Product Safety and Quality Engineering
Manuel Löwer und Nadine Schlüter (Hrsg.)

4

Entwicklung eines Vorgehenskonzepts zur modellbasierten System-konkretisierung und -analyse in den frühen Phasen der Produkt-entwicklung

Florian Riekhof

PRODUKT SICHERHEIT QUALITÄT
BERGISCHE UNIVERSITÄT WUPPERTAL

Product Safety and Quality Engineering – Band 4
Florian Riekhof
Entwicklung eines Vorgehenskonzepts zur modellbasierten System-konkretisierung und -analyse in den frühen Phasen der Produktentwicklung
Herausgeber: Manuel Löwer und Nadine Schlüter,
Bergische Universität Wuppertal,
Fachgebiet Produktsicherheit und Qualität,
Gaußstraße 20, 42119 Wuppertal
Umschlaggestaltung: Franz Wieck

Zugl.: Wuppertal, Univ., Diss. 2024
DOI: https://doi.org/10.25926/BUW/0-184
URN: https://nbn-resolving.org/urn:nbn:de:hbz:468-2-1867

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

Dieses Werk steht, soweit nicht anders angegeben, unter der Creative Commons-Lizenz CC BY-NC 4.0
https://creativecommons.org/licenses/by-nc/4.0/deed.de

BERGISCHE
UNIVERSITÄT
WUPPERTAL

Entwicklung eines Vorgehenskonzepts zur modellbasierten Systemkonkretisierung und -analyse in den frühen Phasen der Produktentwicklung

Dissertation
zur Erlangung eines Doktorgrades

in der
Fakultät für Maschinenbau und Sicherheitstechnik

der
Bergischen Universität Wuppertal

vorgelegt von

Florian Riekhof

Wuppertal 2022

Zusammenfassung

Die Entwicklung technischer Systeme stellt besondere Anforderungen an den Entwicklungsprozess, wobei insbesondere die frühen Phasen der Produktentwicklung entscheidend für die spätere Anforderungserfüllung sind. Aufgrund geringer Informationsverfügbarkeit in diesen Phasen, schnell steigender Komplexität und hoher Erwartungen an die Zuverlässigkeit müssen geeignete Methoden implementiert werden. Um diesen Herausforderungen zu begegnen, bieten Fähigkeits- und Reifegradmodelle einen Ansatz mit generischem Prozessbezug. Das etablierte *Capability Maturity Model Integration for Development* (CMMI-DEV) stellt spezifische Ziele und Praktiken bereit, deren Implementierung die Umsetzung fähiger Prozesse sicherstellt. Ansätze mit konkretem Produktbezug, wie das Reifegradabsicherungs-Modell (RGA-Modell), geben produktbezogene Ergebnisse vor, welche im Rahmen der Produktentwicklung zu erreichen und an festgelegten Meilensteinen zur Erfassung des Produktreifegrads zu bewerten sind. Die Ansätze liefern somit Anforderungen an den Prozess und das Produkt, sie bieten jedoch keine Unterstützung bei der Erreichung der Ergebnisse.

Zur Schließung dieser methodischen Lücke und unter Berücksichtigung der Herausforderungen bei der Entwicklung technischer Systeme (Informationsverfügbarkeit, Komplexität, Zuverlässigkeit) wird ein Vorgehenskonzept entwickelt, welches Anforderungen umsetzt, die aus dem RGA-Modell und dem CMMI-DEV abgeleitet wurden. Für frühe Entwicklungsphasen wird so basierend auf den drei verknüpften Bausteinen Systemkonkretisierung und -analyse (i), Systemmodellierung (ii) und Produktreifegradbewertung (iii) ein Vorgehen definiert, welches eine methodische Unterstützung der Produktentwicklungsprozesse von der Systementwurfsphase bis zur Übergabe in die Phase des domänenspezifischen Entwurfs ermöglicht. Hierbei wird die sukzessive Systemkonkretisierung über Lösungsebenen sowie die Erfassung und Analyse von Systemstrukturen über die Schrittfolge zur Konkretisierung des Lösungsraums und die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifikation und -analyse (i) untersetzt und im Systemmodell (ii) abgebildet. An definierten Meilensteinen, d.h. nach jedem Durchlauf der Schrittfolgen, wird für die aufgespannten Lösungsebenen anhand zuvor definierter Kriterien der Produktreifegrad (iii) ermittelt. Dabei integriert der Ansatz Erkenntnisse aus dem Bereich des Komplexitätsmanagements und der Graphentheorie für die Systemmodellierung. Gleichzeitig wird der Erkenntnis Rechnung getragen, dass Systemstrukturen über strukturelle Relationen und technisch-physikalische Schnittstellen zwischen Systemelementen beschrieben werden können. Die Identifizierung und entwicklungsbegleitende Modellierung dieser Systemstrukturen ermöglicht die Beherrschung von Komplexität und erhöht die Informationsverfügbarkeit. Die Analyse der technisch-physikalischen Schnittstellen ermöglicht die Identifizierung von Störgrößen und somit die zielgerichtete Einbindung von Analysen und Simulationen sowie die Definition geeigneter Optimierungsmaßnahmen mit dem Ziel, frühzeitig störende Einflüsse zu eliminieren und die Zuverlässigkeit des Systems zu erhöhen. Durch das Vorgehenskonzept wird auf diese Weise einerseits die Beherrschung der beschriebenen Herausforderungen (Informationsverfügbarkeit, Komplexität, Zuverlässigkeit) sichergestellt, andererseits wird durch die Umsetzung der abgeleiteten Anforderungen aus dem CMMI-DEV und dem RGA-Modell die methodische Lücke zwischen Ansätzen mit generischem Prozess- und konkretem Produktbezug geschlossen.

Abstract

The development of technical systems places special demands on the development process, whereby the early phases of product development in particular are crucial for the subsequent fulfillment of requirements. Due to low information availability in these phases, rapidly increasing complexity and high reliability expectations, appropriate methods have to be implemented. To meet these challenges, capability and maturity models offer an approach with generic process reference. The established *Capability Maturity Model Integration for Development* (CMMI-DEV) provides specific goals and practices whose implementation ensures the realization of capable processes. Approaches with a concrete product reference, such as the maturity assurance model (RGA model), specify product-related results which are to be achieved within the framework of product development and evaluated at defined milestones for recording the product maturity level. The approaches thus provide requirements for the process and the product, but they do not provide support for achieving these outcomes.

To close this methodological gap and taking into account the challenges in the development of technical systems (information availability, complexity, reliability), a procedural concept is developed which implements requirements derived from the RGA model and the CMMI-DEV. For early development phases, a procedure is defined based on the three linked components of system concretization and analysis (i), system modeling (ii) and product maturity assessment (iii), which enables methodical support of the product development processes from the system design phase to the transfer to the domain-specific design phase. Here, the successive system concretization via solution levels as well as the identification and analysis of system structures via the step sequence for the concretization of the solution space and the step sequence for relation determination and interface identification and analysis (i) is supported and represented in the system model (ii). At defined milestones, i.e. after each run through of the step sequences, the product maturity level (iii) is determined for the spanned solution levels on the basis of previously defined criteria. In doing so, the approach integrates insights from the field of complexity management and graph theory for system modeling. At the same time, it considers the fact that system structures can be described via structural relations and technical-physical interfaces between system elements. The identification and development-accompanying modeling of these system structures allows the control of complexity and increases the availability of information. The analysis of the technical-physical interfaces enables the identification of disturbance variables and thus the targeted integration of analyses and simulations as well as the definition of suitable optimization measures with the aim of eliminating disturbing influences at an early stage and increasing the reliability of the system. On the one hand, the procedural concept ensures that the challenges described (information availability, complexity, reliability) are mastered; on the other hand, the implementation of derived requirements from the CMMI-DEV and the RGA model closes the methodological gap between approaches with a generic process reference and a concrete product reference.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Anhangsverzeichnis

Abkürzungsverzeichnis

CMF	CMMI Modell Foundation
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
CMMI-ACQ	Capability Maturity Model Integration for Acquisition
CMMI-DEV	Capability Maturity Model Integration for Development
CMMI-SVC	Capability Maturity Model Integration for Services
DeCoDe	Demand Compliant Design
DIN	Deutsches Institut für Normung e.V.
DMM	Domain Mapping Matrix
DSM	Design Structure Matrix
DV	Design Verification
ECM	Engineering Change Management
FBA	Fehlzustandsbaumanalyse, auch → *FTA*
FIT	Failure in Time; Anzahl Fehler pro 10^9 Bauelementestunden
FMEA	Fehlzustandsart- und -auswirkungsanalyse (auch: Fehlermöglichkeits- und -einflussanalyse), engl. Failure Mode and Effects Analysis
FTA	Fault Tree Analysis, dt. Fehlzustandsbaumanalyse (FBA)
HoQ	House of Quality
LASM	Linearasynchronmaschine
MDM	Multiple-domain Matrix
MTBF	Mean Time Between Failure, dt. Mittelwert der ausfallfreien Funktionszeit nicht-reparierbarer Systeme
OEM	Original Equipment Manufacturer, dt. Erstausrüster
PEP	Produktentwicklungsprozess
PLC	Product Life Cycle, dt. Produktlebenszyklus
PPF-Verfahren	Produktionsprozess- und Produktfreigabe-Verfahren
PV	Product Validation
Q-ELF	DFG-Projekt Q-ELF (Akronym), **Q**ualitätsorientierter Methodenworkflow für die Produkt**e**ntwicklung eines **L**inearbetriebes in der **F**ördertechnik

QFD	Quality Function Deployment
QMMG	Quality Management Maturity Grid
RG	Reifegrad
RGA-Modell	Reifegradabsicherungs-Modell
RPZ	Risikoprioritätszahl
SEI	Software Engineering Institute
SG	Specific Goal, dt. spezifisches Ziel
SOP	Start of Production, dt. Produktionsanlauf
SP	Specific Practice, dt. spezifische Praktik
SPICe	Software Process Improvement and Capability Determination
SysML	Systems Modeling Language; auf → *UML* basierende, objektorientierte Modellierungssoftware zur Abbildung von Systemen
UML	Unified Modeling Language; objektorientierte Modellierungssoftware zur Abbildung von Systemen
VDA	Verband der Automobilindustrie e.V.
VDI	Verein Deutscher Ingenieure e.V.
ZVEI	Zentralverband Elektrotechnik- und Elektroindustrie e.V.

1 Einleitung

1.1 Motivation

Die Entwicklung anforderungsgerechter technischer Systeme ist eine erklärte Zielstellung der an der Produktentwicklung beteiligten Domänen. Dieses Ziel beinhaltet unterschiedliche Dimensionen und wirkt sich über den gesamten Produktlebenszyklus (Product Life Cycle, PLC) auf alle unternehmerischen Bereiche aus. Untersuchungen und Studien belegen jedoch, dass insbesondere an Schnittstellen zwischen Domänen Probleme auftreten [Schlund et al. 2009], die zunehmend in frühzeitigen Ausfällen und somit unzuverlässigen Systemen oder einer als mangelhaft empfundenen bzw. tatsächlich mangelhaften Anforderungserfüllung resultieren.

Diese Entwicklung wurde zum Anlass genommen, um unter der Überschrift „Überholte Q-Methoden?“ die Eignung bisheriger Konzepte zu hinterfragen: „Braucht das Qualitätsmanagement neue oder zumindest erneuerte Methoden?“ [Taucher 2014]. Hinter dieser Frage steckt die Unsicherheit, ob Methoden aus dem Bereich der Qualitäts- und Zuverlässigkeitstechnik, welche z.T. vor mehreren Jahrzehnten entwickelt wurden, geeignet sind, den Herausforderungen bei der Entwicklung moderner, funktional integrierter und komplexer mechatronischer Systeme gerecht zu werden. Als indirekte Antwort auf die eingangs formulierte Frage fordert WINZER einen systemischen Ansatz, der sich modular bestehender Konzepte bedient und evolutionär zu modifizieren ist [Winzer 2014, S. 16f]. ARTISCHEWSKI und SOMMERHOFF fordern in diesem Kontext die *Qualitätssicherung 4.0*, d.h. die Vernetzung von Qualitätswerkzeugen als Reaktion auf die Industrie 4.0 [Artischewski/Sommerhoff 2014, S. 63].

Unabhängig von der Benennung und der Vision derzeitiger Entwicklungstrends verdeutlichen die Diskussion sowie die zunehmenden technischen Probleme die Notwendigkeit einer Neukonzeptionierung. Hierbei gilt es nicht, einzelne Methoden durch eine weitere Granularisierung auf spezifische Problemfälle und somit ausschließlich bedarfsbezogen anzupassen. Vielmehr muss das Ziel in der Schaffung eines Vorgehenskonzeptes bestehen, welches den aktuellen Herausforderungen der Produktentwicklung gerecht wird und im Sinne der geforderten Modularität die Integration von Methoden unterschiedlicher Domänen ermöglicht und diese zusammenführt. Zur Erreichung dieses Ziels ist ein systemischer Ansatz erforderlich, der etablierte Methoden synergetisch vernetzt.

Dem nachfolgend präsentierten Ansatz liegt daher das Bestreben zugrunde, die Entwicklung anforderungsgerechter technischer Systeme durch eine methodische Unterstützung der Abläufe zu optimieren. Dieses Bestreben ist Bestandteil der vielschichtigen Ansätze zur Optimierung der Produktentwicklung, um ganzheitliche Konzepte zu entwickeln, die Anforderungen bestmöglich erfüllen.

1.2 Problemstellung

Bei der Entwicklung technischer Systeme werden steigende Anforderungen hinsichtlich Leistungsumfang und Zuverlässigkeit oftmals durch eine räumliche und funktionale Integration technischer Systeme umgesetzt [VDI 2206]. So gilt die Mechatronik beispielsweise in der Automobiltechnik als treibender Innovationsfaktor [Diehl 2009, S. 55], wobei der Bedarf und

Wertanteil mechatronischer Systeme kontinuierlich ansteigen [Isermann 2008, S. 15]. In der Mechatronik wirken klassisch autarke Domänen zusammen, bei denen Maschinenbau, Elektrotechnik und Informationstechnik prägende Elemente sind, wie Abb. 1 aufzeigt.

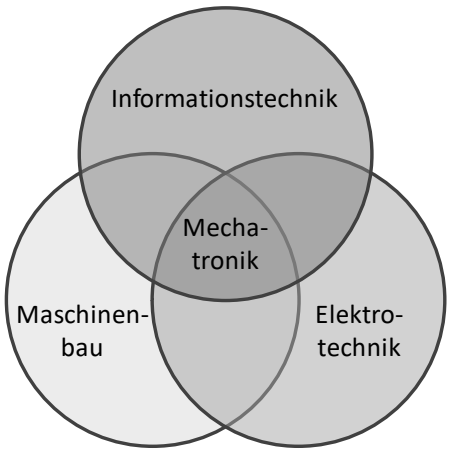


Abb. 1: Domänen der Mechatronik [VDI 2206]

Durch die Systemintegration steigt jedoch auch die Komplexität technischer Systeme [Diehl 2009, S. 1], die Herstellern zunehmend Probleme bereitet und sich unter anderem in steigenden Ausfällen und Rückrufen äußert [Schlund et al. 2009]. So konnte belegt werden, dass Probleme beispielsweise in der Unkenntnis über die Auswirkung von Systemänderungen beruhen [Riekhof et al. 2013a]. Dies steht konträr zu der insbesondere bei Herstellern komplexer Systeme dominierenden Forderung nach hoher Zuverlässigkeit, die selbst bei Investitionsgütern höher wiegt als monetäre Aspekte [Schlund et al. 2009]. Einen praxisbezogenen Beleg dieser Beobachtung liefern Kriterien für den Neuwagenkauf, bei denen die Zuverlässigkeit deutlich vor dem Aussehen des Fahrzeugs und dem Anschaffungspreis seit Jahren dominiert [DAT 2000 – DAT 2021].

Es lässt sich folgern, dass die Entwicklung zuverlässiger und komplexer technischer Systeme eine zentrale Herausforderung der Produktentwicklung ist. Dabei besteht eine Korrelation zwischen Komplexität und Zuverlässigkeit: Die geforderte räumliche und funktionale Integration von domänenspezifischen Lösungen in ein Gesamtsystem verursacht eine Vielzahl neuer Beziehungen zwischen Systemelementen und erhöht die Komplexität. Wird Komplexität nicht beherrscht, können Probleme resultieren, die sich insbesondere aus unbekannten oder nicht berücksichtigten Relationen ergeben. So konnten insbesondere Schnittstellen zwischen den Domänen als Quelle von Zuverlässigkeitsproblemen identifiziert werden [Schlund et al. 2009, Dorociak 2012].

Bei einer Einordnung der beschriebenen Herausforderungen in den zeitlichen Kontext der Produktentwicklungsprozesse (PEP) zeigt sich, dass insbesondere die frühen Phasen der Produktentwicklung von maßgeblicher Bedeutung sind. So lässt sich feststellen, dass die Fehlerverursachung größtenteils in den frühen Phasen der Produktentwicklung erfolgt, die Fehlerentdeckung jedoch oftmals erst in der Fertigung bzw. dem Feld möglich ist. Die Fehlerbeseitigungskosten steigen dabei über die Phasen des Produktlebenszyklus (Product Life Cycle, PLC) stark an [Bertsche et al. 2009, S. 200, Pfeifer 2001, S. 188f, von Regius 2006, S. 9]. Die sogenannte „Rule of Ten“ skizziert diesen Zusammenhang in einer idealisierten Kurve der Fehlerkosten über den PLC, bei der die Fehlerkosten je Lebenszyklusphase (Entwicklung, Produktion, Feld)

um eine Zehnerpotenz steigen, vgl. Abb. 2. Dem Reaktionszwang nach einer Fehlerentdeckung im Feld steht als Alternative die Fehlerverhütung gegenüber, welche ihr größtes Potential in den frühen Entwicklungsphasen entfalten kann.

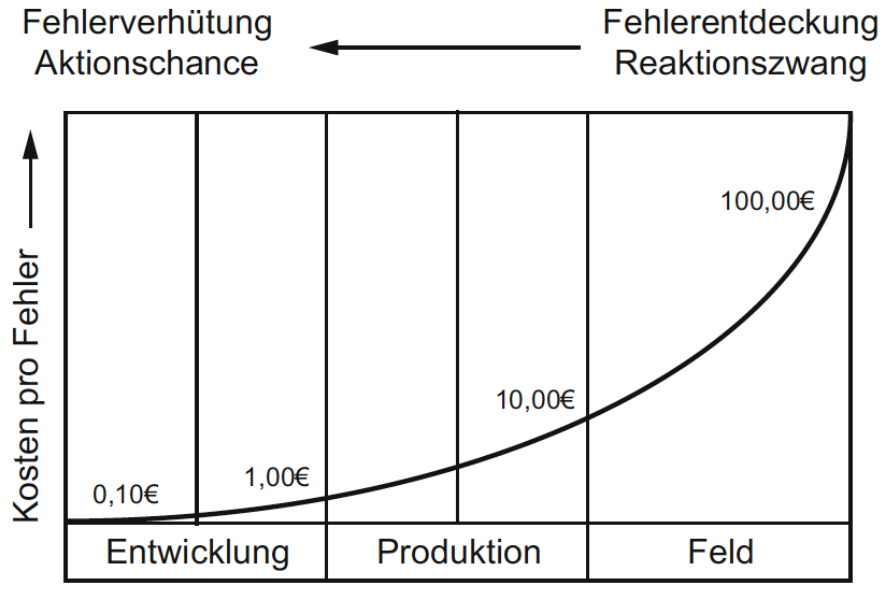


Abb. 2: Fehlerkosten über den PLC (Rule of Ten [Bertsche et al. 2009, S. 200])

Das exponentielle Wachstum der Fehlerkosten resultiert aus dem Delta zwischen dem Zeitpunkt der Fehlerentstehung und der Fehlerbehebung. Oftmals werden Fehler[1] erst in nachgelagerten Phasen entdeckt, was gemäß der „Rule of Ten“ enorme Kostenzuwächse für deren Behebung verursacht [Pfeifer 2001, S. XXVII, von Regius 2006, S. 10, Westkämper 2006, S. 131]. Desweiteren zeigt sich, dass Fehler vorrangig in frühen Produktentwicklungsphasen entstehen. WESTKÄMPER geht davon aus, dass 80 % aller Fehler erst nach der Produktfertigung behoben werden, wobei 75 % der Fehler in der Produktentwicklung verursacht wurden [Westkämper 2006, S. 131]. Abb. 3 stellt diesen Zusammenhang mittels zweier Verteilungskurven für die Fehlerentstehung und die Fehlerbehebung über Produktlebenszyklusphasen dar.

[1] Fehler stellen im Sinne der DIN EN ISO 9000:2015 eine Nonkonformität dar, d.h. die „Nichterfüllung einer Anforderung“ [DIN EN ISO 9000:2015, S. 40]

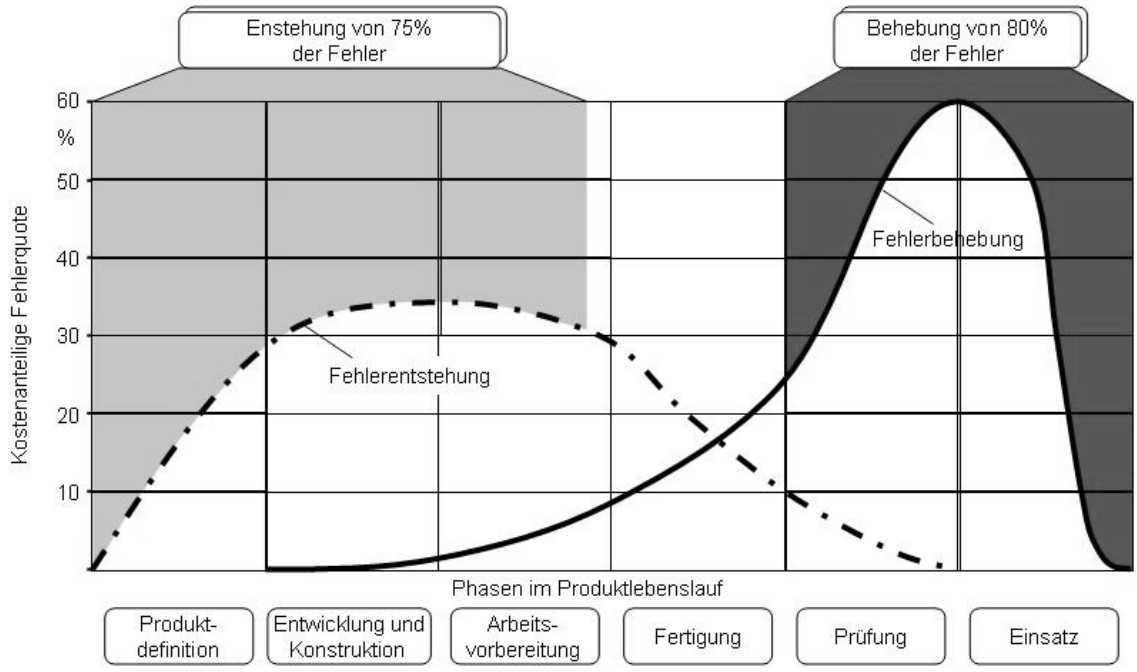


Abb. 3: Fehlerentstehung und Fehlerbehebung [Westkämper 2006, S. 131]

Bezugnehmend auf die im Jahr 2012 zurückgezogene VDI 2247 „Qualitätsmanagement in der Produktentwicklung“ [VDI 2247] beschreibt SCHLUND, dass die Fehlerverursachung zu 75 % in der Produktentwicklung begründet ist und sich 40 % der innerhalb der Garantiezeit auftretenden Ausfälle auf diese Produktfehler zurückführen lassen [Schlund 2011, S. 30f]. Dieser Zusammenhang ist in der Kostenfestlegung und -verursachung begründet, da in den Phasen der Produktentwicklung etwa 75 % der Produktgesamtkosten festgelegt werden [Müller 2007, S. 2, Vajna et al. 1994, S. 8]. Unabhängig von den angegebenen, leicht divergierenden Prozentsätzen der Fehlerverursachung bzw. des Fehlerauftretens, deren angegebene Werte vermutlich eher auf einer Näherung als empirischen Nachweisen beruhen, ist die Kernaussage unstrittig: Fehler werden primär in der Produktentwicklung verursacht und treten in Phasen auf, welcher der Fertigung nachgelagert sind, d.h. der finalen Qualitätsprüfung oder im Feld.

