompakt).

Rupp, C. (**2014**): Requirements-Engineering und -Management - Aus der Praxis von klassisch bis agil. 6., aktualisierte und erweiterte Aufl. München: Carl Hanser Verlag.

Rus, V.; **Lintean**, M.; **Banjade**, R.; **Niraula**, N.; **Stefanescu**, D. (**2013**): SEMILAR: The Semantic Similarity Toolkit. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. Sofia, Bulgaria, 04.08.2013-09.08.2013: Association for Computational Linguistics, S. 163–168.

Rzehorz, C. (**1998**): Wissensbasierte Anforderungsentwicklung auf der Basis eines integrierten Produktmodells. Karlsruhe: Shaker (Forschungsberichte aus dem Institut für Rechneranwendung in Planung und Konstruktion der Universität Karlsruhe, Bd. 98,3) zugl. Dissertation Universität Karlsruhe 1998.

Schernikau, J. (**2001**): Gestaltung von mechatronikgerechten Organisationen in der Produktentwicklung. Aachen: Shaker (Berichte aus der Produktionstechnik, 2001,8) zugl. Dissertation RWTH Aachen 2000.

Schienmann, B. (**2001**): Kontinuierliches Anforderungsmanagement: Prozesse - Techniken - Werkzeuge. München, Boston: Addison-Wesley.

Schlott, S. (**2005**): Wahnsinn mit Methode. In: *Automobil Produktion* (1), S. 38–42.

Schmelzer, H. J. (**1992**): Organisation und Controlling von Produktentwicklungen - Praxis des wettbewerbsorientierten Entwicklungsmanagements. Stuttgart: Schäffer-Poeschel (Management von Forschung, Entwicklung und Innovation, Bd. 11).

Schuh, G. (**1989**): Gestaltung und Bewertung von Produktvarianten : ein Beitrag zur systematischen Planung von Serienprodukten. Düsseldorf: VDI-Verlag (Fortschritt-Berichte VDI Reihe 2: Fertigungstechnik, 177).

Schuh, G. (**2005**): Produktkomplexität managen - Strategien - Methoden - Tools. 2. Aufl. München, Wien: Carl Hanser Verlag.

Schuh, G.; **Deger**, R.; **Jung**, M.; **Meier**, J.; **Lenders**, M. (**2008**): Managing Complexity in Automotive Engineering: Ergebnisse einer Studie. RWTH Aachen, Werkzeugmaschinenlabor WZL. Aachen.

Schuh, G.; **Kuntz**, J.; **Heeg**, K.; **Jussen**, P.; **Koch**, J.; **Breunig**, S. (**2016)**: Identification of variant-creating factors in product service systems. In: 2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM). Bali, Indonesia, 04.12.2016-07.12.2016: IEEE Computer Society Press, S. 582–586.

Schuh, G.; **Riesener**, M.; **Rudolf**, S. (**2014)**: Identifying Preferable Product Variants Using Similarity Analysis. In: *Procedia CIRP* 20, S. 38–43. DOI: 10.1016/j.procir.2014.05.029.

Siddique, Z.; **Rosen**, D. W.; **Wang**, N. (**1998)**: On the Applicability of Product Variety Design Concepts to Automotive Platform Commonality. In: Volume 3: 10th International Conference on Design Theory and Methodology. ASME 1998 Design Engineering Technical Conferences. Atlanta, Georgia, USA, 13.09.1998 - 16.09.1998: American Society of Mechanical Engineers.

Sidner, C. L. (**1983)**: Focusing in the comprehension of definite anaphora. In: Brady, M. und Berwick, R. C. (Hg.): Computational models of discourse. Cambridge, Massachusetts: MIT Press, S. 267–330.

Siegel, M.; **Bond**, F. (**2021)**: OdeNet: Compiling a GermanWordnet from other Resources. In: Proceedings of the 11th Global Wordnet Conference (GWC'21). virtuell, 18.01.2021-21.01.2021: Global Wordnet Association.

Sitte, J.; **Winzer**, P. (**2005)**: Demand compliant design of robotic systems. In: IEEE International Conference Mechatronics and Automation, 2005. Niagara Falls, Ont., Canada, 29.07.2005-01.08.2005: IEEE Computer Society Press, S. 1953–1958.

Sitte, J.; **Winzer**, P. (**2011)**: Demand-Compliant Design. In: *IEEE Trans. Syst., Man, Cybern. A* 41 (3), S. 434–448. DOI: 10.1109/TSMCA.2010.2080672.

Sommerville, I. (**2012)**: Software engineering. 9., aktualisierte Auflage. München: Pearson (Always learning).

Sommerville, I.; **Sawyer**, P. (**2006)**: Requirements engineering - A good practice guide. reprinted. Chichester: Wiley.