Aus der beschriebenen Problematik leitet sich der Bedarf ab, die Fehlervermeidung bzw. -entdeckung in die frühen Phasen der Produktentwicklung zu transferieren. BAUER ET AL. fordern in diesem Zusammenhang, dass „zur Absicherung eines neuen Systems frühzeitige Analysen notwendig [sind], um bereits in der Entwurfsphase Schwachstellen aufzudecken, Fehler zu beseitigen und mögliche Verbesserungen zu identifizieren“ [Bauer et al. 2012, S. 197]. Die gezielte Fehlerprävention in den frühen Entwicklungsphasen stellt einen proaktiven Ansatz dar, der insbesondere aufgrund monetärer und zeitlicher Einsparpotentiale enorme Vorteile gegenüber der reaktiven Fehlerbehebung in späteren Phasen aufweist.

Einem proaktiven Ansatz steht jedoch entgegen, dass in den frühen Phasen der Produktentwicklung weniger Informationen über das System, seine Eigenschaften und sein Verhalten zur Verfügung stehen als in späteren Phasen. Laut CROSTACK und KLUTE sind Informationen über Fehler und deren Folgen allerdings die Voraussetzung, um Entscheidungen zur Fehlerbehandlung zu treffen [Crostack/Klute 2008, S. 404]. Ansätze und Methoden der Zuverlässigkeitstechnik bieten durch die Bestimmung von Zuverlässigkeitskenngrößen eine Möglichkeit zur

Optimierung von technischen Systemen und basieren oftmals auf quantitativen Informationen. Solche Informationen können über Lebensdauertests von Systemen oder Einzelkomponenten ermittelt werden, bei denen diese gezielt bis zum Zeitpunkt des Fehlerauftretens getestet werden. DOROCIAK folgert, dass Methoden der Zuverlässigkeitstechnik somit erst beim Vorliegen eines detaillierten Systementwurfs angewandt werden können und Fehler erst spät erkannt werden [Dorociak 2012]. Eine effiziente Fehlerprävention muss daher durch geeignete methodische und systematische Maßnahmen bereits in der Phase des Systementwurfs ermöglicht werden, bei der auf Basis von Anforderungen die Funktionsstruktur definiert, technische Lösungen ausgewählt und eine erste physische Struktur entworfen wird [Koller/Kastrup 1998, Ponn/Lindemann 2011, VDI 2206].

Um die Eignung sowie die Zuverlässigkeit eines technischen Systems nachzuweisen, haben sich verschiedene Ansätze zur Verbesserung der Informationsverfügbarkeit herausgebildet. Diese Ansätze basieren auf der Nutzung von Daten, die aus vergleichbaren früheren Projekten bzw. Produkten im Feld vorhanden sind und für ein neues Produktentwicklungsvorhaben aus dem Feld zurückgeführt werden (Informationsrückführung). Auch die Verwendung von Daten, die auf Basis bekannter technisch-physikalischer Zusammenhänge und angenommener Belastung eines Systems im Feld das Systemverhalten noch vor Vorliegen physischer Muster simulieren, ist möglich. Da der Fokus dieser Ansätze auf dem Nachweis einer geforderten Zuverlässigkeit bei definierten Beanspruchungen im Feld liegt und nur bedingt eine Aussage darüber liefern kann, ob bei der Produktentwicklung selbst alle notwendigen Schritte durchgeführt wurden, sind diese Ansätze nicht Bestandteil der vorliegenden Arbeit.

Um bereits in frühen Phasen Aussagen über die Eignung eines Systems treffen zu können, erfolgt in der Produktentwicklung die Reifegradabsicherung, bei der regelmäßig bzw. an definierten Meilensteinen der Reifegrad des zu entwickelnden Produkts bestimmt wird[2]. Hierfür werden definierte Kriterien abgefragt, die einen Rückschluss auf eine geforderte Qualität des Gesamtsystems oder von Systembestandteilen zulassen. Der Grad der Umsetzung dieser Kriterien ermöglicht somit eine Bewertung des Reifegrads zum jeweiligen Betrachtungszeitpunkt. Die Reifegradabsicherung stellt für die Produktentwicklung eine Methode dar, den Reifegrad eines Produkts zu erfassen und bei Abweichungen Gegenmaßnahmen einzuleiten. [VDA 2009]

Während die Reifegradbewertung des Verbands der Automobilindustrie (VDA) generischen Charakter besitzt, existieren in der Praxis oftmals unternehmens- oder applikationsspezifische Methoden der Reifegradabsicherung, welche sich an unternehmerischen Gegebenheiten oder am Produkt selbst orientieren und hierfür z.T. bereits während der Akquisephase den Reifegrad bewerten. Typische Methoden sind hier DfX-Ansätze (Design for X), wie das Design for Manufacturing, welches die Herstellbarkeit des Produkts bewertet. In der Praxis haben sich hierfür in der Produktentwicklung DfX-Workshops und -Reviews unterschiedlicher Zielstellungen durchgesetzt. Trotz ihres Mehrwerts für die Erreichung eines hohen Produktreifegrads werden diese Ansätze in der vorliegenden Arbeit nicht betrachtet, da sie durch ihre unternehmens- und

[2] Der Begriff des Reifegrads im Kontext von Produkten entspricht nicht dem Reifegrad- und Fähigkeitsverständnis von Prozessen bzw. Prozessgebieten gemäß Capability Maturity Model Integration (CMMI). Da sich dieser Begriff jedoch in der Produktentwicklung durchgesetzt hat, wird er weiterführend verwendet, jedoch mit der Differenzierung zwischen Methoden der Reifegradabsicherung und dem Produktreifegrad.

applikationsspezifische Fokussierung keinen generischen Anspruch besitzen, welcher sich auf ein applikationsunspezifisches Vorgehen übertragen lässt.

Neben diesen produktorientierten Ansätzen existieren prozessorientierte Modelle, welche die Durchführung definierter Entwicklungsschritte bewerten, um hierüber auf einen Fähigkeits- bzw. Reifegrad[3] eines Prozesses zu schließen. Die Erreichung eines hohen Fähigkeits- bzw. Reifegrads soll sicherstellen, dass ein Produkt als Output des Entwicklungsprozesses systematisch und wiederholbar zu einer hohen Produktreife geführt wird, anstatt zufällig bzw. a posteriori auf Ereignisse im Entwicklungsprozess zu reagieren. Ursprünglich entwickelt im Bereich Softwareentwicklung, haben sich Reifegradmodelle wie das **C**apability **M**aturity **M**odel (CMM) [SEI 2011] oder die **S**oftware **P**rocess **I**mprovement and **C**apability D**e**termination (SPICe) [ISO/IEC 15504] branchenübergreifend durchgesetzt. Diese Modelle definieren, was ein Prozess leisten muss, um einem konkreten Reifegrad bzw. Fähigkeitsgrad zu entsprechen. Aufgrund ihres generischen, branchen-, domänen- und produktübergreifenden Charakters für Entwicklungsprozesse werden Reifegradmodelle auch als Referenzmodelle bezeichnet.

Zwischen Reifegradmodellen und Methoden zur Reifegradabsicherung besteht jedoch eine methodische Lücke. Während Reifegradmodelle den geforderten Output eines Prozesses im Sinne dessen beschreiben, was Prozesse eines bestimmten Reifegrads leisten müssen, bewerten Methoden der Reifegradabsicherung den Reifegrad eines konkreten Produkts zu einem definierten Zeitpunkt. Somit bewegt sich die Produktentwicklung im Spannungsfeld zwischen dem generischen Prozessbezug der Reifegradmodelle und einem konkreten Produktbezug der Reifegradabsicherungsmethoden. Während Reifegradmodelle Anforderungen hinsichtlich zu implementierender Prozesse vorgeben, fragen Reifegradabsicherungsmethoden Messkriterien zur Ermittlung der Produktreife ab und bewerten sie hinsichtlich ihrer Zielerreichung. Eine konsistente Beschreibung der Umsetzung von den Anforderungen bis hin zur Bewertung des Produktreifegrads (das „Wie“), definieren sie jedoch nicht.

Zur Schließung der identifizierten Lücke werden in der Produktentwicklung Vorgehensmodelle angewandt. Diese basieren oftmals auf dem V-Modell [VDI 2206] bzw. Derivaten hiervon. Die zuvor aufgeführten Probleme der Produktentwicklung verdeutlichen allerdings, dass trotz generischer Reifegradmodelle, etablierter Vorgehensmodelle und Methoden zur Reifegradabsicherung Handlungsbedarf besteht. Dies betrifft insbesondere den Umgang mit Schnittstellen zwischen Domänen bzw. entsprechenden Systemelementen und den resultierenden Wechselbeziehungen. Das Ziel besteht somit darin, bereits in frühen Phasen der Produktentwicklung Wechselbeziehungen zwischen Systemelementen zu identifizieren und zu analysieren, um Verbesserungspotentiale aufzuzeigen und zu implementieren. Dabei muss sich ein entsprechender Ansatz in bestehende Vorgehensmodelle integrieren lassen und Referenzmodelle und Methoden der Reifegradabsicherung miteinander synchronisieren. Um wie gefordert zudem technische Systeme bereits in frühen PEP-Phasen bewertbar und gleichzeitig die Komplexität beherrschbar zu machen, müssen die verfügbaren Informationen über das System aufbereitet und zielführend genutzt werden.

[3] Das CMMI unterscheidet zwischen Reifegraden und Fähigkeitsgraden, wobei „Reifegrade [...] die Verbesserungen einer Organisation bezogen auf einen Satz von Prozessgebieten [charakterisieren], Fähigkeitsgrade die Verbesserungen bezogen auf ein einzelnes Prozessgebiet“ [SEI 2011, S. 39]. Demzufolge wird fortführend der Begriff Fähigkeit verwendet, sofern nur Prozesse eines Prozessgebiets wie der Entwicklung betrachtet werden.

Im Bereich des Komplexitätsmanagements schlagen LINDEMANN ET AL. daher drei Strategien für den Umgang mit Komplexität vor [Lindemann et al. 2009, S. 31], welche für die Zielstellung dieser Arbeit von unmittelbarer Bedeutung sind:

1. Erfassung und Bewertung von Strukturen komplexer Systeme
2. Vermeidung und Reduzierung von Komplexität
3. Management und Beherrschung von Komplexität

Die Erfassung von Systemstrukturen ist die grundlegende Bedingung, um komplexe technische Systeme beherrschen zu können. Hierfür haben sich Ansätze der Systemmodellierung durchgesetzt, welche die Elemente technischer Systeme sowie ihre Relationen erfassen und gemäß den Konventionen der Graphentheorie darstellen. Bestehende Ansätze der Systemmodellierung fokussieren allerdings weder auf die für den Systementwurf zentralen Funktionen, noch sehen sie eine Schnittstelle zu Reifegradmodellen oder Methoden der Reifegradabsicherung vor.

Diesen Zusammenhang veranschaulicht Abb. 4. Im Spannungsfeld zwischen Reifegradmodellen, welche Anforderungen mit generischem Prozessbezug bereitstellen, und den Methoden der Reifegradabsicherung, die Messkriterien mit einem konkreten Produktbezug vorgeben, besteht mangels der Konsistenz der bestehenden Ansätze eine methodische Lücke. Das Ziel der vorliegenden Arbeit liegt daher in der Schließung dieser Lücke durch einen konsistenten Ansatz, welcher Reifegradmodelle mit Methoden der Reifegradabsicherung in frühen Phasen der Produktentwicklung zusammenführt.

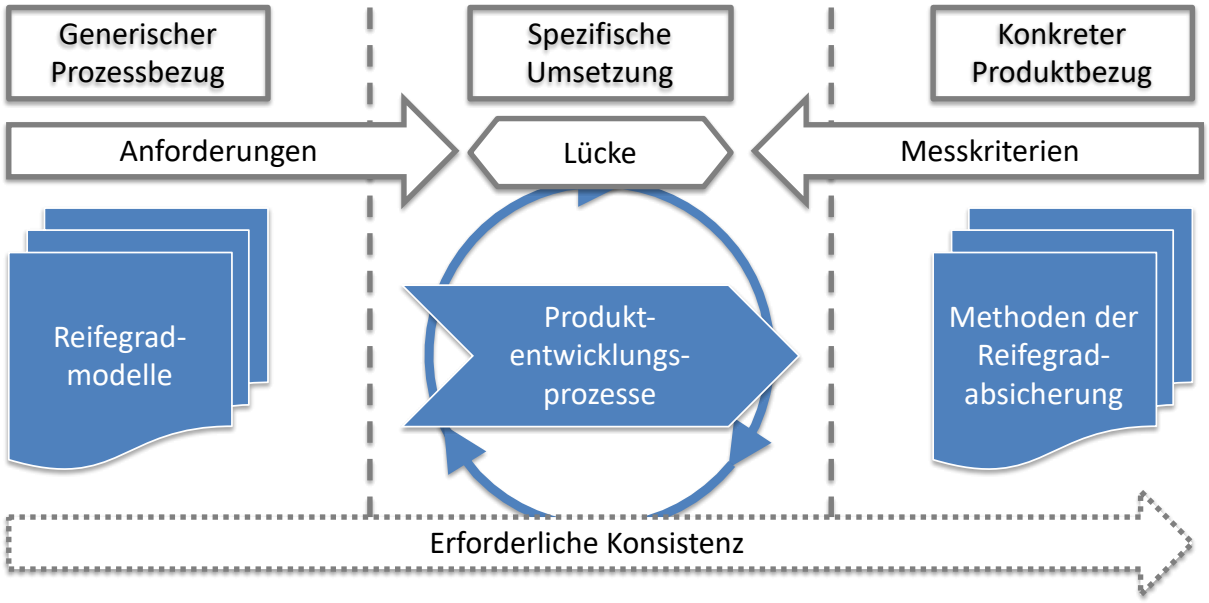


Abb. 4: Methodische Lücke zwischen Reifegradmodellen, PEP und Methoden der RGA

1.3 Lösungskonzept

Zur Schließung der methodischen Lücke zwischen Reifegradmodellen mit generischem Prozessbezug und Methoden der Reifegradabsicherung mit konkretem Produktbezug muss in der Produktentwicklung angesetzt werden. Hier ist es erforderlich, eine gemeinsame Basis zu definieren, die der Produktentwicklung konkrete Hilfestellung liefert und die Bewertung des Produktreifegrads unterstützt. Zu diesem Zweck wird nachfolgend ein Vorgehenskonzept entwickelt, welches sich aus drei Bausteinen zusammensetzt: einer Schrittfolge zur System-

konkretisierung und -analyse (i), einer entwicklungsbegleitenden Systemmodellierung (ii) und einer an Lösungsebenen ausgerichteten Bewertung des Produktreifegrads (iii). Die Grundpfeiler des Ansatzes stellen dabei das bedarfsspezifisch angepasste DeCoDe-Systemmodell (Demand Compliant Design, vgl. [Sitte/Winzer 2011]) sowie eine im Rahmen dieser Arbeit entwickelte Schrittfolge dar. Das Vorgehenskonzept ergänzt die DeCoDe-Methodik, welche technische Systeme über Anforderungen, Funktionen, Komponenten und Prozesse beschreibt, um Funktionskategorien sowie definierte Relationen und Schnittstellen zwischen Systemelementen. Eine abgeleitete Schrittfolge zeigt, wie technische Systeme sukzessiv konkretisiert und bezüglich ihrer Relationen und Schnittstellen analysiert werden können. Dabei erfolgt eine parallele Systemmodellierung, welche an definierten Meilensteinen die kriterienbasierte Bewertung des Systementwurfs ermöglicht.

Das in dieser Arbeit entwickelte Vorgehenskonzept zur modellbasierten Systemkonkretisierung und -analyse ermöglicht einerseits dedizierte Angaben zum Prozess der Modellbildung. Andererseits wird der Produktentwicklungsprozess durch eine Schrittfolge systematisiert und an Lösungsebenen ausgerichtet. Über dieses Vorgehen wird aufgezeigt, welche Schritte wann durchzuführen sind. Gleichzeitig ermöglicht die entwicklungsbegleitende Systemmodellierung die Erfassung des Produktreifegrads auf Basis der Lösungsebenen an definierten Meilensteinen und die Umsetzung korrektiver Maßnahmen bei erkannten Abweichungen. Innerhalb der frühen Phasen der Produktentwicklung fokussiert der Ansatz den Systementwurf gemäß des V-Modells [VDI 2206].

Im Detail setzt sich das Vorgehenskonzept aus drei Bestandteilen zusammen, die iterativ im Produktentwicklungsprozess durchlaufen werden. Die Schrittfolgen bei der Systemkonkretisierung und -analyse (i) beschreiben den generellen Ablauf von Tätigkeiten in der Produktentwicklung. Dabei erfolgt eine Unterteilung in die Schrittfolge zur Systemkonkretisierung des Lösungsraums und die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse. Die Schrittfolge für die Systemmodellierung (ii) zeigt auf, welche Systemelemente und welche Relationen und Schnittstellen zwischen ihnen beschrieben werden. Dabei wird zwischen einer Symbol-, Graphen- und Matrizendarstellung unterschieden. Die Schrittfolge zur Ermittlung des Produktreifegrads (iii) gibt vor, wie basierend auf der parallel zur Produktentwicklung erfolgten Systemmodellierung an vorgegebenen Meilensteinen der Produktreifegrad ermittelt werden kann. Das Vorgehenskonzept setzt dabei Forderungen des CMMI-Reifegradmodells zur Erreichung eines hohen Prozessreifegrads um.

Dieser prinzipielle Lösungsansatz unterstellt verschiedene Arbeitshypothesen, welche sich aus den dargestellten Herausforderungen bei der Entwicklung komplexer technischer Systeme und der identifizierten wissenschaftlichen Lücke aggregieren lassen. Die Arbeitshypothesen sind getroffene Annahmen, welche durch den Ansatz zu verifizieren bzw. zu falsifizieren sind und betreffen verschiedene Aspekte der zugrundeliegenden Forschungsfragen.

So wird angenommen, dass komplexitäts- und zuverlässigkeitsrelevante Informationen über technische Systeme in Form von Relationen und Schnittstellen zwischen Systemelementen bereits in frühen Produktentwicklungsphasen zur Verfügung stehen bzw. identifiziert werden können, mangels einer geeigneten Methodik jedoch nicht oder nur unzureichend genutzt werden. Die Erfassung und Darstellung bzw. Modellierung von Systemelementen und ihrer Relationen und Schnittstellen bildet dabei die Grundvoraussetzung der Komplexitätsbeherrschung. Analog

hierzu kann nur durch die Berücksichtigung zuverlässigkeitsrelevanter Informationen, d.h. durch die Identifizierung von Störgrößen, sichergestellt werden, dass deren Einfluss bewertet und anschließend geeignete Maßnahmen zum Umgang mit diesen getroffen werden. Die Grundlage hierfür stellt die Systemmodellierung über die standardisierte Beschreibung und -visualisierung des Systems dar. Die Systemmodellierung bildet die gemeinsame Sprache bei der domänenübergreifenden Produktentwicklung. Aufgrund ihrer zentralen Bedeutung für den vorgestellten Ansatz stellen Funktionen im Systemmodell die zentrale Systemsicht dar.

Bezogen auf das Vorgehenskonzept wird die Hypothese aufgestellt, dass Reproduzierbarkeit und Transparenz von Entwicklungsergebnissen nur über ein Vorgehenskonzept sichergestellt werden können. Dementsprechend ist eine ausschließliche Beschreibung der Modellierungslogik und der Anwendung von Methoden der Reifegradabsicherung und Berücksichtigung von Best Practices der Reifegradmodelle nicht ausreichend, um reproduzierbar Entwicklungsergebnisse in der geforderten Transparenz zu generieren. Vielmehr ist eine Schrittfolge erforderlich, welche methodische Leitplanken definiert, ohne inhaltliche Einschränkungen vorzunehmen oder das Wissen der entwickelnden Personen ersetzen zu wollen.

Die genutzte Beschreibung des Systems über die DeCoDe-Methodik stellt eine mögliche Art der Systemmodellierung dar. Es existieren jedoch auch weitere Modellierungssprachen und -methodiken, wie beispielsweise die Systems Modeling Language (SysML) [Delligatti 2014]. Die Anwendung von DeCoDe erfolgt daher unter der Hypothese, dass die Modellierungsmethodik zwar eine für das Vorgehenskonzept erforderliche Festlegung darstellt, das Vorgehenskonzept jedoch nicht hierauf beschränkt ist. Es wird daher unterstellt, dass mit moderatem Adaptionsaufwand das Vorgehenskonzept auf andere Modellierungsansätze übertragbar ist und somit universalen Charakter besitzt. Gleiches gilt für die genutzte Modellierungssoftware. Diese stellt ein notwendiges und hilfreiches Werkzeug dar, ist jedoch austauschbar, sofern geforderte Funktionalitäten vorhanden sind.

Innerhalb des Vorgehenskonzepts ermöglichen Lösungsebenen die Bewertung der Produktreife anhand vordefinierter Meilensteine und somit die Kopplung mit Methoden der Reifegradabsicherung. Hingegen stellt der Ansatz selbst die Umsetzung von Best Practices von Reifegradmodellen dar, indem deren Vorgaben zu einem inhärenten Bestand des Vorgehenskonzept gemacht werden.

In Tab. 3 werden die postulierten Hypothesen H01…H09 nach Themenfeldern zusammengefasst dargestellt.

Tab. 1: Themenfelder und Hypothesen

Themenfeld	Hypothese
Informations-verfügbarkeit	Komplexitäts- und zuverlässigkeitsrelevante Informationen über technische Systeme stehen bereits in frühen Produktentwicklungsphasen zur Verfügung (H01)
Komplexität	Die Erfassung und Modellierung von Systemelementen, Relationen und Schnittstellen ermöglicht die Beherrschung von Komplexität (H02)
Zuverlässigkeit	Die gezielte Identifizierung von Störgrößen stellt die Grundvoraussetzung für die Entwicklung eines zuverlässigen Systems dar (H03)
Systemmodell	Ein Systemmodell stellt die standardisierte Visualisierung von Systemelementen, Relationen und Schnittstellen bei der domänenübergreifenden Produktentwicklung sicher (H04)
Systemmodell	Funktionen stellen die zentrale Sicht innerhalb des Systemmodells dar (H05)
Vorgehens-konzept	Ein auf Lösungsebenen basierendes Vorgehenskonzept ermöglicht eine zielgerichtete Erfassung des Produktreifegrads (H06)
Vorgehens-konzept	Eine Schrittfolge stellt die Reproduzierbarkeit und Transparenz von Entwicklungsergebnissen sicher (H07)
Vorgehens-konzept	Das Vorgehenskonzept besitzt universellen Charakter und ist unabhängig von der Modellierungsmethodik (H08)
Vorgehens-konzept	Über das Vorgehenskonzept werden Best Practices des Reifegradmodells umgesetzt (H09)

Aus den jeweiligen Hypothesen lassen sich Anforderungen an den Ansatz ableiten, welche den inhaltlichen Lösungsraum des Ansatzes definieren. Während die Anforderungen, welche sich unmittelbar aus den diskutierten Modellen mit generischem Prozessbezug bzw. den Methoden mit konkretem Produktbezug ergeben, abhängig vom jeweiligen Modell bzw. der Methode und somit lösungsgebunden sind, geben die aus den Hypothesen abgeleiteten Anforderungen eine lösungsneutrale Zielstellung an das Vorgehenskonzept vor.

Bezogen auf die Hypothese zur Verfügbarkeit relevanter Informationen in frühen Phasen der Produktentwicklung wird gefordert, dass diese Informationen zunächst kategorisiert werden müssen. Die projektunabhängige Kategorisierung stellt eine generische Definition dieser Informationen im Sinne des Vorgehenskonzepts dar. Während die zu erfassenden Informationen sich demnach von Projekt zu Projekt unterscheiden können, bilden die Kategorien eine Vorgabe, welche Art von Information zu erfassen ist. Im Kontext der verfügbaren Informationen wird zudem die Hypothese aufgestellt, dass über die Erfassung und Modellierung von Systemelementen, Relationen und Schnittstellen die Komplexitätsbeherrschung ermöglicht wird. Die Erfassung und Modellierung dienen der Nutzbarkeit der Informationen für die Produktentwicklung. Entsprechend lautet die abgeleitete Anforderung, dass eben jene Systembestandteile zu erfassen und modellieren sind.

Bezogen auf Störgrößen wird angenommen, dass deren Identifizierung eine Voraussetzung ist, um zuverlässige Produkte zu entwickeln. Hieraus lässt sich ableiten, dass Kriterien zur Identifizierung von Störgrößen zu benennen sind. Dies stellt ein einheitliches Verständnis von Störgrößen sicher und ermöglicht deren Betrachtung im Systemkontext. Die Visualisierung der verfügbaren Informationen im Systemkontext (Systemelemente, Relationen, Schnittstellen) kann

dabei über ein Systemmodell standardisiert werden. Dieses muss festgelegt und, sofern erforderlich, auf die Belange der funktionsorientierten Produktentwicklung angepasst werden. Da Funktionen hierfür die zentrale Systemsicht darstellen, muss eine geeignete Visualisierung gewählt werden, die der grundsätzlichen Bedeutung der Funktionssicht entspricht.

Die Hypothese, dass ein lösungsebenenbasiertes Vorgehenskonzept die Ermittlung des Produktreifegrads ermöglicht, resultiert in der Anforderung, dass sowohl das Vorgehenskonzept als auch das Systemmodell zur Definition von Lösungsebenen beitragen. Hierüber sind Meilensteine zur Messung des Reifegrads zu etablieren. Wird ein geforderter Reifegrad nicht erreicht, sind geeignete Gegenmaßnahmen zu definieren und umzusetzen. Als Bestandteil des Vorgehenskonzepts muss dabei eine Schrittfolge entwickelt werden, welche einerseits die Transparenz im Entwicklungsprozess sicherstellt und andererseits durch konkrete, aber dennoch lösungsneutrale Vorgaben die Reproduzierbarkeit von Entwicklungsergebnissen ermöglicht. Da angenommen wird, dass das gesamte Vorgehenskonzept einen universellen Charakter besitzt und somit nicht an das DeCoDe-Modell gebunden ist, müssen die genutzten Systemsichten generisch gewählt werden, sodass eine Kompatibilität mit anderen Modellierungsmethoden gegeben ist.

Die letzte Hypothese aus dem Themenfeld des Vorgehenskonzepts betrifft die Einbindung von Best Practices aus dem Reifegradmodell. Hierfür wird postuliert, dass über das Vorgehenskonzept Best Practices umgesetzt werden. Um die These zu verifizieren, müssen Best Practice-Ansätze herausgearbeitet werden und es muss dargestellt werden, wie diese im Vorgehenskonzept implementiert sind.

In Tab. 2 sind die abgeleiteten Anforderungen A01…A09 den Hypothesen H01…H09 gegenübergestellt.

Tab. 2: Anforderungen an das Vorgehenskonzept

Hypothese	Anforderung
Komplexitäts- und zuverlässigkeitsrelevante Informationen über technische Systeme stehen bereits in frühen Produktentwicklungsphasen zur Verfügung (H01)	Komplexitäts- und zuverlässigkeitsrelevante Informationen müssen kategorisiert werden, um identifiziert werden zu können (A1)
Die Erfassung und Modellierung von Systemelementen, Relationen und Schnittstellen ermöglicht die Beherrschung von Komplexität (H02)	Systemelementen, Relationen und Schnittstellen müssen erfasst und modelliert werden (A2)
Die gezielte Identifizierung von Störgrößen stellt die Grundvoraussetzung für die Entwicklung eines zuverlässigen Systems dar (H03)	Kriterien für die Identifizierung von Störgrößen müssen definiert werden (A03)
Ein Systemmodell stellt die standardisierte Visualisierung von Systemelementen, Relationen und Schnittstellen bei der domänenübergreifenden Produktentwicklung sicher (H04)	Ein geeignetes Systemmodell muss ausgewählt und bei Bedarf adaptiert werden (A04)
Funktionen stellen die zentrale Sicht innerhalb des Systemmodells dar (H05)	Für die funktionsorientierte Produktentwicklung müssen geeignete Darstellungsformen gewählt werden, welche der Bedeutung der Funktionen Rechnung tragen (A05)
Ein auf Lösungsebenen basierendes Vorgehenskonzept ermöglicht eine zielgerichtete Erfassung des Produktreifegrads (H06)	Vorgehenskonzept und Systemmodell müssen darauf ausgerichtet sein Lösungsebenen zur Definition geeigneter Meilensteine zu nutzen, den Reifegrad zu bewerten und bei Abweichungen vom geforderten Reifegrad Gegenmaßnahmen implementieren zu können (A06)
Eine Schrittfolge stellt die Reproduzierbarkeit und Transparenz von Entwicklungsergebnissen sicher (H07)	Es ist eine Schrittfolge zu entwickeln, welche wesentliche Entwicklungstätigkeiten im Sinne der Transparenz benennt und durch ihren generischen Ansatz die Reproduzierbarkeit von Entwicklungsergebnissen sicherstellt (A07)
Das Vorgehenskonzept besitzt universellen Charakter und ist unabhängig von der Modellierungsmethodik (H08)	Das Vorgehensmodell muss auf generischen Systemsichten beruhen (A08)
Über das Vorgehenskonzept werden Best Practices des Reifegradmodells umgesetzt (H09)	Best Practices des Reifegradmodells sind herauszuarbeiten und im Vorgehenskonzept umzusetzen (A09)

Die vorgestellten Hypothesen und die abgeleiteten Anforderungen werden im Fazit aufgegriffen. Hierbei erfolgt die Bewertung, ob einerseits die Anforderungen umgesetzt und andererseits die Hypothesen zutreffend sind.

1.4 Aufbau der Arbeit

Zur Beherrschung der skizzierten Problemstellung müssen die wesentlichen Rahmenbedingungen für den Kontext der Arbeit herausgearbeitet werden, unter denen das Vorgehensmodell

entwickelt wird. Dies erfolgt in Kapitel 2. Im Kapitel 2.1 werden die frühen Phasen der Produktentwicklung definiert und von anderen Phasen abgegrenzt. Zudem wird auf Besonderheiten dieser Phase eingegangen. Im Kapitel 2.2 wird die Bedeutung der Informationsverfügbarkeit für die Produktentwicklung behandelt. Hier schließt die Betrachtung der Komplexität als charakteristisches Merkmal technischer Systeme in Kapitel 2.3 an. Anschließend wird in Kapitel 2.4 aufgezeigt, welche Randbedingungen für das Themenfeld der Zuverlässigkeit resultieren.

Im Kapitel 3 wird der Stand der Wissenschaft und Forschung für bestehende Ansätze zur Lösung der Problemstellung dargestellt. Dabei wird in Kapitel 3.1 zwischen Fähigkeits- und Reifegradmodellen mit generischen Prozessbezug und Methoden der Reifegradabsicherung mit konkreten Produktbezug differenziert. Für die Fähigkeits- und Reifegradmodelle als auch für die Methoden der Reifegradabsicherung werden Bewertungskriterien herausgearbeitet und Anforderungen abgeleitet, welche ein Vorgehenskonzept erfüllen muss. In Kapitel 3.2 wird das Komplexitätsmanagement vorgestellt. Hierbei liegt der Fokus auf den Definitionen des Komplexitätsmanagements sowie generellen Optionen zum Umgang mit Komplexität in der Produktentwicklung. Im Kapitel 3.3 wird die Systemmodellierung behandelt und über die Vorstellung der Graphentheorie (Kap. 3.3.1) allgemeingültige Vorgaben aufgezeigt. Anschließend wird dargelegt, wie eine Systemmodellierung auf Basis des DeCoDe umgesetzt werden kann (Kap. 3.3.2).

In Kap. 4 erfolgt die Vorstellung des entwickelten Ansatzes. Hierfür wird zunächst das Vorgehenskonzept in den Produktentwicklungsprozess gemäß des V-Modells eingeordnet (Kap. 4.1) und der Aufbau des Vorgehenskonzepts vorgestellt (Kap. 4.2). Auf detaillierter Ebene werden hierfür dessen Bestandteile beschrieben, d.h. die Systemmodellierung (Kap. 4.2.1), die Schrittfolge zur Systemkonkretisierung und -analyse (Kap. 4.2.2) und die Schrittfolge zur Ermittlung des Produktreifegrads (Kap. 4.2.3).

Im Kap. 5 wird das Vorgehenskonzepts am Beispiel einer Linearasynchronmaschine validiert. Die hieraus generierten Erkenntnisse zur Eignung des Ansatzes als Beitragsleister zur Beherrschung der dargelegten Probleme in der Produktentwicklung und zur Ableitung eines zukünftigen Forschungsbedarfs werden in Kap. 6 zusammengefasst.

2 Rahmenbedingungen der Produktentwicklung

2.1 Frühe Phasen der Produktentwicklung

Die Herausforderungen bei der Entwicklung technischer Systeme durch die Anforderung zuverlässige Produkte trotz räumlicher und funktionaler Integration sowie steigender Komplexität zu entwickeln, sind prinzipiell für alle Phasen des Produktentwicklungsprozesses relevant. Für die frühen Phasen der Produktentwicklung existieren jedoch Besonderheiten, die sich einerseits aus der eingeschränkten Informationsverfügbarkeit und andererseits aus der guten Adaptierbarkeit des Systems auf Basis erkannter problematischer Wechselbeziehungen ergeben. Hierfür ist zu klären, wie sich frühe Phasen von nachfolgenden Phasen der Produktentwicklung abgrenzen.

EHRLENSPIEL beschreibt die Phasen der Produktentwicklung im Pyramidenmodell der Produktkonkretisierung ebenso wie PONN/LINDEMANN sequentiell als funktionelle, prinzipielle physikalische, gestalterische und stoffliche sowie fertigungs- und montagetechnische Lösungsmöglichkeiten [Ehrlenspiel 2009, vgl. Abb. 5].

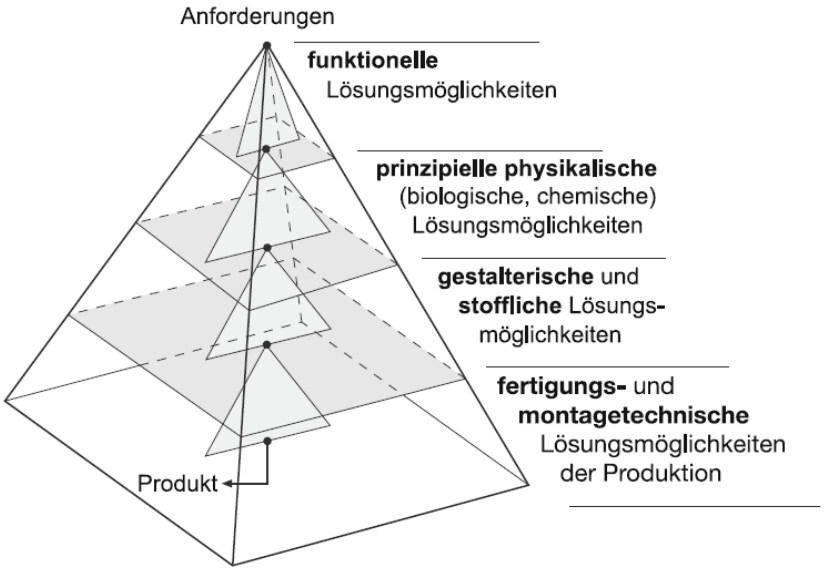


Abb. 5: Pyramidenmodell der Produktkonkretisierung [Ponn/Lindemann 2011, S. 24]

Hierbei bilden Funktionen den ersten Schritt einer auf Anforderungen basierenden Produktentwicklung. Sind funktionelle Lösungsmöglichkeiten definiert, werden ihnen prinzipielle physikalische Lösungsmöglichkeiten zugewiesen. Diese prinzipiellen physikalischen Lösungsmöglichkeiten werden auch als Prinziplösungen beschrieben und untersetzen Elementarfunktionen mit physikalischen Effekten [Koller/Kastrup 1998]. Die gestalterischen und stofflichen Lösungsmöglichkeiten umfassen insbesondere Komponenten. Ihnen folgen als letzter Schritt fertigungs- und montagetechnische Lösungsmöglichkeiten, welche die Herstellung des Systems über Prozesse beschreiben. Das Ergebnis der Produktentwicklung stellt das Produkt dar. Die frühen Phasen der Produktentwicklung können im Pyramidenmodell der Produktkonkretisierung dabei den Phasen zugewiesen werden, in denen grundlegende Entwicklungsentscheidungen getroffen werden. Dies umfasst die funktionellen und prinzipiellen physikalischen Lösungsmöglichkeiten sowie, je nach Detaillierungsgrad, auch gestalterische und stoffliche Lösungsmöglichkeiten [Ponn/Lindemann 2011, S. 24, nach Ehrlenspiel 2009, S. 37]. Die fertigungs- und montagetechnischen Lösungsmöglichkeiten der Produktion sind kein Bestandteil

der frühen Phasen der Produktentwicklung, da zu diesem Zeitpunkt das Produktkonzept vollständig entwickelt ist und bereits die prozessuale Umsetzung in der Fertigung fokussiert wird.

Eine präzisere Abgrenzung zwischen der Konzeptionierung in frühen Phasen der Produktentwicklung und nachfolgenden Phasen bietet das V-Modell gemäß VDI 2206. Diesem sequentiellen, auf Anforderungen basierenden Ansatz für die Entwicklung mechatronischer Systeme liegt der sogenannte Makrozyklus zugrunde, dargestellt in Abb. 6. Die frühen Phasen der Produktentwicklung entsprechen dem Systementwurf, der vom domänenspezifischen Entwurf und der Systemintegration abgegrenzt ist. Während der Systementwurf domänenübergreifend entwickelt wird, erfolgt für den nachfolgenden domänenspezifischen Entwurf eine Aufteilung auf die Domänen Maschinenbau, Elektrotechnik und Informationstechnik. Die Systemintegration führt die domänenspezifischen Entwürfe wieder zusammen und stellt über den Abgleich mit den Anforderungen die Eigenschaftsabsicherung dar. Eine Modellbildung und -analyse ist für alle Phasen des V-Modells vorgesehen. Mit dem Durchlauf des Makrozyklus wird somit basierend auf Anforderungen das Produkt entwickelt [VDI 2206].

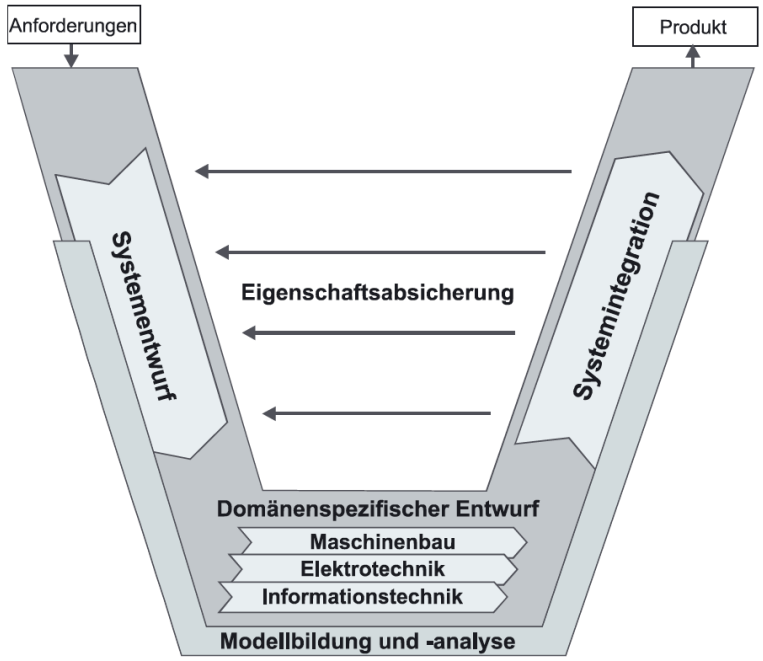


Abb. 6: V-Modell als Makrozyklus [VDI 2206, S. 29]

Der fokussierte Systementwurf gliedert sich in Tätigkeiten auf, deren Ergebnis ein Lösungskonzept darstellt, siehe Abb. 7. Hierbei erfolgt zunächst die Abstraktion der Anforderungen, um die grundlegende Zielstellung zu identifizieren. Darauf aufbauend wird eine Funktionsstruktur erstellt, bei der eine Gesamtfunktion in Teilfunktionen aufgeschlüsselt und durch Wirkprinzipien untersetzt. Die durch weitere Konkretisierung entstehenden Lösungsvarianten werden anschließend bewertet, um ein Lösungskonzept auswählen zu können. [VDI 2006, S. 31f]

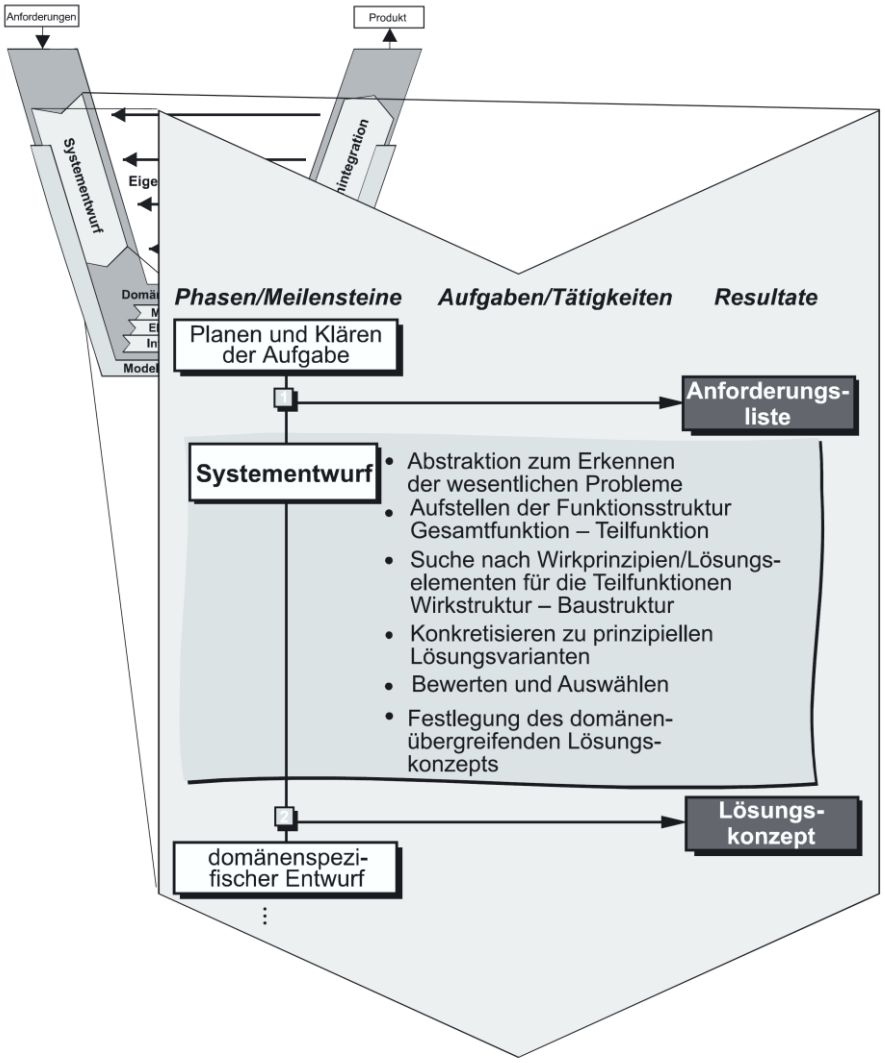


Abb. 7: Tätigkeiten beim Systementwurf [VDI 2206, S. 32]

Im Vergleich mit dem Pyramidenmodell entspricht die Funktionsstruktur des V-Modells den funktionellen Lösungsmöglichkeiten von PONN/LINDEMANN bzw. den Elementarfunktionen von KOLLER/KASTRUP, während die Wirkprinzipien den prinzipiellen physikalischen Lösungsmöglichkeiten bzw. den Prinziplösungen entsprechen. Als Input setzen beide Ansätze Anforderungen voraus, deren Erhebung nicht Bestandteil dieser Arbeit ist.

Neben der VDI 2206 schließen sich weitere Ansätze dem prinzipiellen Vorgehen des V-Modells an [vgl. VDI 2221, S. 9] oder definieren ein Vorgehen, welches auf dem V-Modell beruht bzw. dieses adaptiert. So entwickelte BENDER das speziell für die Entwicklung mechatronischer Produkte ausgelegte 3-Ebenen-Vorgehensmodell [Bender 2005, S. 44ff]. Dieses drei Stränge (Mechanik, Hardware und Software) umfassende Vorgehen unterteilt das V-Modell in die System-, Subsystem- und Komponentenebene. Das Entwicklungsvorgehen sieht auf der Systemebene zunächst eine System-Anforderungs-Analyse vor, auf die ein integraler Systementwurf folgt. Auf Subsystemebene erfolgt eine Unterteilung in die Disziplinen der Mechatronik, wobei im Schritt der IT-Anforderungsanalyse und dem IT-Entwurf Hard- und Software gemeinsam betrachtet werden. Ein domänenspezifischer Entwurf beendet die Subsystemebene und geht in den Feinentwurf bzw. die Ausarbeitung in der jeweiligen Disziplin der Komponentenebene über. Das 3-Ebenen-Vorgehensmodell verdeutlicht damit den fließenden Übergang der

Entwurfsphase von der Systemebene bis zur Komponentenebene. Die zentrale Tätigkeit der frühen Entwicklungsphasen ist das Erstellen eines Funktionsnetzwerks. Dieses besteht aus Funktionen, ihren Schnittstellen und Informationsflüssen zwischen Funktionen und bildet die logische Systemarchitektur ab [Bender 2005, S. 46].

Gleichzeitig weisen GERICKE ET AL. auf die Grenzen sequentieller Abläufe wie den des V-Modells hin, da diese zwar eine logische Schrittfolge bieten, in der Realität jedoch aufgrund von äußeren Rahmenbedingungen wie Materialbeschaffungszeiten oder Terminvorgaben nur bedingt eingehalten werden können. Demzufolge müssen Modelle auch Iterationsschleifen oder die Möglichkeit der Phasenüberlagerung vorsehen [Gericke et al. 2021, S. 80ff].

Zusammenfassend sind frühe Phasen der Produktentwicklung dadurch charakterisiert, dass auf Basis von Anforderungen und ihrer Analyse ein Systementwurf abgeleitet wird, der zunächst integral vom System ausgehend in Subsystem- und schließlich domänenspezifische Entwürfe aufgegliedert wird. Eine genauere Abgrenzung der frühen Phasen ermöglicht die Differenzierung zwischen funktionellen, prinzipiellen physikalischen und baulichen Lösungen gemäß dem Pyramidenmodell der Produktentwicklung [Ponn/Lindemann 2011] und der resultierenden Schrittfolge im Konstruktionsprozess. Im Verständnis der vorliegenden Arbeit sind die frühen Phasen der Produktentwicklung somit wesentlich durch die Festlegung einer Funktionsstruktur zur prinzipiellen Erfüllung der Anforderungen, das Festlegen von Prinziplösungen sowie die Erstellung einer prinzipiellen gestalterischen Lösung charakterisiert, wobei mit steigender Granularität die System-, Subsystem- und in Abhängigkeit spezifischer Entwicklungsprojekte auch die Komponentenebene durchlaufen wird. Dies schließt nicht aus, dass es zu Überschneidungen mit vor- oder nachgelagerten Phasen kommt oder Iterationsschleifen innerhalb der Entwurfsphase genutzt werden.

Gemäß dem dargelegten Verständnis sind Anforderungen die Eingangsgröße der Produktentwicklung, ihre Erfassung und Analyse ist jedoch dem eigentlichen Systementwurf vorgeschaltet und somit nicht Bestandteil dieser Arbeit. Diese Auffassung deckt sich mit dem Verständnis, dass die Entwicklung einen „Satz von Prozessen [darstellt], der Anforderungen in festgelegte Merkmale oder in die Spezifikation eines Produkts, eines Prozesses oder eines Systems umwandelt“ [DIN EN 61160:2006, S. 8]. Gleichwohl sei darauf hingewiesen, dass bedingt durch die hohe Bedeutung vollständiger, realistischer und widerspruchsfreier Anforderungen Ansätze aus dem Bereich des Requirements Engineerings [Mistler 2021, S. 20ff] eine wichtige Schnittmenge zur Produktentwicklung darstellen und während der Produktentwicklung konsistent betrachtet werden sollten.

Die Anforderungserfassung als Eingangsgröße der Produktentwicklung findet demnach keine Berücksichtigung. Dennoch wird der Teil des technischen Änderungsmanagements miteinbezogen, der einen Informationsrückfluss zu Anforderungen vorsieht, wenn auf Basis von Erkenntnissen in den frühen Entwicklungsphasen Änderungen oder Ergänzung der Anforderungsbasis notwendig werden. Dies gilt insbesondere für Erkenntnisse aus der Schnittstellenanalyse bzw. aus dem Feld. Hierüber entsteht eine Verknüpfung zu Ansätzen des Engineering Change Managements (ECM), welche den Umgang mit Produktänderungen definieren, wobei hierunter „Modifikationen an Einzelteilen, Komponenten, Produkten, Software, Stücklisten, produktbeschreibenden Dokumenten wie Zeichnungen und technische Spezifikationen, die im Rahmen des Produktentwicklungsprozesses bereits freigegeben sind (Design Freeze)“ [Lashin 2021, S.