Stechert, C. (**2010)**: Modellierung komplexer Anforderungen. Dissertation. Technische Universität Carolo-Wilhelmina zu Braunschweig, Braunschweig.

Steinhilper, W.; **Sauer**, B. (**2008**): Konstruktionselemente des Maschinenbaus 1 - Grundlagen der Berechnung und Gestaltung von Maschinenelementen. Berlin, Heidelberg: Springer.

Steinhilper, W.; **Sauer**, B. (**2008**): Konstruktionselemente des Maschinenbaus 2 - Grundlagen von Maschinenelementen für Antriebsaufgaben. Berlin, Heidelberg: Springer.

Stoiber, R.; **Glinz**, M. (**2010**): Feature unweaving: Efficient variability extraction and specification for emerging software product lines. In: 4th International Workshop on Software Product Management (IWSPM). Sydney, Australia, 27.09.2010, S. 53–62.

Strube, M.; **Rapp**, S.; **Müller**, C. (**2002**): The influence of minimum edit distance on reference resolution. In: EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10. Philadelphia, PA, USA, 06.07.2002-07.07.2002. Morristown, NJ, USA: Association for Computational Linguistics, S. 312–319.

Tavakoli Kolagari, R.; **Reiser**, M.-O. (**2007**): Reusing Requirements: The Need for Extended Variability Models. In: Arbab, F. und Sirjani, M. (Hg.): Proceedings of the 2007 international conference on Fundamentals of software engineering. Teheran, 17.04.2007-19.04.2007. Berlin, Heidelberg: Springer (ACM Digital Library), S. 129–143.

Telljohann, H.; **Hinrichs**, E.; **Kübler**, S. (**2004**): The Tüba-D/Z Treebank: Annotating German with a Context-Free Backbone. In: Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04). Lisbon, Portugal, 26.05.2004-28.05.2004: European Language Resources Association (ELRA), S. 2229–2235.

Tesch, F. L. (**2010**): Bewertung der Strukturvariabilität von Pkw-Karosseriederivaten. Dissertation. Technische Universität München, München.

Theumert, H.; **Fleischer**, B. (**2007**): Entwickeln - Konstruieren - Berechnen - Komplexe praxisnahe Beispiele mit Lösungsvarianten. 1. Aufl. Wiesbaden: Vieweg (Viewegs Fachbücher der Technik).

Tseng, M. M.; **Jiao**, J. (**1997**): A variant approach to product definition by recognizing functional requirement patterns. In: *Computers & Industrial Engineering* 33 (3-4), S. 629–633. DOI: 10.1016/S0360-8352(97)00209-X.

Turney, P. D. (**2001**): Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In: Goos, G., Hartmanis, J. et al. (Hg.): Machine Learning: ECML 2001. 12th European Conference on Machine Learning Proceedings, Bd. 2167. Freiburg, Germany, 05.09.2001-07.09.2001. Berlin, Heidelberg: Springer (Lecture Notes in Computer Science), S. 491–502.

Ulrich, H.; **Probst**, G. J.B. (**1991**): Anleitung zum ganzheitlichen Denken und Handeln - Ein Brevier für Führungskräfte. 3., erw. Aufl. Bern, Stuttgart: Haupt.

Ulrich, K. T.; **Eppinger**, S. D.; **Yang**, M. C. (**2020**): Product design and development. Seventh edition, International student edition. New York, NY: McGraw-Hill.

VDA (**2007**): Automotive VDA-Standardstruktur Komponentenlastenheft. 1. Aufl. Oberursel: VDA.

VDA (**2010**): Technische Sauberkeit in der Montage - Umgebung, Logistik. Personal und Montageeinrichtungen. 1. Aufl. (Qualitätsmanagement in der Automobilindustrie, 19) (Teil 2).

VDA (**2015**): Jahresbericht 2015. Berlin.

VDA (**2015**): Prüfung der Technischen Sauberkeit. 2. Aufl. (Qualitätsmanagement in der Automobilindustrie, 19) (Teil 1).

Vendég, T. (**2020**): Vier Schritte zur wirkungsvollen Zusammenarbeit von Produktentwicklung und Produktion. In: *Produktion* (13), S. 31.

Versley, Y. (**2007**): Antecedent Selection Techniques for High-Recall Coreference Resolution. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). Prag, Czech Republic, 28.06.2007-30.06.2007: Association for Computational Linguistics, S. 496–505.

Versteegen, G.; **Heßeler**, A.; **Hood**, C. **et al.** (**2004**): Anforderungsmanagement - Formale Prozesse, Praxiserfahrungen, Einführungsstrategien und Toolauswahl. Berlin u.a.: Springer (Xpert.press).

Vogelsang, A.; **Femmer**, H.; **Winkler**, C. (**2016**): Take Care of Your Modes! An Investigation of Defects in Automotive Requirements. In: Daneva, M. und Pastor, O. (Hg.): Requirements Engineering: Foundation for Software Quality. 22nd International Working Conference, REFSQ 2016. Gothenburg, Sweden,

14.03.2016–17.03.2016. Cham: Springer (Lecture Notes in Computer Science, 9619), S. 161–167.

Walker, M. A.; **Joshi**, A. K.; **Prince**, E. F. (Hg.) (**1998)**: Centering theory in discourse. Oxford: Oxford Univ. Press.

Waltl, B. E. (**2018)**: Semantic Analysis and Computational Modeling of Legal Documents. Dissertation. Technische Universität München, München.

Wang, Y.; **Tseng**, M. M. (**2015)**: A Naïve Bayes approach to map customer requirements to product variants. In: *J Intell Manuf* 26 (3), S. 501–509. DOI: 10.1007/s10845-013-0806-2.

Wang, Y.; **Zhang**, J. (**2016)**: Experiment on automatic functional requirements analysis with the EFRF's semantic cases. In: 2016 International Conference on Progress in Informatics and Computing (PIC). Shanghai, China, 23.12.2016 - 25.12.2016: IEEE Computer Society Press, S. 636–642.

Webber, F. D. S. (**2016)**: Semantic Folding Theory And its Application in Semantic Fingerprinting. Wien. Online verfügbar unter https://arxiv.org/ftp/arxiv/papers/1511/1511.08855.pdf, zuletzt geprüft am 15.03.2018.

Wildemann, H. (**1995)**: Komplexitätsmanagement in der Fabrikorganisation. In: *ZWF - Zeitschrift für wirtschaftlichen Fabrikbetrieb* 90 (1-2), S. 21–26.

Wildemann, H. (**2012)**: Variantenmanagement - Leitfaden zur Komplexitätsreduzierung, -beherrschung und -vermeidung in Produkt und Prozess. 20. Aufl. München, München: TCW-Verlag (Leitfaden / TCW Transfer-Centrum für Produktionslogistik und Technologie-Management, 5).

Wolters, H. (**1995)**: Modul- und Systembeschaffung in der Automobilindustrie. Wiesbaden: Deutscher Universitätsverlag; Gabler (Gabler Edition Wissenschaft) zugl. Dissertation Freie Univ. Berlin 1995.

Zimmermann, G. (**1988)**: Produktionsplanung variantenreicher Erzeugnisse mit EDV. Berlin, Heidelberg, New York, London, Paris, Tokyo: Springer (Betriebs- und Wirtschaftsinformatik, 30).

Normen und Standards

ISO. **16355-1**:2021-05: Anwendung von statistischen und verwandten Methoden für neue Technologie und für den Produktentwicklungsprozess - Teil 1: Allgemeine Grundsätze und Perspektive der QFD-Methode.

DIN 199-2:1977-12: Begriffe im Zeichnungs- und Stücklistenwesen; Stücklisten.

DIN EN ISO. **20567-1**:2017-07: Beschichtungsstoffe – Prüfung der Steinschlagfestigkeit von Beschichtungen – Teil 1: Multischlagprüfung.

IEEE Std. **610.12**:1990: IEEE Standard Glossary of Software Engineering Terminology.

DIN IEC. **60050-351**:2014-09: Internationales Elektrotechnisches Wörterbuch.

DIN EN ISO. **8044**:2015-12: Korrosion von Metallen und Legierungen – Grundbegriffe.

DIN EN ISO. **9227**:2017-07: Korrosionsprüfungen in künstlichen Atmosphären – Salzsprühnebelprüfungen.

VDI/VDE. **3694**:2014-04: Lastenheft/Pflichtenheft für den Einsatz von Automatisierungssystemen.

VDI. **2221**:1993-05: Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte.

DIN. **69901-5**:2009-01: Projektmanagement – Projektmanagementsysteme – Teil 5: Begriffe.

DIN EN ISO . **9000**:2015-11: Qualitätsmanagementsysteme – Grundlagen und Begriffe.

IEEE Std. **830**:1998: Recommended Practice for Software Requirements Specifications.

DIN ISO. **8855**:2013-11: Straßenfahrzeuge – Fahrzeugdynamik und Fahrverhalten – Begriffe.

DIN. **70010**:2001-04: Systematik der Straßenfahrzeuge.

VDI. **2230 Blatt 2**:2014: Systematische Berechnung hochbeanspruchter Schraubenverbindungen Mehrschraubenverbindungen.

VDI. **2230 Blatt 1**:2015: Systematische Berechnung hochbeanspruchter Schraubenverbindungen Zylindrische Einschraubenverbindungen.