921] verstanden werden. Wie definiert bezieht sich das ECM auf einen freigegebenen Entwicklungsstand. Oftmals werden erforderliche Änderungen auch erst nach dem sogenannten Start of Production (SOP, dt. Produktionsanlauf) während der Produktnutzung bekannt, sodass diese Ansätze nicht bzw. nur bedingt für die frühen Phasen der Produktentwicklung greifen. Vertiefend sei an dieser Stelle auf den Ansatz zur Anforderungsaktualisierung in der Produktentwicklung [Schlund 2011] verwiesen.

2.2 Informationsverfügbarkeit

Die geringe Informationsverfügbarkeit stellt in den frühen PEP-Phasen ein grundsätzliches Problem dar. Die Analyse von Ausfallursachen zeigt, dass bei technischen Systemen durch Schnittstellencharakteristik und Funktionsintegration bedingte Probleme auftreten, die sich auf eine unzureichende Informationsverfügbarkeit zurückführen lassen. Die mangelnde Berücksichtigung aller Produktlebenszyklusphasen [Schlund et al. 2009, S. 57] hebt zudem den zeitlichen Charakter der Informationsverfügbarkeit hervor. Bezogen hierauf ergibt sich aus der „Rule of Ten“ [Bertsche et al. 2009, S. 200] die Notwendigkeit, Fehler möglichst frühzeitig zu vermeiden. Zudem stellt die Produktentwicklung die wesentliche Phase der Fehlerentstehung dar [Westkämper 2006, S. 131]. Entsprechend müssen im Produktentwicklungsprozess „die richtigen Informationen zum rechten Zeitpunkt zur Verfügung stehen“ [Ehrlenspiel 2009, S. 192].

Zur genaueren Begriffsabgrenzung werden Informationen zunächst im Bedeutungskontext betrachtet. Nach WILD stehen Zeichen, Daten, Informationen und Wissen in einer „aufsteigenden Begriffshierarchie“ [Wild 2007, S. 7]. Werden Zeichen zu Zeichenfolgen zusammengefasst und folgen dabei einer Ordnungsregel, stellen sie Daten dar, die in einen Bedeutungskontext Informationen ergeben [Meinsen 2003, S. 18]. Die Aufnahme und Interpretation von Informationen durch den Menschen lässt dann Wissen entstehen [ebd., S. 33f]. Das Ziel ist somit, im Rahmen der Produktentwicklung den Entwicklern Daten im geforderten Bedeutungskontext bereitzustellen, damit diese als Informationen nutzbar sind.

Die Herausforderung besteht darin, in frühen Entwicklungsphasen Informationen über das System für die Bewertung des Reifegrads sowie abgeleitete Entwicklungsentscheidungen zu generieren. Unter Berücksichtigung der Erkenntnisse zur Fehlerentstehung [Westkämper 2006] sind somit die Phasen kritisch, in denen ohnehin aufgrund des niedrigen Produktreifegrads des Systems wenig Informationen zur Verfügung stehen. Durch ein geeignetes Wissensmanagement kann die Informationsverfügbarkeit in frühen Phasen zwar verbessert werden, grundsätzlich stehen jedoch einsatz- und anwendungsrelevante Informationen wie das Ausfallverhalten mit Daten zu Fehlerart und -häufigkeit erst zur Verfügung, wenn sich ein Produkt im Feld befindet. Bei der zeitlichen Betrachtung der Informationsverfügbarkeit über den PLC wächst das Wissen über ein Produkt an, während die Entscheidungsfreiheit bezüglich der Systemgestaltung abnimmt [Dietz et al. 1998]. Der Schnittpunkt der entsprechenden Kurven wird in der Phase des konzeptionellen Entwurfs erreicht. Das Ziel, die Informationsverfügbarkeit in frühen Phasen der Produktentwicklung zu erhöhen, kann demnach durch Maßnahmen wie Simulationen und Berechnungen oder durch die Nutzung von organisationalem Wissen sowie die Generierung von Produktwissen durch die Gewinnung von Felddaten verbessert werden.

Organisationales bzw. institutionelles Wissen ist u.a. in Prozessen von Organisationen vorhanden [Mescheder/Sallach 2012, S. 15f] und kann dadurch nutzbar gemacht werden. Beispiele hierfür sind unternehmensspezifische Leitfäden oder Datenbanken. Ansätze zur Nutzung individuellen und institutionellen Wissens setzen voraus, dass Erfahrungen aus der Entwicklung vergleichbarer Vorgängersysteme bestehen. Besteht kein produktspezifisches Vorwissen, bieten Fähigkeits- und Reifegradmodelle einen Ansatz zur Nutzung von abstrahiertem Wissen zu Produktentwicklungsprozessen, siehe Kap. 3.1.1.

Ein weiteres, etabliertes Verfahren zur Verbesserung der Informationsverfügbarkeit stellen Ansätze der Felddatenrückführung dar. Felddaten sind Daten, „die im Zusammenhang mit der Nutzung eines Produktes im Feld“ [Edler 2001, S. 5] generiert werden. Ihre Rückführung in die Produktentwicklung stellt einen rückwärtsgerichteten Informationsfluss dar, welcher eine Verbesserung des PEP ermöglicht [ebd., S. 2f] und einen Beitrag dazu leistet, Systeme besser an die spezifischen Einsatzbedingungen anzupassen [Hentzschel/Jung 2002, S. 143]. Dabei setzen bestehende Ansätze hinsichtlich des Umgangs mit Informationen aus Felddaten verschiedene Schwerpunkte, wobei rückgeführte Felddaten oftmals für verbesserte Zuverlässigkeitsanalysen oder eine optimierte Systemauslegung genutzt werden. Die modellbasierte Felddatenrückführung in die Produktentwicklung, welche unter Berücksichtigung des Produktumfelds nicht nur das technische, sondern auch das soziotechnische Produktsystem umfasst, zielt speziell auf die Anwendung für komplexe Systeme ab und bietet ein fünfschrittiges Vorgehenskonzept zur Behebung von Qualitätsproblemen [Mamrot 2014]. Das Zuverlässigkeitsmanagement auf Basis von Felddaten, welches insbesondere für unvollständige und ungenaue Daten ein Analysekonzept bietet, stellt einen Ansatz zur Verbesserung der Informationsverfügbarkeit durch die Nutzung synthetischer Stichproben und Monte-Carlo-Simulationen dar [Delonga 2007]. Die ganzheitliche Datenerfassung für verbesserte Zuverlässigkeitsanalysen, über die ein modularer Ansatz zur Identifizierung von Zuverlässigkeitsdatenquellen verfolgt wird, bewertet die Leistungsfähigkeit von Zuverlässigkeitsdatenquellen über einen Quotienten, der zwischen Nutzen- und Aufwandskriterien bei der Datenerfassung unterscheidet und abwägt [Leopold 2012].

Auch innerhalb der Zuverlässigkeitstechnik wurden Verfahren entwickelt, um trotz einer geringen Informationsverfügbarkeit Aussagen über das System zu ermöglichen. So bieten beispielsweise die domänenübergreifende Zuverlässigkeitsbewertung in frühen Entwicklungsphasen [Wedel et al. 2007] oder die „situationsbasierte qualitative Modellbildung und Analyse“ [Bertsche et al. 2009] Möglichkeiten der Systemanalyse in frühen Entwicklungsphasen.

Alle Ansätze benötigen ihrerseits Informationen, welche in der Organisation vorhanden sein müssen bzw. setzen die Felddatenrückführung für eine Systemoptimierung voraus. Insbesondere in den frühen Phasen einer Produktneuentwicklung sind die hierfür erforderlichen Daten jedoch nicht oder nur bedingt verfügbar bzw. eine Felddatenrückführung nicht umsetzbar. Benötigte Informationen über das System zur Durchführung der Bewertung des Reifegrads stehen somit nicht unmittelbar zur Verfügung und müssen in Ergänzung der erwähnten Methoden entwicklungsbegleitend generiert werden.

Das Ziel für frühe Phasen der Produktentwicklung besteht somit darin, entwicklungsrelevante Informationen, die einen Einfluss auf die Anforderungserfüllung oder die Zuverlässigkeit und damit letztlich auf den Reifegrad des Systems besitzen, zu identifizieren und sie der Produktentwicklung zur Verfügung zu stellen. Ein Vorgehensmodell, welches die Lücke zwischen

prozessseitigen Reifegradmodellen und produktseitigen Methoden der Reifegradabsicherung schließt, muss demnach explizit Schnittstellen identifizieren und analysieren, um ihre Bedeutung bezogen auf in Relation stehende Systemelemente zu bewerten. Der in diesem Kontext verwendete Begriff der Informationsverfügbarkeit bezieht sich daher auf Informationen über das technische System sowie Wechselwirkungen innerhalb seiner Teilsysteme. Prozessinformationen, insbesondere von Herstellungsprozessen, sind nicht Bestandteil dieser Betrachtung, auch wenn die Herstellung technischer Produkte deren Eigenschaften beeinflusst. Gemäß der in Kap. 2.1 vollzogenen Eingrenzung sind diese fertigungs- und montagetechnischen Lösungsmöglichkeiten nicht Bestandteil der frühen Phasen der Produktentwicklung.

2.3 Komplexität

Eine wesentliche Einflussgröße der Produktentwicklung stellt die Komplexität technischer Systeme dar. Der Begriff der Komplexität wird von verschiedenen Disziplinen genutzt und oftmals in Abhängigkeit des Forschungsziels unterschiedlich definiert [Schoeneberg 2014, S. 14].

Eine grundsätzliche Differenzierung für die Produktentwicklung bietet MAURER, nach der sich die Komplexität auf die Felder Markt, Produkt, Prozess und Organisation erstreckt, die zueinander in Wechselbeziehung stehen [Maurer 2007, S. 2f]. Gemäß dem Fokus der vorliegenden Arbeit wird die Produktkomplexität betrachtet, d.h. die Komplexität technischer Systeme und nicht jene, die beispielsweise aus der Interaktion mit technischen Systemen, ihren Herstellungsprozessen oder ihrem Vertrieb einhergeht. Allerdings existieren auch innerhalb des Produktfelds unterschiedliche Auffassungen darüber, was Komplexität ist und über welche Merkmale sich der Grad von Komplexität bestimmen lässt. So wird in der Informationstechnik Komplexität unter Nutzung des Programmieraufwands in „Lines of Code“ angegeben und steht synonym für die Komplexität mechatronischer Systeme. Dabei steigt der Programmieraufwand seit Ende der 1970er Jahre mit ca. 10^2 Lines of Code bis heute ($>10^9$ Lines of Code) durch die Entwicklung softwarelastiger Funktionen im Kraftfahrzeug stetig an [Ständer 2011, S. 5]. Unter Zugrundelegung dieser Größe als Maßeinheit für Komplexität lässt sich folgern, dass Komplexität durch eine stärkere Vernetzung zwischen ehemals autarken Teilsystemen stetig zunimmt.

Die Definition über ausschließlich logische Systembestandteile erweitert NOLLAU. In Abhängigkeit des Aufbaus technischer Systeme benennt er das Vorhandensein von Energiespeichern, die der Elektrotechnik, technischen Mechanik, Fluidtechnik und Thermodynamik zugerechnet werden können, als Kennzeichen steigender Komplexität [Nollau 2009, S. 17]. Diese Definition fokussiert vor allem den Aufwand bei der Bildung mathematischer Modelle zur Beschreibung eines dynamischen Systemverhaltens [ebd., S. 14f]. Die Komplexität eines technischen Systems wird demnach maßgeblich durch sein dynamisches, nicht-lineares Verhalten bestimmt. KREIMEYER und LINDEMANN bezeichnen daher auch in Anlehnung an LEE die schwierige Einschätzung von Folgen, die Eingriffe in Systeme haben, als Merkmal von Komplexität [Kreimeyer/Lindemann 2011, S. 41, Lee 2003]. Diese Auffassung zur Definition von Komplexität präzisiert KROLL, indem er Komplexität in einen hierarchischen Zusammenhang zum Aufbau und Umgang mit Systemen setzt. Demnach bildet die untere Ebene ein kompliziertes System, welches eine „vergleichsweise einfache Struktur hat und sein Verhalten prinzipiell gut berechenbar ist“ [Kroll 2013, S. 4]. Das komplexe System besitzt eine „Vielzahl an Komponenten

und Verbindungen, Wechselwirkungen oder Abhängigkeiten“, welche „schwierig zu beschreiben, verstehen, steuern, entwerfen, ändern und/oder vorherzusagen“ [ebd., S. 4] sind.

Eine Besonderheit stellen chaotische Systeme dar. Diese Systeme müssen vom Aufbau bzw. ihrer Struktur weder kompliziert noch komplex sein, allerdings ist ihr Verhalten höchst komplex bzw. chaotisch, da „geringe Unterschiede in den Anfangsbedingungen zu völlig unterschiedlichen Trajektorien der Systemvariablen führen“ [ebd., S. 4]. Diese Erkenntnis geht auf Henri Poincaré zurück, der die Zufälligkeit von Ereignissen bei geringen Änderungen von Eingangsbedingungen beschrieb[4]. Beispiele für chaotisches, nichtlineares Systemverhalten beschreiben oftmals Pendel oder Massenschwinger [vgl. Magnus et al. 2008, S. 254ff]. Da chaotische Systeme bzw. chaotisches Systemverhalten einen für den Fokus dieser Arbeit nicht relevanten Sonderfall darstellen, werden sie nicht weiter betrachtet.

Die Problematik uneinheitlicher Definitionen und Abgrenzungen dessen, was Komplexität ist und wie sie sich messen lässt, fasst LEE zusammen: „It is not surprising to see that there is no consensus on the definition of complexity so far“ [Lee 2003, S. 20]. Für die vorliegende Arbeit ist daher die Festlegung einer Definition notwendig. Eine erste Eingrenzung für das Produktfeld innerhalb der Produktentwicklung stellt MAURER bereit. Die für diese Arbeit relevante strukturelle Komplexität stellt demnach ein messbares Systemmerkmal dar, welches sich aus der Quantität von Systembestandteilen und ihren Verbindungen ergibt [Maurer 2007, S. 32]. Obgleich das Verständnis von Komplexität als sich aus Systembestandteilen und deren Wechselbeziehungen ergebendes Systemmerkmal geteilt wird, bestehen zwischen den gezeigten Definitionen Divergenzen hinsichtlich der Quantifizierbarkeit und ihrem Nutzen. Im Verständnis dieser Arbeit besitzt Komplexität keinen festzulegenden Zielwert, unter dem beispielsweise von geringer Komplexität ohne negative Auswirkung auf das System auszugehen wäre. Demnach ist die erfasste Komplexität einer Systemstruktur ein wertungsfreier Fakt. Strukturelle Komplexität wird daher als Systemmerkmal verstanden, welches durch geeignete Maßnahmen beherrscht werden muss, ohne es zwangsläufig zu reduzieren. Demzufolge lautet die dieser Arbeit zugrundeliegende Definition in Anlehnung an die benannten Quellen wie folgt:

> *Die strukturelle Komplexität technischer Systeme ist ein inhärentes, dimensionsloses Merkmal, welches aus der Anzahl und Art von physischen und logischen Elementen und ihrer technisch-physikalischen oder logisch-strukturellen Beziehungen zueinander resultiert.*

Unter Zugrundelegung dieser Definition könnte irrtümlich der Eindruck entstehen, dass sich die Komplexität mit der Anzahl *beschriebener* Beziehungen zwischen Systemelementen erhöht. Während die Anzahl von Elementen relativ einfach bestimmbar ist, ist die Bestimmung der Anzahl von Beziehungen deutlich aufwändiger. Eine nicht bekannte oder nicht beschriebene Beziehung ist trotzdem existent, auch wenn sie für die Bestimmung der Komplexität nicht herangezogen wird. Ob sie beschrieben wird oder nicht, ändert demnach nicht ihre Existenz oder die vorhandene Komplexität. Dementsprechend ist es erforderlich, Beziehungsarten zu definieren, um eine vollständige und korrekte Erfassung zu ermöglichen.

[4] „Es kann vorkommen, dass kleine Abweichungen in den Anfangsbedingungen schließlich große Unterschiede in den Phänomenen erzeugen. Ein kleiner Fehler zu Anfang wird später einen großen Fehler zur Folge haben. Vorhersagen werden unmöglich, und wir haben ein zufälliges Ereignis.“ [Henri Poincaré (1903), nach Meschede 2006, S. 1085]

2.4 Zuverlässigkeit

Die Zuverlässigkeit technischer Systeme wird definiert als Zusammenfassung von Funktionsfähigkeit bzw. Verfügbarkeit über einen definierten Zeitraum [DIN 61025, S. 9, Rakowsky/Richardson 2001, S. 222f]. Dabei handelt es sich bei der Funktionsfähigkeit bzw. der Verfügbarkeit um eine Eigenschaft eines technischen Systems, welche durch einen fehlerfreien Systemzustand charakterisiert ist, dem sogenannten *up state*. Dieser liegt vor, wenn sich ein System „in einem Zustand [befindet], in dem es fähig (engl. able) ist, sich wie gefordert zu verhalten" [Müller 2013, S. 20]. Die IEC 60050-191 definiert diesen *up state* als „state of being able to perform as required" [ebd., S. 19, nach IEC 60050-191]. Der *up state* bezeichnet demnach ein einsatzbereites System.

Bedingt durch ein Fehlereignis (*failure*, „loss of ability to perform as required" [ebd., S. 19, nach IEC 60050-191]), welches in einem Fehlzustand (*fault state*) mündet, oder durch Maßnahmen der vorbeugenden Instandhaltung, d.h. der bewussten Außerbetriebnahme, welche im *preventive maintenance state* mündet, gelangt ein System in einen *down state*, in dem es „nicht fähig [ist], sich wie gefordert zu verhalten" [ebd., S. 20]. Abb. 8 veranschaulicht diese Zusammenhänge und verbindet sie über eine ODER-Verknüpfung.

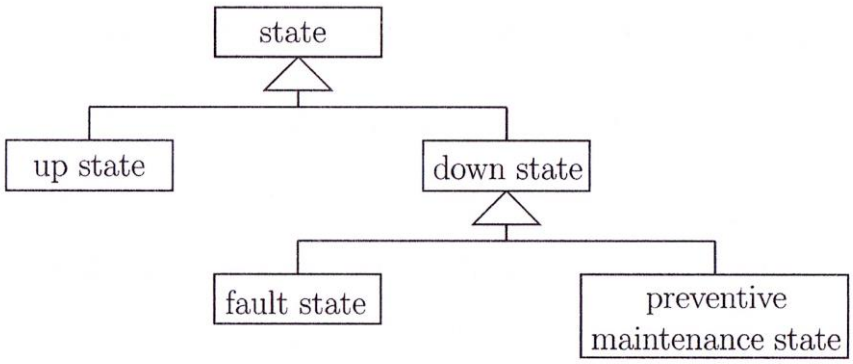


Abb. 8: Hierarchische Beziehung der Zustände [Müller 2013, S. 20]

Das Ziel der Zuverlässigkeit besteht nun darin, die beschriebene Funktionsfähigkeit bzw. die Verfügbarkeit des Systems über einen definierten Zeitraum, der *required time*, sicherzustellen, welche als „time intervall for which the item is required to be in an up state" ([ebd., S. 22, nach IEC 60050-191]) definiert ist. Der Hintergrund dieser Forderung ist darin begründet, dass Zuverlässigkeit ein wesentliches Merkmal für die Bewertung der Qualität eines Produkts aus der Sicht des Kunden ist [Bertsche et al. 2009, S. 4]. So zeigt die Analyse der DAT-Reports zwischen den Jahren 2000 und 2021, dass Anforderungen an die Zuverlässigkeit das Hauptkriterium beim privaten Neuwagenkauf sind [DAT 2000 – DAT 2021]. Diese Aussage trifft ebenso auf den industriellen Bereich zu, bei dem in einer Industriebefragung im Rahmen des Projekts Q-ELF nachgewiesen werden konnte, dass auch bei kostenintensiven Investitionsgütern wie intralogistischen Anlagen die Zuverlässigkeit an erster Stelle steht [Riekhof et al. 2014].

Zuverlässigkeit dient dabei als Oberbegriff determinierbarer Zuverlässigkeitskenngrößen wie beispielsweise der Verfügbarkeit *A*, der Überlebens- und Ausfallwahrscheinlichkeit *R(t)* bzw. *F(t)* oder der mittleren Lebensdauer *E(t)* [Meyna/Pauli 2010]. Die Zuverlässigkeit technischer Systeme lässt sich über zuverlässigkeitstechnische Kenngrößen beschreiben, wobei im Kern die Frage der Funktionsfähigkeit steht: Wie lange können geforderte Funktionen ausfallfrei

umgesetzt werden, welche Größen beeinflussen die Zuverlässigkeit und wie kann eine hohe Zuverlässigkeit bei vertretbarem Entwicklungs- und Kostenaufwand erreicht werden? Zur Beantwortung dieser Fragen kommen Methoden der Zuverlässigkeitstechnik zum Einsatz, deren Ziel die Bestimmung zuverlässigkeitsrelevanter Kenngrößen ist.

Bei der Durchführung von Zuverlässigkeitsanalysen wird zwischen qualitativen und quantitativen Analyseverfahren unterschieden, wobei quantitative Methoden auf die Berechnung konkreter Kenngrößen abzielen, während qualitative Methoden eine systematische Untersuchung eines Systems bezüglich der Auswirkungen und Ursachen von Fehlern und Fehlzuständen ermöglichen [Bertsche et al. 2009, S. 8]. Typische quantitative Analyseverfahren sind die Weibull-Analyse zur Bestimmung von Lebensdauerverteilungen in Abhängigkeit der Ausfallraten untersuchter Komponenten [DIN 61649], das Markoff-Verfahren zur Analyse von Zustandsübergangsdiagrammen, mittels derer Wahrscheinlichkeiten des Eintritts verschiedener sicherer und unsicherer Systemzustände berechnet werden können [DIN 61165] oder Zuverlässigkeitsblockdiagramme, welche die Struktur technischer Systeme mittels der booleschen Algebra abbilden [DIN 61078]. Im Gegensatz hierzu sind qualitative Methoden primär auf die Analyse von Fehlerursachen und -wirkungen bzw. deren proaktive Vermeidung ausgelegt, wobei Kennzahlen eher dem Vergleich zwischen verschiedenen Systemen und einer Klassifizierung dienen, weshalb diese Verfahren mitunter auch als semi-quantitativ bezeichnet werden. Ein etabliertes Verfahren stellt die Fehlzustandsart- und -auswirkungsanalyse (engl. Failure Mode and Effects Analysis, FMEA) dar, welche Fehlzustände technischer Systeme identifiziert sowie ihre Bedeutung, Auftretenswahrscheinlichkeit und Entdeckungsmöglichkeit bewertet und als Produkt zu einer dimensionslosen Risikoprioritätszahl (RPZ) zusammenfasst [DIN 60812]. Zudem werden mögliche Ursachen und Auswirkungen des Fehlzustands bestimmt. Weitere Verfahren sind die Fehlzustandsbaumanalyse (FBA; auch engl. Fault Tree Analysis, FTA), bei der Basisereignisse, die zu einem Ausfall als Endereignis führen können, innerhalb einer Systemstruktur aufgezeigt und über Gatter in Beziehung ihrer Ausgangs- und Eingangsereignisse gesetzt werden [DIN 61025]. Als semi-quantitative Methode ermöglicht die FTA zusätzlich die Bestimmung von Zuverlässigkeitskenngrößen des Systems sowie die Berechnung von Kenngrößen einzelner Komponenten wie Importanzen. Eine weitere qualitative und weit verbreitete Methode ist das Ishikawa-Diagramm, welches Einflussgrößen auf die Qualität und Zuverlässigkeit eines Systems als Ursache-Wirkungs-Diagramm darstellt.

Unabhängig davon, ob quantitative oder qualitative Zuverlässigkeitsanalysen durchgeführt werden, stellt ihre Anwendung per se keine Verbesserung der Zuverlässigkeit oder der Qualität technischer Systeme dar. Stattdessen dienen sie der Analyse, der Bestimmung von Kenngrößen, zeigen Ursache-Wirkungs-Beziehungen auf oder bilden die Entscheidungsgrundlage, ob korrektive Maßnahmen erforderlich sind. Sie liefern der Produktentwicklung damit wichtige Informationen, ob technische Systeme in der Lage sind, die an sie gestellten Anforderungen aus Sicht der Zuverlässigkeit zu erfüllen. Um anwendbar zu sein, benötigen die Methoden der Zuverlässigkeitstechnik Informationen über das System. Diese beziehen sich auf konkrete Kenngrößen wie Ausfallraten oder Ausfalldichte, Wechselbeziehungen zwischen Systemelementen, die Systemstruktur oder das Verhalten im Feld in Interaktion mit anderen technischen

Systemen[5] und der Systemumwelt. Je nachdem, welchen Fokus Zuverlässigkeitsanalysen haben, benötigen und generieren sie unterschiedliche zuverlässigkeitsrelevante Informationen. Beispielsweise wird zur Berechnung der sogenannten Mean Time Between Failure (MTBF), d.h. der mittleren Lebensdauer *T* eines Systems, die Fehlerrate λ der jeweiligen Systemkomponenten benötigt. Für elektronische Bauteile wird diese Fehlerrate in über die FIT-Rate[6] beschrieben [Eberlin/Hock 2014, S. 30ff]. Für mechanisch beanspruchte Systeme werden zur Ermittlung von Zuverlässigkeitskenngrößen Muster in der Entwicklungsphase gerafften Tests, sogenannten End-of-Life (EoL) Tests ausgesetzt, bei denen bestimmte Parameter über eine einen definierten Zeitraum auf das System wirken, um die Überlebenswahrscheinlichkeit *R(t)* zu bestimmen. In der Praxis stößt die Aussagekraft geraffter EoL-Tests jedoch an ihre Grenzen, da die Stichprobengröße in EoL-Tests oftmals gering ist und ihnen konstante Belastungen zugrunde liegen, welche nicht mit der realen Beanspruchung des Systems im Feld übereinstimmen bzw. Unsicherheiten aufweisen. Obwohl für beschleunigte Zuverlässigkeitstests Ansätze bestehen, welche die „Prognose der operativen Zuverlässigkeit bei zeitabhängigen Belastungen mit einem Konfidenzniveau“ [Kremer/Bertsche 2018, S. 395] ermöglichen, bleibt die grundsätzliche Problemstellung bestehen, dass diese Ansätze nicht oder nur über die Rückführung von Daten aus Vorgängerprodukten für die frühen Phasen der Produktentwicklung nutzbar sind.

Abstrakt betrachtet stellen Informationen, die sich auf die Relationen zwischen Elementen beziehen, strukturelle Merkmale des Systems in den Vordergrund. Diese können sich auf Relationen zwischen einzelnen Elementen oder des gesamten Systems beziehen und lassen sich beispielsweise für Zuverlässigkeitsanalysen nutzen, die den strukturellen Aufbau von Systemen zur Ableitung von Aussagen über das System nutzen. Wird das Gesamtsystem betrachtet, sind insbesondere Ursache-Wirkungs-Zusammenhänge relevant. Liegt der Fokus auf der Interaktion des Systems mit seiner Umwelt, sind Informationen zum Systemverhalten oder Herstellungs- und Nutzungsprozessen notwendig. Hierfür haben sich insbesondere die Produkt- und Prozess-FMEA durchgesetzt, vgl. [VDA 4.3]. Diese Informationen ermöglichen eine gegenüber elementbezogenen Informationen zu Komponenten umfassendere Betrachtung des Systems, benötigen jedoch bessere Kenntnisse über Elemente und Relationen.

Unabhängig vom Ziel einer Zuverlässigkeitsanalyse ist diese prinzipiell auf das Vorhandensein zuverlässigkeitsrelevanter Informationen angewiesen. Entsprechend werden Informationen zu Elementen, Relationen, dem Gesamtsystem und/oder seinen Schnittstellen zur Systemumwelt benötigt. Im Umkehrschluss determiniert die Informationsverfügbarkeit die Durchführbarkeit von Zuverlässigkeitsanalysen. Die Informationsverfügbarkeit steht somit in unmittelbarer Wechselbeziehung zur Zuverlässigkeit, indem sie einerseits Informationen über das System benötigt und andererseits neue Informationen liefert. Für den Kontext der Arbeit muss daher bereits in den frühen Phasen der Produktentwicklung sichergestellt werden, dass gezielt solche Informationen generiert werden, welche die Nutzung weiterführender Zuverlässigkeitsanalysen sowohl auf Komponenten- als auch auf Systemebene ermöglichen.

---

[5] In Abgrenzung zu technischen Systemen, die aus in Wechselwirkung zueinander stehenden Elementen bestehen, werden unter soziotechnischen Systemen solche verstanden, welche „aus den Wechselbeziehungen zwischen Menschen und Maschinen oder Anlagen [bestehen]“ [Winzer 2016, S. 82]. Letztere sind kein Betrachtungsgegenstand der vorliegenden Arbeit.

[6] Failure in Time; 1 FIT = $1x10^{-9}$ 1/h [SN 29500-2, S. 6]

3 Stand der Wissenschaft und Technik

Wie im vorherigen Kapitel aufgezeigt, wird die Produktentwicklung insbesondere in ihren frühen Phasen durch Informationsverfügbarkeit, Komplexität und Zuverlässigkeit beeinflusst. Die Beherrschung dieser sich gegenseitig beeinflussenden Aspekte ist Kernbestandteil dieser Arbeit. Wie bereits dargelegt, existieren Ansätze zur Beherrschung dieser Herausforderungen, welche sich insbesondere den Themenbereichen Fähigkeits- und Reifegradmodelle, Komplexitätsmanagement, Zuverlässigkeitsmanagement und Systemmodellierung zuordnen lassen. Für diese Bereiche wird nachfolgend der Stand der Wissenschaft und Forschung erläutert sowie Anforderungen an den eigenen Ansatz abgeleitet.

3.1 Fähigkeits- und Reifegradmodelle und Methoden der Reifegradabsicherung

Die Messung von Prozessergebnissen und die kontinuierliche Verbesserung stellen zentrale Forderungen des Qualitätsmanagements dar. Im Speziellen fordert die DIN EN ISO 9001:2000 für die Produktentwicklung eine „systematische Entwicklungsbewertung“, bei der „die Fähigkeit der Entwicklungsergebnisse zur Erfüllung der Anforderungen“ nachzuweisen ist [DIN EN ISO 9001:2000, S. 26]. Die Erfüllung von gesetzlichen Anforderungen und von Kundenanforderungen an das Produkt und den Produktionsprozess stellt damit ein wesentliches Ziel der Produktentwicklung dar. In der Automobilzuliefererindustrie, welche als exemplarische Branche für die Entwicklung und Fertigung komplexer mechatronischer Systeme genannt werden kann, existieren eine Vielzahl gesetzlicher und branchenabhängiger Reglementierungen. Gleichzeitig müssen hohe Qualitätsansprüche der OEM[7] und der Endkunden erfüllt werden. Die Nachweisführung der Anforderungserfüllung erfolgt hierfür über das Produktionsprozess- und Produktfreigabe-Verfahren (PPF-Verfahren), bei dem mit Serienbetriebsmitteln produzierte Erstmuster produktspezifischen Tests unterzogen werden. Wird dieser Bemusterungsprozess durch den Nachweis der prozess- und produktseitigen Anforderungserfüllung erfolgreich abgeschlossen, erteilt der Kunde, d.h. der OEM oder der Tier 1[8], in der Regel die Freigabe zur Serienlieferung [VDA Band 2, S. 9ff, 18ff]. Mit dem SOP erfolgt daraufhin die Übergabe des Entwicklungsprojekts an die Serienproduktion [VDA 2011], wobei die eigentliche Entwicklung bereits zu einem früheren Zeitpunkt abgeschlossen ist und der Serienanlauf beginnt. Die in der Nullserie produzierten Erstmuster bilden dann die Grundlage der Nachweisführung der Anforderungserfüllung im PPF-Verfahren. Die Phasen des Serienanlaufs mit ihren zentralen Meilensteinen zeigt Abb. 9. Wird zusätzlich die Lieferkette betrachtet, verschieben sich die Phasen und Meilensteine je untergeordneter Lieferebene auf der Zeitachse nach links, sodass dem OEM bei SOP über alle Lieferebenen freigegebene und qualifizierte Produkte zur Verfügung stehen.

[7] Original Equipment Manufacturer
[8] Tier (dt. Ebene) bezeichnet die hierarchische Folge innerhalb der Lieferkette in der Automobilindustrie

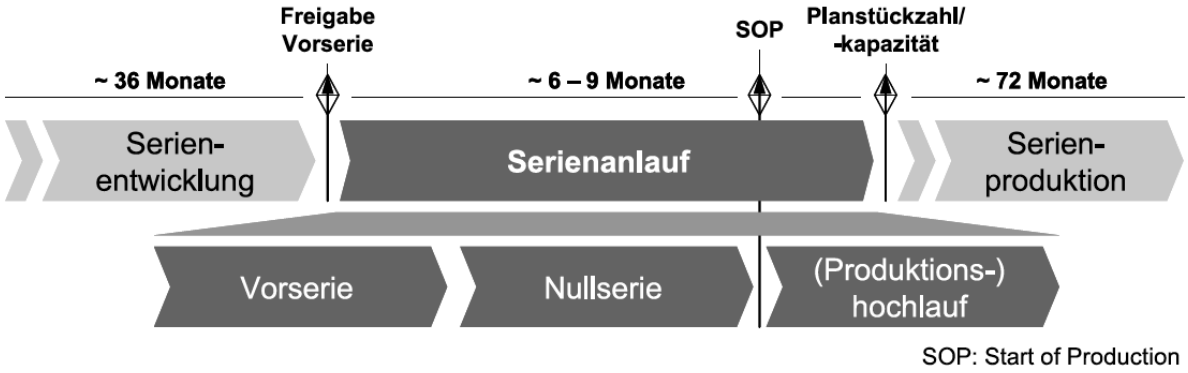


Abb. 9: Phasen des Serienanlaufs [Schuh et al. 2008, S. 2]

Bedingt durch den Umstand, dass nach Schätzungen 75 % aller Fehler in der Produktentwicklung verursacht werden [Westkämper 2006] und durch die resultierenden Kosten der Fehlerbehebung in nachgelagerten Prozessphasen [Pfeifer 2001, von Regius 2006] sind Hersteller bestrebt, Fehler zu vermeiden bzw. frühzeitig zu entdecken. Für die Phase der Produktentwicklung, in der noch keine zur Nachweisführung nutzbaren Muster aus Vor- oder Nullserie für die Design Verification (DV) oder Product Validation (PV) zur Verfügung stehen, sind geeignete Strategien erforderlich, um einerseits Voraussetzungen für eine erfolgreiche Produktentwicklung zu schaffen und andererseits entwicklungsbegleitend den geforderten Soll-Zustand des Produkts mit dem Ist-Zustand zu vergleichen. Diesbezügliche Ansätze lassen sich über ihren Bezug zum Produkt bzw. zum Produktentwicklungsprozess unterscheiden und werden über ihren generischen Prozess- bzw. konkreten Produktbezug differenziert.

Ein generischer Prozessbezug liegt vor, wenn nicht das eigentliche Produkt, sondern der Prozess der Produktentwicklung im Fokus steht. Entwicklungsprozesse stellen eine sich prinzipiell wiederholende Schrittfolge dar, obgleich die Produktentwicklung Projektcharakter besitzt und typische Projektmerkmale aufweist. Zu diesen gehören eine dem Zweck angemessene Organisation und Planung, definierte Ziele, ein definiertes Projektende, einen Abgleich von Ziel und Vorgaben [Burghardt 2013, S. 9] sowie Einmaligkeit im Sinne der Projektkonstellation, Interdisziplinarität, arbeitsteilige Projektbearbeitung und eine fortschreitende Konkretisierung [Meyer/Reher 2016, S. 2]. Bei der Produktentwicklung können somit auch trotz projektspezifischer Anpassungen, dem sogenannten Tailoring, generische Prozessschritte und Abläufe identifiziert werden, deren Standardisierung und Verbesserung zu wiederholbaren guten Ergebnissen führt. Ansätze mit generischem Prozessbezug, die konkrete Vorgaben über erwartete Ergebnisse machen, werden als Fähigkeits- und Reifegradmodelle bezeichnet [Paulk 2009].

Bezogen auf den konkreten Produktbezug erfolgt die Nachweisführung der Anforderungserfüllung final über das PPF-Verfahren bzw. vergleichbare Verfahren. In der sich im Sinne des V-Modells konkretisierenden Produktentwicklung existieren jedoch bereits vor Erreichen der Freigabe für die Vorserie Entwicklungsstände auf domänenspezifischer Komponentenebene sowie durch sukzessive Systemintegration auf domänenübergreifender Subsystem- und Systemebene. Deren Eigenschaftsabsicherung ermöglicht frühzeitig eine Aussage zur Anforderungserfüllung [Bender 2005, VDI 2206] und stellt einen konkreten Produktbezug dar, bei der an definierten Meilensteinen (Quality Gates) die Anforderungserfüllung des Produkts gemessen wird [Feldhusen/Grote 2021, S. 885]. Dies geschieht oftmals über Checklisten oder Workshops, welche den Entwicklungsfortschritt anhand von meilensteinspezifischen Vorgaben

messen [Winzer 2016, S. 264f] und eine Aussage zum Reifegrad des Produkts ermöglichen [VDA 2009]. Die Ermittlung der Produktreife an Quality Gates dient somit der Bewertung des Entwicklungsfortschritts zum Ende einer Projektphase und der Entscheidung über die Fortsetzung des Projekts vor Beginn einer neuen Projektphase [Dourado et al. 2013, S. 85]. Dabei determiniert das für die Produktentwicklung genutzte Vorgehensmodell die Produktlebenszyklusphasen, an denen eine Bewertung erfolgt [Herrmann et al. 2013, S. 79f].

3.1.1 Fähigkeits- und Reifegradmodelle

Die Bewertung des Fähigkeitsgrads und des Reifegrads stellt einen generischen Ansatz zur Schaffung einer Grundlage für die Prozessverbesserung dar. Dies erfolgt üblicherweise anhand von Modellen, welche durchzuführende Prozesse und deren Ergebnisse auf verschiedenen, aufeinander aufbauenden Fähigkeits- und Reifegraden definieren. Für das Verständnis und den Aufbau dieser Modelle ist die Betrachtung der Entwicklung von Fähigkeits- und Reifegradmodellen zweckdienlich.

3.1.1.1 Entwicklung der Fähigkeits- und Reifegradmodelle

Historisch gesehen wurde das Reifegraddenken von CROSBY entwickelt. Basierend auf der Erkenntnis, dass Qualitätsprobleme primär der Produktion zugeschrieben wurden, obwohl das Management und die Produktentwicklung letztlich die Produktqualität stärker beeinflussen können als die Produktion selbst [Crosby 1979, S. 16ff], entwickelte CROSBY das „Quality Management Maturity Grid“ (QMMG). Demnach lassen sich Prozesse über fünf aufeinander aufbauende Stufen („Stages“) charakterisieren [Crosby 1979, S. 21ff]:

1. Uncertainty (Unsicherheit)
2. Awakening (Erwachen)
3. Enlightenment (Erkenntnis)
4. Wisdom (Verständnis)
5. Certainty (Sicherheit)

In der Stufe 1 (Uncertainty) reagiert das Management unkontrolliert und im Sinne einer „Qualitätspolizei“ auf Qualitätsprobleme, welche auf der fehlenden Erfüllung von Anforderungen beruhen („Conformance to requirement“ [Crosby 1979, S. 15]). Dabei wird die Ursache eher auf personeller anstatt auf systematischer Ebene gesucht, wodurch eine Qualitätsverbesserung verhindert wird. [Crosby 1979, S. 26f]

Die Stufe 2 (Awakening) beschreibt die Erkenntnis in einem Unternehmen, dass ein Qualitätsmanagement notwendig ist, um Probleme zu lösen. Die Problemlösung erfolgt jedoch primär nachgelagert durch erhöhte Prüf- und Überwachungsaufwände, langfristige Lösungen werden nur unzureichend berücksichtigt. Hinsichtlich der Qualitätskosten existiert ein mangelndes Verständnis für den Umfang, sodass oftmals zu geringe Kosten veranschlagt werden – CROSBY nennt als Verhältnis der betrachteten zu realen Qualitätskosten das Verhältnis 1:6. Insbesondere für den Fall, dass durch kurzfristige Erfolge die sogenannte „Management Attention“ entfällt, droht ein Rückfall auf Stufe 1. [Crosby 1979, S. 27f]

Das Erreichen der Stufe 3 (Enlightenment) geht einher mit der Definition einer Qualitätspolitik und der Einführung eines Qualitätsverbesserungsprogramms. Dabei wird der Qualitätsabteilung innerhalb des Unternehmens eine größere Bedeutung und Handlungskompetenz übertragen.

Bei der Lösung von Qualitätsproblemen stehen Fakten anstelle von Personen im Vordergrund. Dennoch bleiben weiterhin ein Drittel der Qualitätskosten unerkannt. [Crosby 1979, S. 28]

Unternehmen, welche die Stufe 4 (Wisdom) erreichen, haben geeignete Maßnahmen eingeführt, um die sogenannten Costs of Quality[9] zu reduzieren. Qualität wird als gemeinschaftliche Aufgabe verstanden und in die Verantwortung niedrigerer Managementebenen gegeben, um Änderungen permanent in der Unternehmenskultur zu verankern. Es resultiert jedoch die Gefahr, dass eine Lockerung der Maßnahmen zu einer Nachlässigkeit führen kann, was die Stufe 4 zur kritischsten Stufe des Quality Management Maturity Grids macht. [Crosby 1979, S. 29f]

Mit dem Erreichen der Stufe 5 (Certainty) haben Unternehmen ein umfassendes System zur Vorbeugung eingeführt, sodass Qualitätsprobleme selbst kaum noch auftreten. Die verursachten Qualitätskosten entsprechen im Wesentlichen den Kosten für die Qualitätsabteilung und den Kosten für Nachweisuntersuchungen. Das Qualitätsmanagement ist ein integraler Bestandteil des Unternehmensmanagements und der QM-Leiter Teil des Unternehmensvorstands. [Crosby 1979, S. 30f]

Die von CROSBY definierten Stufen des QMMG fasst AKKASOGLU in Anlehnung an BECKER ET AL. zusammen als „keine Kenntnis über Qualitätsprobleme“ (Stufe 1), „Infragestellung der Existenz von Qualitätsproblemen“ (Stufe 2), „Systematische Problemidentifikation und -lösung“ (Stufe 3), „Routinierte Fehlervermeidung“ (Stufe 4) und „Absolute Kenntnis über Qualitätsprobleme“ (Stufe 5) [Akkasoglu 2013, S. 10, Becker et al. 2009].

Zur Bewertung des Reifegrads einer Organisation entwickelte CROSBY die sechs Messkategorien [Crosby 1979, S. 31f]:

1. Management understanding and attitude
2. Quality organization status
3. Problem handling
4. Cost of quality as % of sales
5. Quality improvement actions
6. Summation of company quality posture

Diese Kategorien bilden mit den fünf Stufen zur Bewertung des Reifegrads ein Raster (engl. Grid) und enthalten je Kategorie und Reifegrad eine Aussage über die erreichte Umsetzung in der bewerteten Organisationseinheit. Anhand des QMMG kann eine Evaluation des Reifegrads erfolgen. Hierfür sind als Assessoren der Geschäftsführer (*General Manager*), der Qualitätsmanager (*Quality Manager*) sowie ein Mitarbeiter der bewerteten Abteilung (*Staff Member*) vorgesehen. In Abb. 10 ist das QMMG dargestellt, welches unter „Measurement Categories“ die sechs Messkategorien aufführt und gemeinsam mit den Stages 1 – 5 das Raster bildet. Das QMMG kann somit direkt für die Bewertung eines auditierten Bereichs genutzt werden, indem die Zielerreichung je Messkriterium angekreuzt wird und somit die erreichte Stufe ergibt.

[9] Der Begriff „Cost of Quality“ (dt. Qualitätskosten) bezeichnet nach CROSBY die Kosten, welche beispielsweise aus Nacharbeit, Verschrottung oder Gewährleitung resultieren [Crosby 1979, S. 178f]. Im heutigen Sprachgebrauch werden diese Kosten parallel auch als Nicht-Qualitätskosten beschrieben, engl. Non-quality expenses (NQE) bzw. Costs of poor quality (COPQ).

Quality Management Maturity Grid (Crosby)	Assessor:			Department:	
Measurement Categories	**Stage 1: *Uncertainty***	**Stage 2: *Awakening***	**Stage 3: *Enlightenment***	**Stage 4: *Wisdom***	**Stage 5: *Certainty***
Management understanding and attitude	No comprehension of quality as a management tool. Tend to blame quality department for "quality problems".	Recognising that quality management may be of value but not willing to provide money or time to make it all happen.	While going through quality improvement programme learn more about quality management; becoming supportive and helpful.	Participating. Understand absolutes of quality management. Recognise their personal role in continuing emphasis.	Consider quality management as an essential part of company system.
Quality organisation status	Quality is hidden in manufacturing or engineering departments. Inspection probably not part of organisation. Emphasis on appraisal and sorting.	A stronger quality leader is appointed but main emphasis is still on appraisal and moving the product. Still part of manufacturing or other.	Quality department reports to top management, all appraisal is incorporated and manager has role in management of company.	Quality manager is an officer of company; effective status reporting and preventive action. Involved with customer affairs and special assignments.	Quality manager on board of directors. Prevention is main concern. Quality is a thought leader.
Problem handling	Problems are fought as they occur; no resolution; inadequate definition; lots of yelling and accusations.	Teams are set up to attack major problems. Long-range solutions are not solicited.	Corrective action communication established. Problems are faced openly and resolved in an orderly way.	Problems are identified early in their development. All functions are open to suggestion and improvement.	Except in the most unusual cases, problems are prevented.
Cost of quality as % of sales	Reported: Unknown Actual: 20%	Reported: 3% Actual: 18%	Reported: 8% Actual: 12%	Reported: 6.5% Actual: 8%	Reported: 2.5% Actual: 2.5%
Quality improvement actions	No organised activities. No understanding of such activities	Trying obvious "motivational" short-range efforts.	Implementation of a multi-step programme (e.g. Crosby's 14-step) with thorough understanding and establishment of each step.	Continuing the multi-step programme and starting other pro-active / preventive product quality initiatives.	Quality improvement is a normal and continued activity.
Summary of company quality posture	"We don't know why we have problems with quality".	"Is it absolutely necessary to always have problems with quality?"	"Through management commitment and quality improvement we are identifying and resolving our problems."	"Defect prevention is a routine part of our operation."	"We know why we do not have problems with quality."

Abb. 10: Quality Management Maturity Grid [Cusick 2020, S. 4 nach Crosby 1979, S. 32/33]

Das Quality Management Maturity Grid dient damit als Vorläufer moderner Fähigkeits- und Reifegradmodelle. Als prägende Weiterentwicklung kann das vom Software Engineering Institute (SEI) im August 1991 in der Version 1.0 herausgegebene „Capability Maturity Model for Software“ (CMM) [Paulk 2009, S. 12] angesehen werden. Das CMM führt den Prozessansatz des QMMG fort und definiert mit der 1993 erschienenen Version 1.1 die fünf Stufen der Prozessreife *Initial*, *Repeatable*, *Defined*, *Managed* und *Optimizing* [Paulk et al. 1993, S. 7ff], welche trotz leichter Änderungen bis heute Bestand haben. Abb. 11 zeigt die fünf Stufen des CMM und benennt jeweils die wesentliche Verbesserung, welche für das Erreichen einer höheren Stufe erforderlich ist.

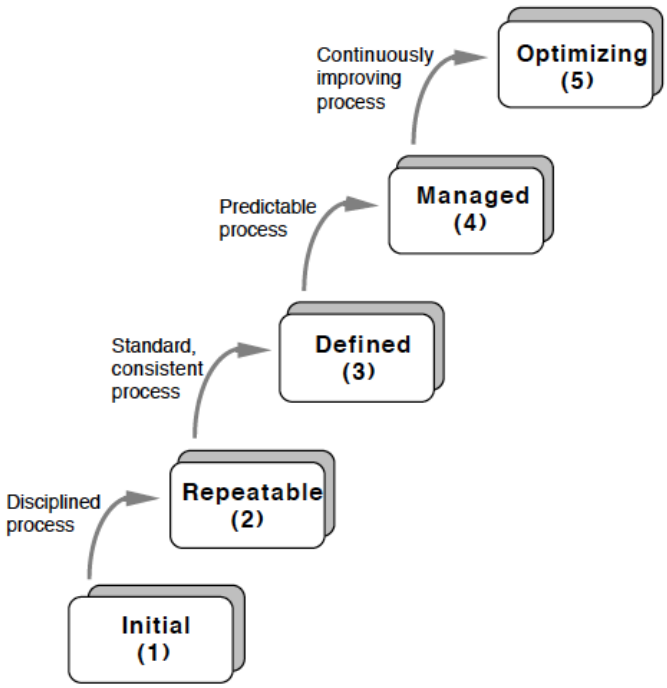


Abb. 11: The Five Levels of Software Process Maturity [Paulk et al. 1993, S. 8]

Aus den CMMs, aus denen neben der Softwareentwicklung auch Versionen mit dem Fokus auf dem Systems Engineering entstanden, resultierte das Capability Maturity Model Integration (CMMI). Dieses teilt sich in jeweils eigene Modelle für die Akquise (CMMI-ACQ), Dienstleistungen (CMMI-SVC) und die Entwicklung (CMMI-DEV) auf. Alle Modelle liegen in der Version 1.3 vor und wurden im Jahr 2010ff veröffentlicht [SEI 2011, S. 18].