Abkürzungsverzeichnis

2WD	two wheel drive = Front- oder Heckantrieb
4WD	four wheel drive = Allradantrieb
AI	Artificial Intelligence
AM	Anforderungsmanagement
AWD	all wheel drive = Allradantrieb
BEV	Battery Electric Vehicle = Elektrofahrzeug
BoW	Bag of Words
BPMN	Business Process Model & Notation
CAD	Computer Aided Design
CNN	convolutional neural networks
COP	Carry Over Part, Gleichteil
CVA	Commonality and Variability Analysis
DeCoDe	Demand Compliant Design (Sitte et al. 2005; Sitte et al. 2011)
DISCO	Extracting DIStribu-tionally similar words using Cooccurrences
EAC	European Accreditation and Certification
ECU	Electronic Control Unit, Steuergerät
EDL	Entwicklungsdienstleister
EPK	ereignisgesteuerte Prozessketten
ESA	Explicit Semantic Analysis
et al.	et alia = und andere
EU	Europäische Union
f.	folgende
ff.	fortfolgende
FN	false negative
FP	false positive
FuE	Forschung und Entwicklung
FWD	front wheel drive = Frontantrieb
HAL	Hyperspace Analogues to Language
HoQ	House of Quality
HTM	Hierarchical Temporal Memory

ICE	Internal Combustion Engine = Verbrennungsmotor
ID	Identifikationsnummer
KDVM	Konfigurationsgekoppelte dezentr. Variabilitätsmodellierung (Boutkova 2014)
KI	künstliche Intelligenz
KNN	künstliches neuronales Netz
KVP	kontinuierlicher Verbesserungsprozess
LH	Lastenheft
LSA	Latent Semantic Analysis
LSTM	long short-term memory
LSV	Leistungsvereinbarung
Mgmt.	Management
MIA	Merkmalsidentifizierungsansatz (Boutkova 2014)
MIA Add-in	Military Transport Aircraft Division Inheritance Add-in (Monzon 2008)
ML	maschinelles Lernen, engl. machine learning
MnP	Monate nach Produktion
MRAM	Method for Requirements Authoring Management (Mannion et al. 2008)
NGD	Normalized Google Distance
NLP	Natural Language Processing
NP	Nominalphrase
OEM	Original Equipment Manufacturer
oiU	ohne inhaltlichen Unterschied
OLE	object linking and embedding
PDM	Produkt Daten Management
PHEV	Plugin Hybrid Electric Vehicle= Hybridantrieb
PMI-IR	Pointwise Mutual Information - Information Retrieval
POS-tagging	Wortartannotierung, engl. part of speech tagging
QFD	Quality Function Deployment
QM	Qualitätsmanagement
R2F	Requirements to Feature Ansatz (Boutkova 2014)

RE	Requirements Engineering
RECAA	Requirements Engineering Complete Automation Approach (Körner 2014)
REFS	Requirements Engineering Feedback System
ReMaS	Requirements Management System (Jörg 2005)
RESI	Requirements Engineering Specification Improver (Körner 2014)
RM	Requirements Management
RM&E	Requirements Management and Engineering
RNN	recurrent neural network
RWD	rear wheel drive = Heckantrieb
SA	Sonderausstattung
SDR	Sparse Distributed Representation
SFT	Semantic Folding Theory
SOP	Start of Production
STTS	Stuttgart-Tübingen-Tagset
SUV	Sports Utility Vehicle
TN	true negative
TP	true positive
UML	Unified Modeling Language
vgl.	vergleiche
VMEA	Variant Modes and Effects Analysis
VP	Verbalphrase
Z	Zerlegungsfehler

Eine kundenorientierte Produktentwicklung ist wichtig, um mit einem Produkt am Markt erfolgreich zu sein. Gleichzeitig werden Anforderungen, die an ein Produkt gestellt werden, vielfältiger. Ein verbreitetes Vorgehen zum Umgang mit der großen Anforderungsmenge ist die unstrukturierte Wiederverwendung durch das Kopieren einzelner Anforderungen bis hin ganzer Anforderungsdokumente.
Um die Wiederverwendung von Anforderungen aus Komponentenlastenheften zu unterstützen, wurde daher eine automatisierte Methode zur Strukturierung der Anforderungen für den Aufbau einer Anforderungsbibliothek entwickelt, die gleichzeitig die Unterschiede in den Anforderungen transparent macht.
Die entwickelte Methode verwendet den Ansatz des Semantic Folding aus dem Bereich des Natural Language Processing (NLP), um Anforderungen inhaltlich automatisiert miteinander zu vergleichen. Am Beispiel von Komponentenlastenheften aus der Fahrwerksentwicklung wird gezeigt, dass 92,3 % der Anforderungen korrekt durch die automatisierte Methode als inhaltlich ähnlich oder verschieden erkannt werden. Dadurch können gezielt Unterschiede in den Anforderungen von den Entwicklern hinterfragt und wenn möglich vereinheitlicht werden.

ISBN 978-3-7376-1140-4