Im März 2018 kündigte das CMMI Institute die Veröffentlichung des CMMI V2.0 an[10], welches die bisherigen Modelle CMMI-ACQ, CMMI-DEV und CMMI-SVC zusammenführen soll. Aktuell kann auf das CMMI V2.0 nur über den kostenpflichtigen CMMI Model Viewer zugegriffen werden[11]. Für die vorliegende Arbeit wird ausschließlich das frei zugängliche CMMI-DEV 1.3 genutzt.

Eine vergleichende Betrachtung zwischen dem CMMI-DEV und dem CMMI V2.0 sowie eine Einschätzung zur Validität der Nutzung des CMMI-DEV wird in Kap. 3.1.1.2.4 vorgestellt.

3.1.1.2 Capability Maturity Model Integration for Development

Das Capability Maturity Model Integration for Development (CMMI-DEV) stellt ein auf die Belange von Unternehmen der Produkt- und Dienstleistungsentwicklung ausgerichtetes Modell des CMMI dar. Das CMMI-DEV liegt in der Version 1.3 aus dem Jahr 2011 in deutscher Sprache vor und basiert, wie auch das CMMI-ACQ und das CMMI-SVC, auf der CMMI Model Foundation (CMF), welche die gemeinsame Grundlage bildet. Das CMF stellt die gemeinsame Struktur dar und umfasst 16 Kernprozessgebiete, die für alle Modelle identisch sind. Für die spezifischen Modelle wie das CMMI-DEV werden sogenannte Konstellationen gebildet,

[10] Siehe Pressemitteilung des CMMI Institute vom 28.03.2018: https://cmmiinstitute.com/news/press-releases/march-2018/annoucingv2, zuletzt aufgerufen am 05.11.2019.
[11] Siehe Beschreibung zum CMMI Model Viewer des CMMI Institute: https://cmmiinstitute.com/products/cmmi/cmmi-v2-products, zuletzt aufgerufen am 05.11.2019.

welche die für das jeweilige Modell erforderlichen Prozessgebiete umfassen sowie spezifische Modellkomponenten enthalten. Die Prozessgebiete der Konstellationen können dabei inhaltlich voneinander abweichen bzw. die Kernprozessgebiete um konstellationsspezifische Prozessgebiete ergänzen. [SEI 2011, S. 4, S. 19ff]

Für das CMMI-DEV existieren 22 Prozessgebiete, die der nachfolgenden Tab. 3 zu entnehmen sind und die sich in vier Kategorien (Entwicklung, Projektmanagement, Prozessmanagement, Unterstützung) einteilen lassen. Neben den 16 Kernprozessgebieten enthält das CMMI-DEV ein gemeinsames Prozessgebiet, welches in mehr als einem, aber nicht in allen Modellen vorkommt, sowie fünf entwicklungsspezifische Prozessgebiete, welche ausschließlich für das CMMI-DEV Gültigkeit besitzen. Ein Prozessgebiet beschreibt dabei „eine Gruppe verwandter Praktiken in einem Gebiet, die bei gemeinsamer Umsetzung einen Satz von wichtigen Zielen für Verbesserungen in diesem Gebiet erfüllen“ [SEI 2011, S. 23]. Der angegebene Reifegrad zeigt, für welche zu erreichende Stufe die Umsetzung des jeweiligen Satzes von Prozessgebieten erforderlich ist. Prozessgebiete mit niedrigem Reifegrad sind in ihrer Bedeutung fundamentaler und müssen auch für niedrigere Reifegradstufen umgesetzt werden, während Prozessgebiete mit einem hohen Reifegrad für fortschrittliche Unternehmen mit einer hohen Prozessreife erforderlich sind. Die Reihenfolge ergibt sich aus der alphabetischen Sortierung der englischen Abkürzungen der Prozessgebiete.

Tab. 3: Prozessgebiete des CMMI-DEV [vgl. SEI 2011, S. 23, S. 45f]

Bezeichnung Prozessgebiet	Bezeichnung englisch	Kürzel	Kategorie	Reife-grad
Ursachenanalyse und -beseitigung	Causal Analysis and Resolution	CAR	Unterstützung	5
Konfigurationsmanagement	Configuration Management	CM	Unterstützung	2
Entscheidungsfindung	Decision Analysis and Resolution	DAR	Unterstützung	3
Fortgeschrittenes Projektmanagement	Integrated Project Management	IPM	Projekt-management	3
Messung und Analyse	Measurement and Analysis	MA	Unterstützung	2
Organisationsweite Prozessentwicklung	Organizational Process Definition	OPD	Prozess-management	3
Organisationsweite Prozessausrichtung	Organizational Process Focus	OPF	Prozess-management	3
Organisationsweites Leistungsmanagement	Organizational Performance Management	OPM	Prozess-management	5
Organisationsweite Prozessleistung	Organizational Process Performance	OPP	Prozess-management	4
Organisationsweite Aus- und Weiterbildung	Organizational Training	OT	Prozess-management	3
Produktintegration	Product Integration	PI	Entwicklung	3
Projektverfolgung und -steuerung	Project Monitoring and Control	PMC	Projekt-management	2
Projektplanung	Project Planning	PP	Projekt-management	2
Prozess- und Produkt-Qualitätssicherung	Process and Product Quality Assurance	PPQA	Unterstützung	2
Quantitatives Projektmanagement	Quantitative Project Management	QPM	Projekt-management	4
Anforderungsentwicklung	Requirements Development	RD	Entwicklung	3
Anforderungsmanagement	Requirements Management	REQM	Projekt-management	2
Risikomanagement	Risk Management	RSKM	Projekt-management	3
Zulieferungsmanagement	Supplier Agreement Management	SAM	Projekt-management	2
Technische Umsetzung	Technical Solution	TS	Entwicklung	3
Validierung	Validation	VAL	Entwicklung	3
Verifizierung	Verification	VER	Entwicklung	3

Die beschriebenen 22 Prozessgebiete des CMMI-DEV besitzen jeweils eigene Modellkomponenten. Modellkomponenten im Sinne des CMF unterteilen sich in erforderliche, erwartete und informative Komponenten und charakterisieren das jeweilige Prozessgebiet. Dabei stellen die erforderlichen Komponenten diejenigen spezifischen und generischen Ziele dar, die für die Erreichung eines konkreten Reife- oder Fähigkeitsgrads zu erfüllen sind. Erwartete Komponenten leiten sich aus den erforderlichen Komponenten ab und beschreiben die Umsetzung der Ziele durch spezifische und generische Praktiken. Informative Komponenten dienen der Erläuterung und Veranschaulichung. [SEI 2011, S. 21f]

Der Zusammenhang der CMMI-Modellkomponenten wird in Abb. 12 veranschaulicht. Auf Ebene des Prozessgebiets wird der Zweck des Prozessgebiets beschrieben, zudem gibt er einführende Hinweise und die Nennung weiterer, in Beziehung stehender Prozessgebiete. Als informative Komponenten sind diese Modellkomponenten als ovale Felder dargestellt. Das Prozessgebiet unterteilt sich in spezifische und generische Ziele (erforderliche Modellkomponenten; dargestellt als Rechteck mit gerundeten Kanten), welche wiederum spezifische und generische Praktiken besitzen (erwartete Modellkomponenten; dargestellt als Raute). Auf der untersten Ebene enthalten die spezifischen und generischen Praktiken zusätzlich informative Komponenten, die typischen Arbeitsergebnisse und Subpraktiken bzw. Subpraktiken und die Ausarbeitung generischer Praktiken.

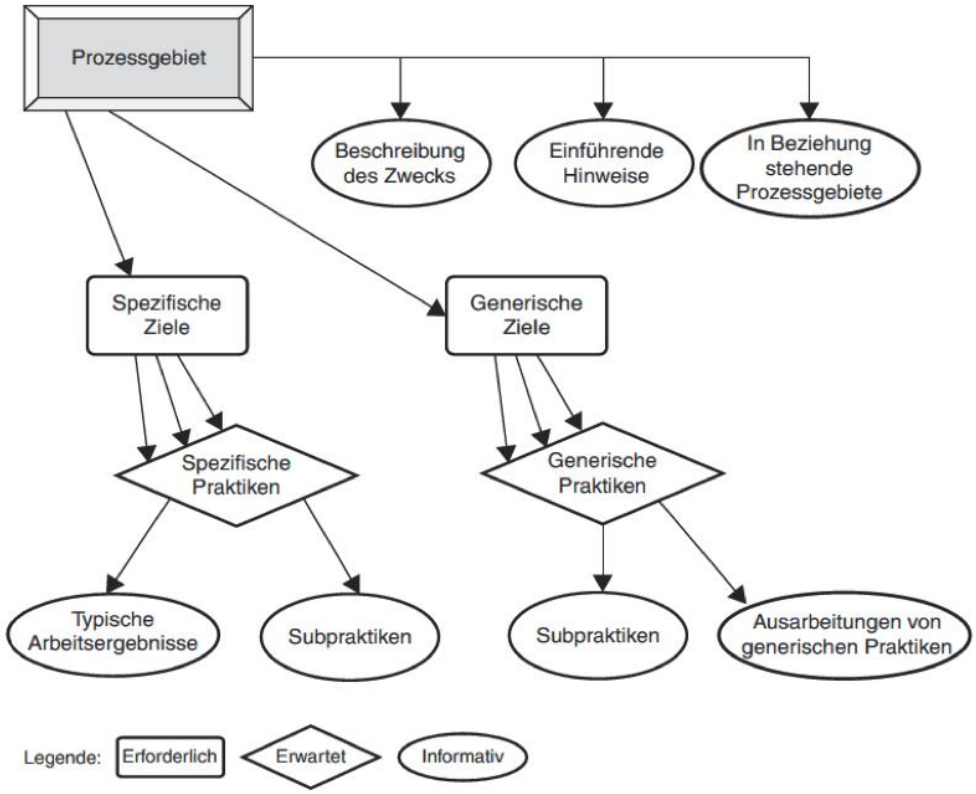


Abb. 12: CMMI-Modellkomponenten [SEI 2011, S. 22]

Mit der Differenzierung zwischen generischen Zielen und Praktiken sowie spezifischen Zielen und Praktiken wird der Aufbau der Modellkomponenten präzisiert. Damit besitzt ein Prozessgebiet sogenannte eindeutige Merkmale, d.h. spezifische Ziele, welche nur für dieses Prozessgebiet Gültigkeit besitzen und deren Erfüllung erforderlich für einen Fähigkeitsgrad ist. Um ein spezifisches Ziel zu erfüllen, müssen spezifische Praktiken umgesetzt werden, die als erwartete Modellkomponenten notwendig sind. Dies ermöglicht die Erreichung des Fähigkeitsgrads in einem konkreten Prozessgebiet.

Die generischen Ziele unterscheiden sich von den spezifischen Zielen in ihrem Bezug zu den Prozessgebieten. Während die spezifischen Ziele nur Merkmale für ein konkretes Prozessgebiet beschreiben, beziehen sich generische Ziele auf mehrere Prozessgebiete und legen den Fokus auf die Institutionalisierung der Prozesse. Die generischen Praktiken sind ebenso wie die spezifischen Praktiken erwartete Modellkomponenten, über welche die generischen Ziele erfüllt werden. [SEI 2011, S. 24ff]

Die erforderlichen und erwarteten Modellkomponenten (Ziele und Praktiken) müssen für eine Prozessverbesserung erreicht bzw. umgesetzt werden. Die Bewertung der Erreichung spezifischer und generischer Ziele sowie die Umsetzung der spezifischen und generischen Praktiken erfolgt über sogenannte Appraisals (Beurteilungen). Hierfür ist zunächst festzulegen, ob für das Appraisal die Darstellung in Reife- oder Fähigkeitsgraden gewünscht ist [SEI 2011, S. 46].

Bei der Darstellung in Fähigkeitsgraden wird die Erreichung der generischen Ziele auf ein einzelnes Prozessgebiet bewertet. Im Gegensatz hierzu bezieht sich das Appraisal bei der Darstellung in Reifegraden auf einen Satz von Prozessgebieten. Entsprechend müssen die erforderlichen Ziele und erwarteten Praktiken aller enthaltenen Prozessgebiete erfüllt und umgesetzt werden. Diese Unterscheidung der Darstellung in Fähigkeits- und Reifegraden zeigt Abb. 13.

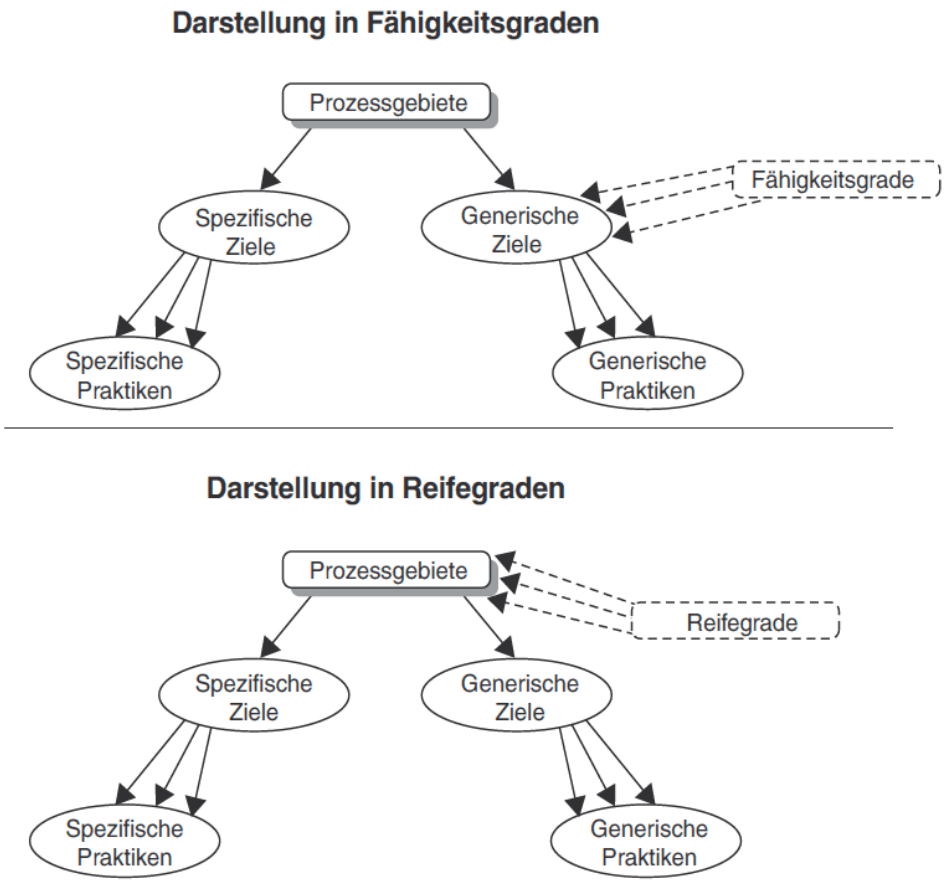


Abb. 13: Die Struktur der Darstellung in Fähigkeits- und Reifegraden [SEI 2011, S. 43]

Mit der Differenzierung zwischen Fähigkeitsgraden und Reifengraden ermöglicht das CMMI unterschiedliche Wege der Darstellung. Beide Optionen orientieren sich hinsichtlich ihrer Bewertung an dem gestuften Bewertungsmodell. Zur Erreichung eines Fähigkeitsgrads müssen die zugehörigen generischen Ziele des jeweiligen Prozessgebiets erfüllt sein. Im Gegensatz hierzu erfordert die Erreichung eines Reifegrads die Umsetzung der generischen sowie der spezifischen Praktiken aller zugehörigen Prozessgebiete [SEI 2011, S. 36ff]. Die Darstellung der Fähigkeits- und Reifegrade erfolgt über Grade, wobei die sich nur auf ein Prozessgebiet beziehenden Fähigkeitsgrade die Grade 0 (Unvollständig) bis 3 (Definiert) erreichen können, während sich die prozessgebietsübergreifenden Reifegrade in den Graden 1 (Initial) bis 5 (Prozessoptimierung) darstellen lassen. Die Fähigkeits- und Reifegrade 1 bis 3 sind auf gleicher

Ebene dargestellt, wobei der Fähigkeitsgrad 1 als durchgeführt und der Reifegrad 1 als initial definiert sind, vgl. Tab. 4.

Der Versatz erreichbarer Fähigkeits- und Reifegrade ist darin begründet, dass die prozessgebietsübergreifenden Reifegrade die Erreichung der generischen und spezifischen Praktiken aller erforderlichen Prozessgebiete voraussetzen. Die Erreichung eines Reifegrads für einen Satz von Prozessgebieten setzt daher die Erreichung der generischen Ziele eines einzelnen Prozessgebiets voraus. Sind die generischen Ziele eines einzelnen Prozessgebiets nicht vollständig erreicht, erfolgt die Bewertung mit dem Fähigkeitsgrad „unvollständig“. Die Darstellung eines Satzes von Prozessgebieten mittels Reifegrad ist somit nur sinnvoll, wenn alle im Satz enthaltenen Prozessgebiete mindestens den Fähigkeitsgrad 1 („Durchgeführt“) erreichen.

Tab. 4: Gegenüberstellung der Fähigkeits- und Reifegrade [vgl. SEI 2011, S. 35]

Grad	Darstellung in Fähigkeitsgraden	Darstellung in Reifegraden
0	Unvollständig	
1	Durchgeführt	Initial
2	Geführt	Geführt
3	Definiert	Definiert
4		Quantitativ geführt
5		Prozessoptimierung

Der Fokus dieser Arbeit liegt auf der Entwicklung eines Ansatzes zur Schließung der Lücke zwischen generischem Prozessbezug von Fähigkeits- und Reifegradmodellen und dem konkreten Produktbezug von Methoden der Reifegradabsicherung. Hierfür müssen die spezifischen Ziele und spezifischen Praktiken betrachtet und hinsichtlich resultierender Vorgaben analysiert werden. Die generischen Ziele und generischen Praktiken sind in ihrer Definition zu unkonkret, um hieraus direkte Vorgaben abzuleiten. Entsprechend werden die spezifischen Ziele und Praktiken aller Prozessgebiete nachfolgend hinsichtlich ihrer Relevanz für den Kontext dieser Arbeit bewertet. Hierfür werden zunächst die zugrundeliegenden Auswahlkriterien und Bewertungsstufen im Bewertungsschema vorgestellt.

3.1.1.2.1 Bewertungsschema des CMMI-DEV

Die Bewertung, ob spezifische Ziele oder spezifische Praktiken für die Zielstellung der Arbeit relevant sind, hängt von ihrem Bezug zu konkreten Entwicklungsvorgaben ab. Jedes Prozessgebiet umfasst ein bis drei spezifische Ziele, welche durch jeweils zwei bis sieben spezifische Praktiken untersetzt sind. Für die Bewertung wurden die spezifischen Ziele und die spezifischen Praktiken aller Prozessgebiete auf ihre Relevanz analysiert. Ein spezifisches Ziel ist dann relevant, wenn mindestens eine der das Ziel untersetzenden, spezifischen Praktiken als relevant bewertet ist. Entsprechend wurden die spezifischen Praktiken aller Prozessgebiete bewertet. Die Relevanz des spezifischen Ziels bezieht sich dabei auf die höchste Bewertung der zugehörigen spezifischen Praktik. Ist ein spezifisches Ziel als relevant bewertet, bedeutet dies nicht, dass alle zugehörigen spezifischen Praktiken relevant sind. Analog hierzu ist ein Prozessgebiet relevant, wenn mindestens ein zugehöriges spezifisches Ziel als relevant bewertet ist. Auch hier

impliziert ein als relevant bewertetes Prozessgebiet nicht, dass alle zugehörigen Ziele oder Praktiken Relevanz besitzen müssen.

Die Bewertungskriterien sind nachfolgend in Tab. 5 dargestellt und setzen auf der untersten Ebene der spezifischen Praktiken an. Spezifische Praktiken sind dann relevant, wenn sie direkte Vorgaben zu den definierten Kriterien liefern. Die Bewertungskriterien umfassen die in den vorherigen Kapiteln identifizierten Einflussgrößen.

Tab. 5: Bewertungskriterien spezifischer Ziele und Praktiken

Kriterium	Beschreibung
Frühe Entwicklungsphasen	Spezifische Praktiken sind relevant, wenn ihre Anwendung in den frühen Phasen der Produktentwicklung gefordert ist
Schnittstellenidentifizierung und -analyse	Spezifische Praktiken sind relevant, wenn die Durchführung einer Schnittstellenidentifizierung oder -analyse gefordert ist
Komplexität	Spezifische Praktiken sind relevant, wenn die Komplexitätserfassung, -bewertung oder das Komplexitätsmanagement bezogen auf das Produktfeld gefordert ist
Zuverlässigkeit	Spezifische Praktiken sind relevant, wenn die Identifizierung und/oder Analyse quantitativer oder qualitativer, zuverlässigkeitstechnischer Informationen gefordert ist[12]
Reifegradabsicherung	Spezifische Praktiken sind relevant, wenn die Durchführung einer produktbezogenen Reifegradabsicherung gefordert ist

Diese Bewertungskriterien bilden gemeinsam mit den Bewertungsstufen das Bewertungsschema. Die Abstufung umfasst drei Stufen (keine Relevanz, geringe Relevanz, hohe Relevanz), die in Tab. 6 dargestellt sind. Je Bewertungsstufe wird zwischen spezifischer Praktik, spezifischem Ziel sowie dem Prozessgebiet unterschieden und die Bewertungsstufe über eine kurze Beschreibung erläutert.

[12] Unter dem Bewertungskriterium zur Zuverlässigkeit werden ebenfalls Forderungen aufgenommen, welche eine Ursachenanalyse fordern.

Tab. 6: Bewertungsstufen spezifischer Ziele und Praktiken sowie der Prozessgebiete

Bedeutung	Bezug	Beschreibung
Keine Relevanz	Praktik	Die Praktik enthält keine Forderung bezüglich eines Bewertungskriteriums
	Ziel	Es existiert keine relevante Praktik, welche das Ziel untersetzt
	Prozessgebiet	Es existiert kein relevantes Ziel, welches das Prozessgebiet untersetzt
Geringe Relevanz	Praktik	Die Praktik enthält eine Forderung bezüglich eines Bewertungskriteriums
	Ziel	Es existiert eine relevante Praktik, welche das Ziel untersetzt
	Prozessgebiet	Es existiert ein relevantes Ziel, welches das Prozessgebiet untersetzt
Hohe Relevanz	Praktik	Die Praktik enthält mehr als eine Forderung bezüglich eines Bewertungskriteriums
	Ziel	Es existiert mehr als eine relevante Praktik, welche das Ziel untersetzt
	Prozessgebiet	Es existiert mehr als ein relevantes Ziel, welches das Prozessgebiet untersetzt

Aufgrund des generischen Charakters des CMMI-DEV sind die genannten Bewertungskriterien oftmals nicht explizit gefordert oder erwähnt. Daher wird für die Relevanzbewertung geprüft, ob die Praktiken, Ziele und Prozessgebiete die dargestellten Kriterien direkt oder indirekt enthalten und hieraus eine Anforderung abgeleitet werden kann. So können eine Praktik, ein Ziel oder ein Prozessgebiet als relevant bewertet werden, wenn Anforderungen bezüglich eines Kriteriums enthalten sind, auch wenn dieses nicht explizit aufgeführt ist. Beispielsweise enthält das Prozessgebiet CAR „Causal Analysis and Resolution (Ursachenanalyse und -beseitigung)“ das spezifische Ziel (engl. „specific goal“, SG) SG1 „Ursachen für ausgewählte Ergebnisse ermitteln“ und darunter die spezifische Praktik (engl. „specific practice“, SP) SP1.2 „Ursachen analysieren“. In den zugehörigen Subpraktiken ist beschrieben, dass Probleme zu analysieren sind, um deren Ursachen zu identifizieren [SEI 2011, S. 142ff]. Diese Forderung zur Ursachenanalyse lässt sich auf die Kriterien *Schnittstellenidentifizierung und -analyse* sowie *Zuverlässigkeit* anwenden. Beide Kriterien sind zwar nicht explizit gefordert, ihre Umsetzung stellt jedoch eine Möglichkeit dar, die Forderungen des CMMI-DEV zu erfüllen.

Im Umkehrschluss verweist das CMMI-DEV selbst darauf, dass aufgrund allgemeiner Formulierungen insbesondere informativer Modellkomponenten wie Subpraktiken diese nur eine Möglichkeit der Umsetzung darstellen. Sie sind daher „weder notwendig noch hinreichend, um das Prozessgebiet umzusetzen“ [SEI 2011, S. 73]. Entsprechend können auch andere Lösungen zur Erfüllung genutzt werden. Demzufolge werden die Bewertungskriterien als relevant angesehen, wenn sie eine mögliche Umsetzung der Forderungen darstellen.

3.1.1.2.2 Bewertung des CMMI-DEV

Für die Bewertung der Relevanz wurden alle 22 Prozessgebiete des CMMI-DEV hinsichtlich der Vorgaben für die fünf Kriterien „Frühe Entwicklungsphasen“, „Schnittstellen-

identifizierung und -analyse“, „Komplexität“, „Zuverlässigkeit“ und „Reifegradabsicherung“ analysiert. Diese Anforderungen finden sich beim CMMI-DEV üblicherweise in Form von beispielhaft erwarteten Arbeitsergebnissen oder in Form von Subpraktiken. Beides sind informative Komponenten des CMMI und somit nicht verpflichtend. Im Gegensatz zu den spezifischen Zielen und spezifischen Praktiken sind sie jedoch konkret genug, um eine Anforderung zur Umsetzung ableiten zu können. Für die Bewertung wurde daher mit den beispielhaft erwarteten Arbeitsergebnissen und den Subpraktiken die unterste Ebene der Prozessgebiete betrachtet. Sofern hieraus mindestens eine Anforderung an die Kriterien abgeleitet werden konnte, wurde die Praktik als gering relevant bewertet. Ließen sich Anforderungen für mehrere Kriterien ableiten, wurde die entsprechende spezifische Praktik (SP) mit einer hohen Relevanz bewertet. Enthielt die Praktik keine ableitbaren Anforderungen, wurde sie als nicht relevant bewertet.

Auf Ebene der spezifischen Ziele (SG) wurden die Einzelbewertungen der spezifischen Praktiken zusammengefasst und die Relevanz entsprechend der Bewertungsstufen gebildet. Auf der obersten Ebene wurde die Relevanz der Prozessgebiete bewertet. In Abhängigkeit der im jeweiligen Prozessgebiet enthaltenen spezifischen Ziele ergibt sich eine geringe Relevanz, wenn mindestens ein spezifisches Ziel Relevanz besitzt. Eine hohe Relevanz resultiert, wenn mehr als ein spezifisches Ziel ist. Dabei ist es unerheblich, ob die spezifischen Ziele eine geringe oder hohe Relevanz besitzen. Sofern für ein spezifisches Ziel keine Inhalte vorliegen, ist das jeweilige Ziel nicht anwendbar und wird als n/a (not applicable, dt. nicht anwendbar) gekennzeichnet.

Exemplarisch erfolgt die Bewertung nachfolgend für das Prozessgebiet CAR (Ursachenanalyse und -beseitigung), welches für die Erreichung des Reifegrads 5 erforderlich ist. Das Prozessgebiet CAR besteht aus den beiden spezifischen Zielen SG 1 (Ursachen für ausgewählte Ergebnisse ermitteln) und SG 2 (Ursachen für ausgewählte Ergebnisse angehen). Zunächst wurden die spezifischen Praktiken SP der spezifischen Ziele SG dahingehend analysiert, ob sich Anforderungen für die Bewertungskriterien (Frühe Entwicklungsphasen, Schnittstellen, Komplexität, Zuverlässigkeit, Reifegradabsicherung) ableiten ließen.

Das SG 1 des Prozessgebiets CAR setzt sich zusammen aus zwei spezifischen Praktiken: SP 1.1 (Ergebnisse für die Analyse auswählen) und SP 1.2 (Ursachen analysieren). Für SP 1.1 lautet die Beschreibung „Diese Tätigkeit kann durch ein Ereignis ausgelöst werden (reaktiv) oder zur regelmäßigen Durchführung geplant werden, z.B. zu Beginn einer neuen Phase oder Aufgabe (vorausschauend)“ [SEI 2011, S. 143]. Weder aus der Beschreibung noch aus den Beispielen für Arbeitsergebnisse ließ sich die Relevanz für eines der Bewertungskriterien ableiten. Entsprechend besitzt SP 1.1 keine Relevanz und wurde entsprechend gekennzeichnet.

SP 1.2 definiert, dass „der Zweck dieser Analyse darin [besteht], Maßnahmen festzulegen, um die ausgewählten Ergebnisse zu beseitigen. Dazu werden die relevanten Ergebnisdaten analysiert und Vorschläge für Aktionen zur Umsetzung aufgestellt“ [SEI 2011, S. 144]. Konkreter wird in der zugehörigen Subpraktik 2 (Ausgewählte Ergebnisse analysieren, um ihre Ursachen zu ermitteln) empfohlen, Ursache-Wirkungs-Zusammenhänge zu betrachten, beispielsweise über ein Ishikawa-Diagramm [SEI 2011, S. 145]. SP 1.2 wurde daher für das Bewertungskriterium Zuverlässigkeit als relevant eingestuft. Da SP 1.2 in Summe für nur ein Bewertungskriterium relevant ist, wurde die Relevanz dieser spezifischen Praktik als gering bewertet. Die

aggregierte Bewertung auf Ebene des spezifischen Ziels SG 1 ergibt somit ebenfalls eine geringe Relevanz, da nur eine der beiden spezifischen Praktiken überhaupt eine Relevanz besitzt.

Analog zur Bewertung des spezifischen Ziels SG 1 wurde für das Ziel SG 2 verfahren. Dieses besteht aus den spezifischen Praktiken SP 2.1 (Vorgeschlagene Maßnahmen umsetzen), SP 2.2 (Auswirkungen von umgesetzten Maßnahmen bewerten) und SP 2.3 (Daten der Ursachenanalyse aufzeichnen). Alle drei spezifischen Praktiken sind ausschließlich für das Bewertungskriterium Zuverlässigkeit relevant und sind entsprechend jeweils mit einer geringen Relevanz bewertet. Da alle drei Praktiken jedoch relevant sind, akkumuliert sich die Bewertung des spezifischen Ziels SP 2 zu einer hohen Relevanz, ebenso wie die Relevanz des gesamten Prozessgebiets CAR, da beide untersetzenden Ziele SG 1 und SG 2 relevant sind.

Die Relevanzbewertung erfolgte für alle Praktiken, Ziele und Prozessgebiete nach diesem Schema. In Summe konnten auf diese Weise 14 relevante spezifische Ziele identifiziert werden, deren spezifische Praktiken erwartete Arbeitsergebnisse oder Subpraktiken Anforderungen zu den definierten Kriterien enthielten bzw. aus diesen abgeleitet werden konnten. Hieraus resultieren die neun relevanten Prozessgebiete CAR, PI, PMC, PP, QPM, RD, RSKM, SAM und TS. Tab. 7 fasst diese Ergebnisse zusammen, führt die Relevanz der spezifischen Ziele auf und nennt die daraus gebildete Relevanz des Prozessgebiets. Die vollständige Übersicht inklusive der Relevanzbewertung aller spezifischen Praktiken für die Bewertungskriterien kann dem Anhang entnommen werden.

Tab. 7: Bewertung der Prozessgebiete und spezifischen Ziele

Akronym	Kategorie	Reifegrad	Prozessgebiet	Relevanz Prozessgebiet	SG 1	Relevanz SG 1	SG 2	Relevanz SG 2	SG 3	Relevanz SG 3
CAR	Unterstützung	5	Ursachenanalyse und -beseitigung	hoch	Ursachen für ausgewählte Ergebnisse ermitteln	gering	Ursachen für ausgewählte Ergebnisse angehen	hoch	n/a	n/a
CM	Unterstützung	2	Konfigurationsmanagement	keine	Baselines etablieren	keine	Änderungen verfolgen und lenken	keine	Integrität etablieren	keine
DAR	Unterstützung	3	Entscheidungsfindung	keine	Alternativen bewerten	keine	-	n/a	-	n/a
IPM	Projektmanagement	3	Fortgeschrittenes Projektmanagement	keine	Projektspezifisch definierte Prozesse verwenden	keine	Koordination von und Zusammenarbeit mit relevanten Stakeholdern	keine	-	n/a
MA	Unterstützung	2	Messung und Analyse	keine	Mess- und Analysetätigkeiten ausrichten	keine	Messergebnisse bereitstellen	keine	-	n/a
OPD	Prozessmanagement	3	Organisationsweite Prozessentwicklung	keine	Prozess-Assests der Organisation etablieren	keine	-	n/a	-	n/a
OPF	Prozessmanagement	3	Organisationsweite Prozessausrichtung	keine	Prozessverbesserungsmöglichkeiten bestimmen	keine	Prozessverbesserungen planen und umsetzen	keine	Prozess-Assets der Organisation ausrollen und Lessons Learned	keine
OPM	Prozessmanagement	5	Organisationsweites Leistungsmanagement	keine	Betriebliche Leistung führen	keine	Verbesserungen auswählen	keine	Verbesserungen ausrollen	keine
OPP	Prozessmanagement	4	Organisationsweite Prozessleistung	keine	Prozessleistungsbaselines und -modelle etablieren	keine	-	n/a	-	n/a
OT	Prozessmanagement	3	Organisationsweite Aus- und Weiterbildung	keine	Organisationsweite Fähigkeit zur Aus- und Weiterbildung etablieren	keine	Aus- und Weiterbildung bereitstellen	keine	-	n/a
PI	Entwicklung	3	Produktintegration	hoch	Produktintegration vorbereiten	keine	Schnittstellenkompatibilität sicherstellen	hoch	Produktbestandteile zusammenbauen und das Produkt ausliefern	gering
PMC	Projektmanagement	2	Projektverfolgung und -steuerung	gering	Das Projekt gegenüber dem Plan überwachen	hoch	Korrekturmaßnahmen zum Abschluß führen	keine	-	n/a
PP	Projektmanagement	2	Projektplanung	hoch	Schätzungen etablieren	gering	Projektpläne erstellen	gering	Zusagen zu Plänen einholen	keine
PPQA	Unterstützung	2	Prozess- und Produkt-Qualitätssicherung	keine	Arbeitsabläufe und -ergebnisse objektiv bewerten	keine	Objektiven Einblick geben	keine	-	n/a
QPM	Projektmanagement	4	Quantitatives Projektmanagement	gering	Quantitatives Management vorbereiten	gering	Projekte quantitativ führen	keine	-	n/a
RD	Entwicklung	3	Anforderungsentwicklung	hoch	Kundenanforderungen entwickeln	keine	Produktanforderungen entwickeln	gering	Anforderungen analysieren und validieren	gering
REQM	Projektmanagement	2	Anforderungsmanagement	keine	Anforderungen verwalten	keine	-	n/a	-	n/a
RSKM	Projektmanagement	3	Risikomanagement	gering	Risikomanagement vorbereiten	hoch	Risiken erkennen und analysieren	keine	Risiken abschwächen	keine
SAM	Projektmanagement	2	Zulieferungsmanagement	gering	Vereinbarungen mit Lieferanten treffen	keine	Vereinbarungen mit Lieferanten erfüllen	gering	-	n/a
TS	Entwicklung	3	Technische Umsetzung	hoch	Lösungen für Produktbestandteile auswählen	hoch	Designs entwickeln	hoch	Produktentwürfe umsetzen	keine
VAL	Entwicklung	3	Validierung	keine	Validierung vorbereiten	keine	Produkte oder Produktbestandteile validieren	keine	-	n/a
VER	Entwicklung	3	Verifizierung	keine	Verifizierung vorbereiten	keine	Peer-Reviews durchführen	keine	Ausgewählte Arbeitsergebnisse verifizieren	keine

Von den in den 14 relevanten spezifischen Ziele enthaltenen spezifischen Praktiken konnten insgesamt 23 Praktiken identifiziert werden, welche Anforderungen an die Bewertungskriterien enthalten. Von diesen 23 als relevant bewerteten spezifischen Praktiken enthalten vier Praktiken Anforderungen an frühe Entwicklungsphasen, sieben an die Schnittstellenidentifizierung und -analyse, eine Praktik enthält Anforderungen bezogen auf die Produktkomplexität, vier auf das Themenfeld Zuverlässigkeit, wobei hierunter auch Forderungen nach einer Ursachenanalyse mitgezählt wurden, und sieben Praktiken enthalten Forderungen zur Reifegradabsicherung. Abb. 14 fasst diese Ergebnisse graphisch zusammen.

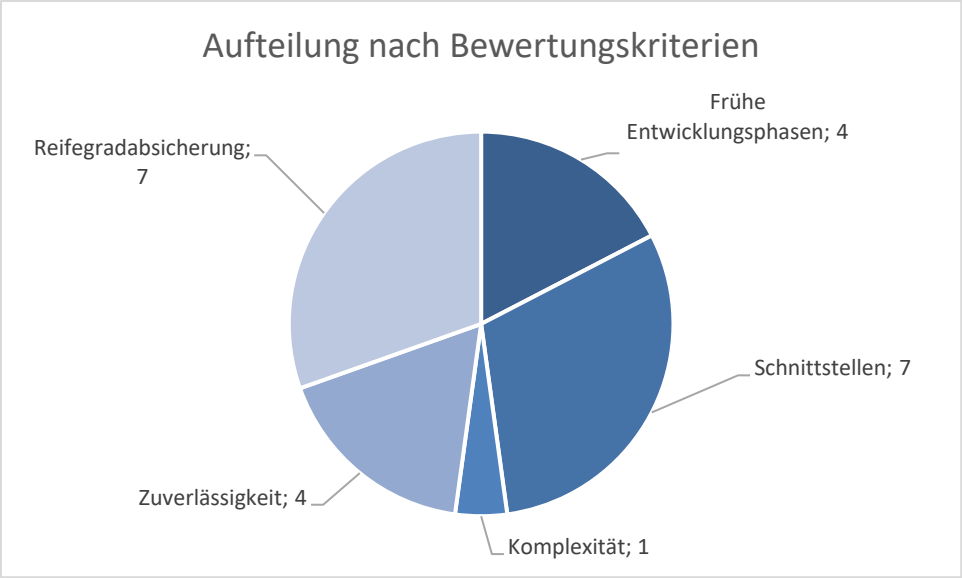


Abb. 14: Aufteilung der relevanten Praktiken nach Bewertungskriterien

Da der Wortlaut der spezifischen Ziele und Praktiken zum Teil nicht explizit als Forderung formuliert ist, wurden für die Ableitung von Anforderungen alle identifizierten Textpassagen umformuliert. Die Gegenüberstellung der umformulierten Anforderungen mit dem ursprünglichen Wortlaut kann dem Anhang entnommen werden. Teilweise wurden die Anforderungen verschiedener spezifischer Praktiken in eine gemeinsame Anforderung zusammengeführt, so dass insgesamt 19 verschiedene Anforderungen resultieren[13], wobei eine Anforderung zwei verschiedenen Kriterien zugeordnet ist. Neben den kriterienbezogenen Anforderungen konnte eine weitere Anforderung identifiziert werden, welche nicht nach dem vorgestellten Schema kategorisierbar ist, jedoch eine wichtige Forderung für den zu entwickelnden Ansatz darstellt.

In Tab. 8 sind die Anforderungen den Kriterien zugeordnet. Für jede Anforderung ist ihre Quelle unter Angabe des Akronyms des Prozessgebiets sowie dem spezifischen Ziel (SG) genannt. Die abgeleiteten Anforderungen werden genutzt, um Vorgaben für den zu entwickelnden Ansatz zu definieren sowie diesen hinsichtlich der Anforderungserfüllung zu bewerten. In mehreren Fällen waren die Forderungen ähnlich, sodass beim Kriterium Frühe Entwicklungsphasen aus zwei bzw. beim Kriterium Reifegradabsicherung aus drei Praktiken eine gemeinsame Anforderung abgeleitet wurde.

[13] In Summe wurden 20 Anforderungen identifiziert, wobei zwei Anforderungen inhaltlich identisch, aber unterschiedlichen Prozessgebieten zugeordnet sind. Diese Anforderungen wurden nur einfach gezählt.

Tab. 8: Abgeleitete Anforderungen des CMMI-DEV

Kriterium	Prozessgebiet und Praktik	Abgeleitete Anforderung
Frühe Entwicklungsphasen	PI, SP 2.2	Ein Schnittstellenmanagement muss frühzeitig implementiert werden
	RD, SP 3.5	Anforderungen müssen frühzeitig validiert werden
	RSKM, SP 1.1	Risiken müssen frühzeitig identifiziert werden
	RSKM, SP 1.3	Eine Risikomanagementstrategie muss frühzeitig festgelegt werden
Schnittstellenidentifizierung und -analyse	PI, SP 2.1	Schnittstellen müssen vollständig erfasst werden
	PI, SP 2.2	Ein Schnittstellenmanagement muss frühzeitig implementiert werden[14]
	PI, SP 3.3	Die Schnittstellenkompatibilität muss am Gesamtsystem bzw. den Subsystemen nachgewiesen werden
	TS, SP 1.2	Schnittstellen zwischen Produktbestandteilen müssen beschrieben werden
	TS, SP 2.1	Es müssen alle externen Schnittstellen und die wichtigen internen Schnittstellen identifiziert werden
	TS, SP 2.3	Schnittstellenkategorien müssen anhand definierter Kriterien entwickelt werden
	RD, SP 2.3	Identifizierte Schnittstellen müssen genutzt werden, um hieraus Anforderungen an das System abzuleiten
Komplexität	TS, SP 1.1	Die Komplexität von Subsystemen muss bei der Auswahl von Lösungen berücksichtigt werden
Zuverlässigkeit	CAR, SP 1.2	Für relevante Ergebnisse müssen die Ursachen ermittelt werden und geeignete Maßnahmen entwickelt und umgesetzt werden
	CAR, SP 2.1	
	CAR, SP 2.2	Die Effektivität getroffener Maßnahmen muss gemessen und analysiert werden
	CAR, SP 2.3	Erkenntnisse der Ursachenanalyse müssen dokumentiert und als Lessons Learned für andere Projekte aufbereitet werden
Reifegradabsicherung	QPM, SP 1.1	Es müssen geeignete Qualitätsziele definiert und auf einzelne Projektphasen bezogen werden, um die Zielerreichung projektbegleitend zu überwachen
	PMC, SP 1.6	
	PP, SP 1.3	Es müssen Meilensteine im Projekt definiert werden, an denen die Produktreife bewertet wird, um hieraus Entscheidungen über den weiteren Projektablauf abzuleiten
	PMC, SP 1.7	
	PP, SP 2.1	
	SAM, SP 2.1	Der Reifegrad zugekaufter Subsysteme muss projektbegleitend überwacht werden
	TS, SP 2.1	Es müssen geeignete Kriterien definiert werden, mit denen eine Reifegradbewertung durchführbar ist
n/a	RD, SP 3.2	Es muss eine funktionale Architektur gebildet werden

[14] Diese Anforderung ist ebenfalls den frühen Entwicklungsphasen zugeordnet

3.1.1.2.3 Fazit zum CMMI-DEV

Das CMMI-DEV ist ein komplexes Modell zur stufenweisen Erreichung reifer Prozesse und einer fähigen Organisation. Hierfür werden Forderungen an zu implementierende Prozesse gestellt, deren Umsetzung als Fähigkeits- bzw. Reifegrad ausgedrückt wird. Die Forderungen beziehen sich auf verschiedene Prozessgebiete und decken nicht nur die eigentliche Produktentwicklung, sondern auch korrelierende Prozesse wie das Projektmanagement oder die organisationsweite Aus- und Weiterbildung ab.

Bezogen auf die Bewertungskriterien können aus dem CMMI-DEV insgesamt 19 Anforderungen abgeleitet werden, die sich schwerpunktmäßig auf die Themengebiete Schnittstellenidentifizierung und -analyse sowie Reifegradabsicherung konzentrieren. Das Bewertungskriterium Komplexität spielt hierbei eine untergeordnete Rolle.

Bedingt durch den generischen Prozessbezug liefert das CMMI-DEV eine Vielzahl zu implementierender Prozesse, welche wiederholbar gute Ergebnisse der Organisation sicherstellen sollen. Die Forderungen des CMMI-DEV (vgl. Tab. 8) sind somit als Eingangsgröße eines zu entwickelnden Ansatzes zu verstehen, wie eine modellbasierte Produktentwicklung in den frühen Entwicklungsphasen umgesetzt werden kann. Das Modell enthält jedoch keine Angaben dazu, wie die Produktentwicklung selbst gestaltet werden sollte. Dies untermauert die These der methodischen Lücke zwischen Ansätzen mit generischem Prozess- und konkretem Produktbezug, welche es zu schließen gilt.

3.1.1.2.4 Abgrenzung zum CMMI 2.0

Das CMMI 2.0 ersetzt seit Oktober 2020 formal das CMMI V1.3 und die sog. Appraisals werden seitdem ausschließlich auf Basis des CMMI 2.0 akzeptiert. Das CMMI 2.0 steht als kostenpflichtiges Modell auf der Internetseite des CMMI Institute zur Verfügung und wird in englischer, spanischer und chinesischer Sprache als Lizenz angeboten[15]. Im Gegensatz zum CMMI V1.3 ist das CMMI 2.0 somit nicht frei verfügbar, auch nicht für wissenschaftliche Zwecke.

Das CMMI 2.0 baut ebenso wie das CMMI V1.3 auf einem gestuften Modell zur Reifegradbewertung auf, wobei beim CMMI 2.0 die Reifegradstufe 0 neu eingeführt wird, welche beim CMMI V1.3 nur für die Darstellung in Fähigkeitsgraden existiert. Somit resultieren für das CMMI 2.0 die Reifegrade *Incomplete*, *Initial*, *Managed*, *Defined*, *Quantitatively Managed* und *Optimizing*. Beide Versionen bieten eine eigene Umsetzung für verschiedene Modelle. Beim CMMI V1.3 sind dies *Aquisition*, *Service* und *Development* (ACQ, SVC, DEV), während beim CMMI 2.0 das Akquisemodell ACQ ersetzt wird durch das Modell *Supplier Management* (SPM). Wesentliche Unterschiede zwischen den Modellen bestehen in ihrer Architektur. Das CMMI-DEV besitzt vier Kategorien (Unterstützung, Projektmanagement, Prozessmanagement, Entwicklung), denen die 22 Prozessgebiete mit ihren jeweiligen spezifischen bzw. generischen Zielen und Praktiken zugeordnet sind, vgl. die Modellkomponenten des CMMI-DEV (Abb. 12). Das CMMI 2.0 führt die neuen Kategorien *Doing*, *Managing*, *Enabling* und *Improving* ein. Innerhalb dieser Kategorien existieren neun sogenannte *Capability Areas*, denen wiederrum die Prozessgebiete (*Practice areas*) zugeordnet sind. Diese unterteilen sich in 20 unterschiedliche Practice Areas gegenüber den 22 Prozessgebieten des CMMI-DEV. Die

[15] https://cmmiinstitute.com/products/cmmi/cmmi-v2-products, zuletzt aufgerufen am 05.05.2022

Prozessgebiete des CMMI 2.0 sind teilweise identisch mit denen des CMMI-DEV (CAR, DAR, TS, PI, OT, CM, SAM) bzw. ähnlich. [Balla et al. 2020]

Zur Referenzierung der Prozessgebiete des CMMI 1.3 auf das CMMI 2.0 stellt das CMMI Institute ein sogenanntes *Practice Mapping* zur Verfügung, in welchem die spezifischen Praktiken aller drei Submodelle (DEC, SVC, ACQ) auf die Praktiken und den Reifegrad des CMMI 2.0 bezogen werden. Generell existieren Entsprechungen aller 20 Practice Areas des CMMI 2.0 beim CMMI-DEV mit Ausnahme der Practice Areas *Implementation Infrastucture* (II) und *Governance* (GOV). Konkret sind über das *Practice Mapping* alle Prozessgebiete des CMMI-DEV den Praktiken des CMMI 2.0 zugeordnet und haben bis auf die SP 1.7 des Prozessgebiets OPD und die SP 1.6 des Prozessgebiets IPM eine oder mehrere entsprechende Praktiken. [CMMI Institute 2018]

Tab. 9 zeigt die neun relevanten Prozessgebiete des CMMI-DEV mit ihren 19 relevanten spezifischen Praktiken und die entsprechenden Praktiken des CMMI 2.0. Allen Prozessgebieten steht mindestens eine Praktik im CMMI 2.0 gegenüber wobei diese z.T. anderen, neuen Capability Areas zugeordnet sind, beispielsweise EST und PLAN für PP.

Tab. 9: Referenzierbare Praktiken beim CMMI 2.0

Prozessgebiet CMM-DEV	Spezifische Praktik CMMI-DEV	Referenzierte Praktiken CMMI 2.0
CAR	1.2	CAR 2.2, CAR 3.2, CAR 4.1
	2.1	CAR 3.3
	2.2	CAR 4.2, CAR 5.1
	2.3	CAR 3.4, CAR 3.5, CAR 4.2, CAR 5.1
PI	2.1	PI 3.1
	2.2	PI 3.1, PI 3.2
	3.3	PI 2.5, PI 3.3
PMC	1.6	MC 3.1
	1.7	MC 3.1
PP	1.3	EST 2.1, PLAN 2.1
	2.1	PLAN 2.3
QPM	1.1	PLAN 4.1
RD	3.2	RDM 2.2, RDM 3.2
	3.5	RDM 3.7
RSKM	1.1	RSK 3.1
SAM	2.1	SAM 2.1, SAM 2.2, SAM 3.1, SAM 3.2
TS	1.2	TS 3.4
	2.1	TS 2.1
	2.3	TS 3.1, TS 3.6

Es zeigt sich, dass bezogen auf die Practices Areas des CMMI 2.0 und die Prozessgebiete des CMMI-DEV eine große Schnittmenge besteht. So existieren für alle 22 Prozessgebiete des CMMI-DEV entsprechende Practice Areas des CMMI 2.0, wobei dieses über zwei zusätzliche Practice Areas verfügt, welche im CMMI-DEV nicht existieren (II, GOV). Bezogen auf die als relevant identifizierten spezifischen Praktiken, aus denen Anforderungen abgeleitet werden konnten (vgl. Tab. 8), zeigt der Vergleich, dass allen relevanten SP im CMMI 2.0 Praktiken gegenüberstehen. Diese sind z.T. mehreren oder anderen Reifegraden zugeordnet, was für die Ableitung von Anforderungen an den Ansatz jedoch irrelevant ist. Als Ergebnis des Abgleichs zwischen CMMI-DEV und CMMI 2.0 wird daher gefolgert, dass die relevanten Bestandteile

gleich oder ähnlich sind, sodass die Verwendung des CMMI-DEV anstelle des CMMI 2.0 inhaltlich korrekt und unter Berücksichtigung der wissenschaftlichen Zugänglichkeit zulässig ist.

3.1.2 Methoden der Reifegradabsicherung

Die Reifegradabsicherung ist ein meilensteinorientierter Ansatz zum Abgleich von geforderten mit vorhandenen Produkteigenschaften. Methoden der Reifegradabsicherung verfolgen das Ziel, durch die Bewertung der Produktreife zu einem gewählten Zeitpunkt eine Entscheidungsgrundlage zu bieten, ob das Produkt in der Lage ist, die gestellten Anforderungen zu erfüllen oder ob zusätzliche Maßnahmen im Projekt erforderlich sind, um die Qualität des Produkts sicherzustellen. Im Gegensatz zu Referenzmodellen, welche einen generischen Prozessbezug verfolgen, besitzen Methoden der Reifegradabsicherung einen konkreten Produktbezug. Hierfür fokussieren Methoden der Reifegradabsicherung den Produktreifegrad im Sinne der Anforderungserfüllung zu einem bestimmten Meilenstein.

Die Meilensteine, zu denen eine Bewertung des Produktreifegrads erfolgt, werden oftmals auch als Quality Gate bezeichnet, da in Abhängigkeit des Bewertungsergebnisses eine Entscheidung hinsichtlich der Projektfortsetzung getroffen wird. Über die PEP-Phasen sind üblicherweise verschiedene Quality Gates verteilt, die sich an typischen Meilensteinen und den zu diesen Zeitpunkten erwarteten Ergebnissen orientieren. Methoden der Reifegradabsicherung liefern dabei die Struktur, indem im Vorfeld definiert wird, welchen Entwicklungsstand ein Produkt an einem Quality Gate besitzen soll. Üblicherweise wird ebenfalls ein Bewertungsschema vorgegeben. Die nachfolgende Abb. 15 veranschaulicht die Reifegradabsicherung an einem abstrakten Quality Gate und liefert als Ergebnis eine Entscheidung zum weiteren Projektfortschritt. Hierbei folgt das Quality Gate als Meilenstein auf eine Aktivität in der Phase *n*. Mit Abschluss der Aktivität wird das erreichte Ergebnis bewertet, wobei im Vorfeld definierte Anforderungen herangezogen werden. In Abhängigkeit der Bewertung erfolgt eine Entscheidung, ob die erreichten Ergebnisse ein Fortschreiten zur nächsten Phase *n+1* zulassen, ob Optimierungen erforderlich sind oder ggf. das Projekt beendet werden muss.

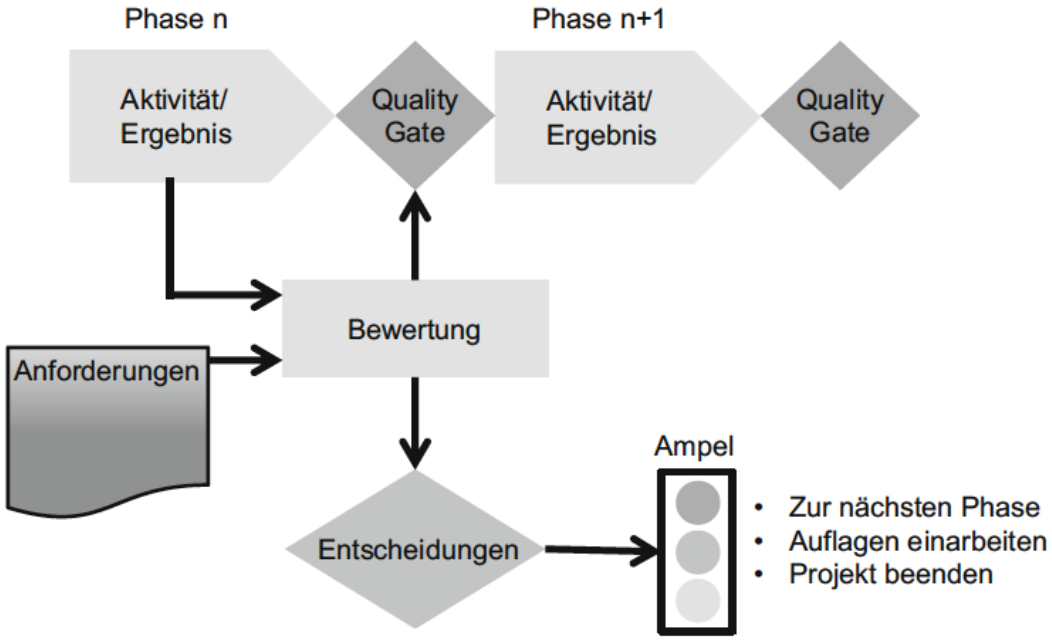


Abb. 15: Quality Gates in der Entwicklung und Konstruktion [Feldhusen/Grote 2021, S. 885]

Die Erfassung eines Produktreifegrads setzt die Definition von Anforderungen bezogen auf erwartete Produkteigenschaften und -merkmale zu einem Quality Gate voraus. Die Schwierigkeit besteht darin, dass bedingt durch die Heterogenität von Produkten und Entwicklungsprozessen die Ableitung von Anforderungen und Bewertungskriterien für die Reifegradbewertung und -absicherung oftmals unternehmens- oder produktspezifisch erfolgt und somit für den zu entwickelnden Ansatz nicht zur Verfügung steht. Für die nachfolgende Analyse von Methoden der Reifegradabsicherung wird daher das Reifegradabsicherungs-Modell des VDA betrachtet, welches für verschiedenste Applikation angewandt werden kann und somit die Forderung nach einem unternehmens- und produktunspezifischen Ansatz erfüllt.

3.1.2.1 Reifegradabsicherungs-Modell

In der Automobilindustrie verlagert sich die Entwicklungstiefe zunehmend von den OEMs zu den Lieferanten [VDA 2009, S. 6], welche vermehrt als Entwicklungs- und Systemlieferant agieren anstatt als reiner Teilelieferant. Diese Änderung erfordert eine Bewertung der Produktreife in der gesamten Lieferkette, um frühzeitig mögliche Abweichungen zu identifizieren und bei Bedarf Gegenmaßnahmen einzuleiten. Der VDA empfiehlt, die Reifegradabsicherung auf allen Stufen der Lieferkette anzuwenden, d.h. jeweils zwischen Lieferant und Kunde. Der Anwendung vorausgehend wird eine Risikoeinstufung mit Hilfe einer ABC-Analyse empfohlen, bei der der jeweilige Kunde anhand verschiedener Kriterien der Kategorien „Lieferumfangbezogene Kriterien (Produkt)“, Produktionsprozess bezogene Kriterien (Prozess)“, Terminbezogene Kriterien“ und „Lieferantenbezogene Kriterien“ seinen Lieferanten bewertet. Diese Risikobewertung führt zu einer Einstufung als A-Lieferant („hohes Reifegradrisiko“), B-Lieferant („mittleres Reifegradrisiko“) oder C-Lieferant („geringes Reifegradrisiko“) und bestimmt, ob der Lieferant die Reifegradabsicherung selbst bzw. unter Aufsicht des Kunden durchführen kann oder ob der Kunde – bei einem hohen Reifegradrisiko – direkt in die Bewertung des Reifegrads involviert werden muss [VDA 2009, S. 11ff].

Das Reifegradabsicherungs-Modell (RGA-Modell) des VDA setzt sich aus acht aufeinander aufbauenden Reifegraden RG0 bis RG7 zusammen und umfasst alle Phasen der Produktentwicklung von der Konzeptphase bis zum Serienanlauf[16]. Während die Inhalte der Reifegrade klar benannt sind, ist ihre Absicherung abhängig vom Projektterminplan. Die erforderlichen Quality Gates können somit relativ frei geplant werden, wobei RG0 in der Konzeptphase ansetzt und RG7 nach dem SOP erfolgt. Jeder Reifegrad besteht aus mehreren Reifegradindikatoren, wie Projektmanagement oder PPF (Produkt- und Prozessfreigabe), welche zum Teil Bestandteil mehrerer Reifegrade sind. Sie entfallen jedoch, sobald sie als Indikator nicht mehr benötigt werden. So ist mit dem RG6 „Produkt- und Prozessfreigabe“ die Produktentwicklung abgeschlossen und das Produkt geht in die Serienfertigung über. Für den RG7 „Projektabschluß, Verantwortungsübergabe an Serie, Start Requalifikation“ rücken daher andere Reifegradindikatoren in den Fokus, wie beispielsweise das Änderungsmanagement.

Zur Bewertung des Reifegrads existieren insgesamt 12 Indikatoren, die verschiedene Themenbereiche abdecken, wobei abhängig von der Relevanz die Indikatoren nur ausgewählten Reifegraden zugeordnet sind. So ist beispielsweise der Indikator zum Projektmanagement relevant für alle Reifegrade, während der Indikator zur Absicherung der Serie nur für die Reifegrade

[16] Der Serienanlauf umfasst die Phase von bis zu sechs Monaten nach SOP, vgl. [VDA 2009, S. 15].

RG3 und RG7 Anforderungen enthält [VDA 2009, S. 17]. Den Aufbau des RGA-Modells veranschaulicht Abb. 16. Die Reifegrade RG0 bis RG7 sind hier sequentiell auf einer Zeitachse aufgeführt und nennen jeweils die dem Reifegrad zugehörigen Indikatoren.

Abb. 16: Übersicht über die Reifegrad-Inhalte [VDA 2009, S. 15]

Jedem Reifegradindikator ist ein Satz von Messkriterien zugeordnet. Die Messkriterien geben vor, welche Ergebnisse bezogen auf einen Reifegradindikator zu einem bestimmten Reifegrad umgesetzt sein müssen und variieren zwischen den Reifegraden. Entsprechend können identische Indikatoren für unterschiedliche Reifegrade voneinander abweichende Messkriterien enthalten. Die Messkriterien sind dabei so formuliert, dass sie auf Entwicklungsergebnisse aller beteiligten Domänen angewendet werden können. Messkriterien können hierbei auch in Form von Wiederholfragen definiert sein. Diese enthalten keine neuen Anforderungen, sondern sichern die Erfüllung von in vorherigen Reifegraden definierten Anforderungen ab. Generell sind alle Messkriterien vollständig zu erfüllen, um in der Reifegrad-Ampel-Kaskade (siehe Abb. 17) eine grüne Bewertung zu erhalten. [VDA 2009, S. 16f]

3.1.2.2 Ablauf der Reifegradbewertung

Die Reifegradbewertung erfordert die Durchführung der Schritte Voraussetzung, Start und Steuerung. Während die Schritte Voraussetzung und Start initial erfolgen und die Rahmenbedingungen der Reifegradbewertung schaffen, unterteilt sich die Steuerung in die drei Phasen Vorbereitung, Bewertung und Umsetzung und wird für jeden Reifegrad durchlaufen [VDA 2009, S. 18, S. 25].

Als Voraussetzung der Reifegradbewertung müssen organisatorische und technische Aspekte wie die Freigabe der Durchführung, vorhandene Lastenhefte, Lieferumfänge und eine Risikobewertung zwischen Lieferant und Auftraggeber abgestimmt werden. Für den Start der Reifegradbewertung müssen u.a. die Meilensteine entlang der Lieferkette sowie die zugeordneten Reifegrade RG0 bis RG7 terminiert werden. Weiterhin müssen Verantwortliche für die

Methode der Reifegradbewertung sowie die Inhalte benannt, eine Terminplanung abgeleitet sowie Kriterien für die Übergabe von der Entwicklungs- in die Serienphase definiert werden [VDA 2009, S. 18ff]. Der RG0 wird vollständig durch den Auftraggeber durchgeführt, während alle nachfolgenden Reifegrade in Abhängigkeit des Messkriteriums durch den Auftraggeber und den Lieferanten durchgeführt werden, vgl. [ebd., S. 41ff].

Der Schritt „Steuerung“ stellt den Kern der Reifegradbewertung dar und ist in die Unterphasen Vorbereitung, Bewertung und Umsetzung gegliedert. Die Vorbereitung umfasst organisatorische Aspekte und wird nicht vertiefend behandelt. Für die Bewertung existieren je Reifegradindikator mehrere Messkriterien, deren Umsetzung einzeln bewertet wird, wobei jedes Kriterium mit „ja“ oder „nein“ beantwortet werden kann. In Abhängigkeit der Antwort und gegebenenfalls erforderlicher Aktivitäten resultiert eine Ampelbewertung. Eine grüne Reifegradbewertung wird erreicht bei einer mit „ja“ beantworteten Frage und unter der Voraussetzung, dass keine korrektiven Aktivitäten erforderlich sind. Muss ein Messkriterium mit „nein“ beantwortet werden, resultiert eine gelbe Reifegradbewertung, sofern die zuvor festgelegten Projektziele mit zu definierenden Aktivitäten erreichbar sind. Sollte die Maßnahme eine Zielanpassung erforderlich machen, resultiert eine rote Reifegradbewertung. Für die Gesamtbewertung des Reifegradindikators wird die schlechteste Bewertung der Messkriterien herangezogen. Sofern ein rot bewertetes Messkriterium vorliegt, welches eine Zielanpassung erforderlich macht und sich auf das Projektziel auswirkt oder einen Lieferantenwechsel erforderlich macht, sind im Anschluss alle zuvor erreichten Reifegrade zu wiederholen.

Die sich aus gelben und roten Reifegradbewertungen ergebenden Maßnahmen müssen so terminiert werden, dass sie bis zum nächsten Reifegrad umgesetzt werden. Dies ist Bestandteil der Phase „Umsetzung“. Außerdem müssen spätestens beim RG7 alle Messkriterien mit grün bewertet sein. Hieraus resultiert die Reifegrad-Ampel-Kaskade, welche über die Projektlaufzeit exemplarisch die Bewertung von Reifegraden darstellt, vgl. Abb. 17. Das Beispiel zeigt die Bewertung einzelner Reifegrade an definierten RG-Meilensteinen und stellt die Logik der Gesamtbewertung dar, welche sich aus der jeweils schlechtesten Bewertung eines einzelnen Reifegrads zum jeweiligen RG-Meilenstein ergibt.

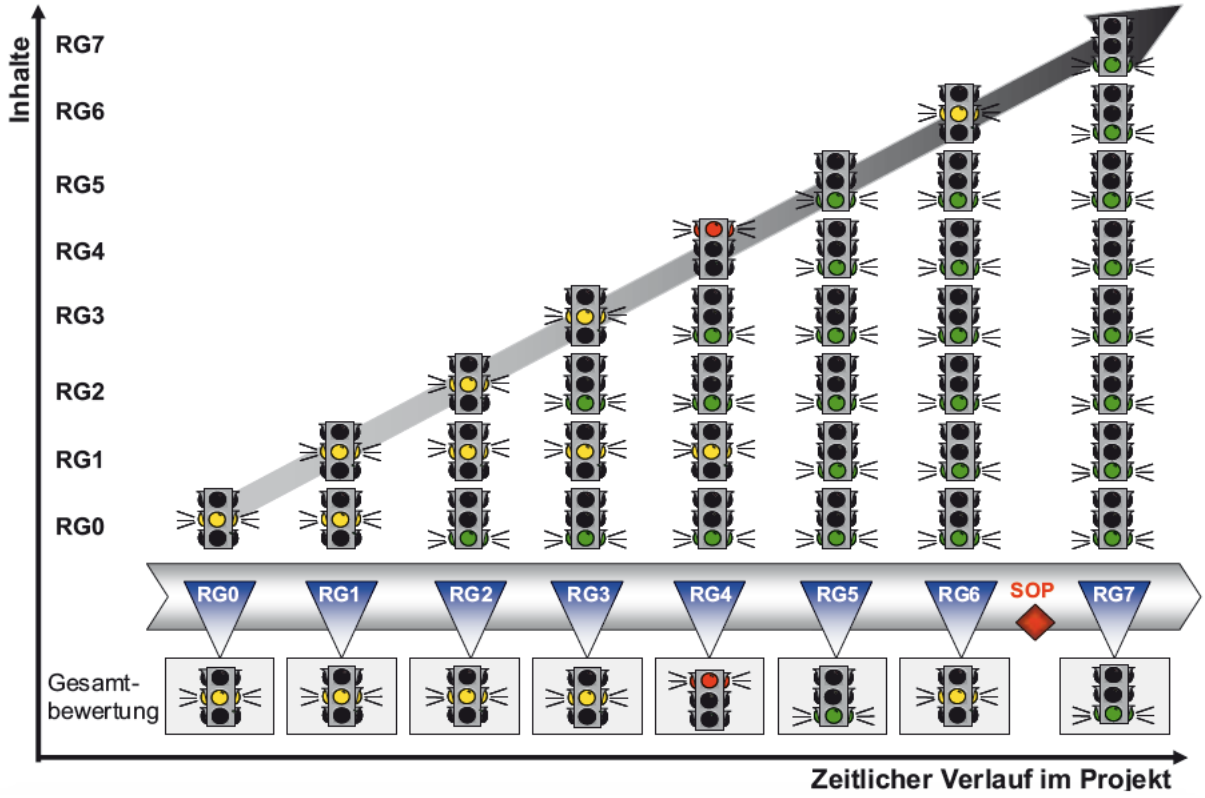


Abb. 17: Beispiel einer Reifegrad-Ampel-Kaskade [VDA 2009, S. 28]

3.1.2.3 Relevante Reifegrade, Indikatoren und Messkriterien des RGA-Modells

Für die Ableitung von Anforderungen an den zu entwickelnden Ansatz besitzen nur die Reifegradindikatoren Relevanz, die sich auf die frühen Phasen der Produktentwicklung beziehen. Da im RGA-Modell die Meilensteine projektspezifisch festgelegt werden können, müssen die Reifegrade inhaltlich hinsichtlich ihrer Anwendbarkeit in den frühen Phasen der Produktentwicklung bewertet werden. Es werden nur Messkriterien analysiert, welche in Reifegraden aufgeführt sind, die diesen Phasen zugeordnet werden können. Dies beinhaltet die Reifegrade RG0 „Innovationsfreigabe und Serienentwicklung“, RG1 „Anforderungsmanagement für Vergabeumfang“, RG2 „Festlegung der Lieferkette und Vergabe der Umfänge“ sowie RG3 „Freigabe der technischen Spezifikation“. Während beim RG2 ein vergabefähiges Angebot auf Basis eines abgestimmten Lastenhefts vorliegt [VDA 2009, S. 50f], existiert beim RG3 bereits ein abgestimmtes Pflichtenheft, auf dessen Basis nachgewiesen wird, dass der vorliegende Entwicklungsstand die Kundenanforderungen erfüllt [VDA 2009, S. 61f]. Dies entspricht dem Systementwurf, welcher als Output die frühen PEP-Phasen gegenüber nachgelagerten Projektphasen abgrenzt. RG4 „Produktionsplanung abgeschlossen“ setzt bereits die Beauftragung der Fertigungsanlagen voraus, was eine weitestgehend abgeschlossene Produktentwicklung impliziert. Entsprechend werden RG4 und nachfolgende Reifegrade in der Analyse nicht berücksichtigt. Für die Bewertung werden somit die Messkriterien der Indikatoren analysiert, die in den RG0 bis RG3 enthalten sind.

Tab. 10 zeigt die Zuordnung der Indikatoren zu den Reifegraden auf. Anforderungen können sowohl als Messkriterien enthalten sein (dargestellt durch einen schwarzen Kreis) als auch in Form von Wiederholfragen (dargestellt durch einen schwarz umrandeten, ungefüllten Kreis). Wiederholfragen enthalten keine neuen Anforderungen, sondern sichern ihre konsistente

Erfüllung durch die erneute Abfrage ab. Für die Ableitung der Anforderungen werden nur Messkriterien betrachtet, da Wiederholfragen keine neuen Anforderungen enthalten. Sind sowohl neue Messkriterien als auch Wiederholfragen enthalten, werden dem RG beide Kategorien zugeordnet.

Tab. 10: Zuordnung der Indikatoren zu den Reifegraden[17]

Indikator	RG0	RG1	RG2	RG3	RG4	RG5	RG6	RG7
Innovation / Konzept / Zuverlässigkeit	●	●	○	○	○	○	○	
Projektmanagement	●	●○	●○	○	○	○	○	●
Risikomanagement	●	●	●	●	○	○		
Beschaffungsprozess	●	●	●		●			
Lieferkette / Teileversorgung		●	●○	●○	●○	●○	●○	
Änderungsmanagement			●	○	○	○	○	●
Produktentwicklung		●	●	●	●	●		
Produktabsicherung		●	●	●○	○	○	○	
Prozessentwicklung			●	●	●			
Prozessabsicherung			●	●○	●○	●○	○	●
Produkt- und Prozessfreigabe (PPF)			●		●	●	●	
Absicherung Serie				●				●

Für die frühen Phasen der Produktentwicklung, welche inhaltlich durch die Reifegraden RG0 bis RG3 beschrieben werden, sind alle Indikatoren potentiell relevant, sodass ihre Messkriterien hinsichtlich Anforderungen analysiert werden müssen. Die Messkriterien werden gemäß den nachfolgend definierten Kriterien geprüft und für die Bewertung des zu entwickelnden Ansatzes genutzt.

3.1.2.4 Bewertungsschema des RGA-Modells

Das Bewertungsschema des RGA-Modells ist grundsätzlich identisch mit dem des Reifegrad- und Fähigkeitsmodells CMMI-DEV. Bei den Bewertungskriterien entfällt für das RGA-Modell jedoch das Kriterium zu frühen Phasen der Produktentwicklung, da durch die zu analysierenden Reifegrade RG0 bis RG3 bereits ein direkter Bezug zu diesen Phasen gegeben ist und nicht gesondert gefordert werden muss. Ein weiterer Unterschied liegt im Kriterium zur produktbezogenen Reifegradabsicherung. Während das Reifegrad- und Fähigkeitsmodell CMMI-DEV einen generischen Prozessbezug besitzt und somit Anforderungen mit einem konkreten Produktbezug besondere Anforderungen darstellen, ist beim RGA-Modell der konkrete Produktbezug grundsätzlich gegeben und wird nicht gesondert hervorgehoben. Das Kriterium zur Reifegradabsicherung wird daher ersetzt durch den Funktionsnachweis. Dieses Kriterium ist relevant, wenn Messkriterien den Nachweis vereinbarter Funktionen am Gesamtsystem oder an Teilsystemen fordern. Tab. 11 fasst die Bewertungskriterien für das RGA-Modell zusammen und beschreibt die Voraussetzung für die Bewertung als relevantes Kriterium.

[17] vgl. [VDA 2009, S. 17]

Tab. 11: Bewertungskriterien der Indikatoren und Messkriterien

Kriterium	Beschreibung
Schnittstellenidentifizierung und -analyse	Messkriterien sind relevant, wenn die Durchführung einer Schnittstellenidentifizierung oder -analyse gefordert ist
Komplexität	Messkriterien sind relevant, wenn die Komplexitätserfassung, -bewertung oder das Komplexitätsmanagement bezogen auf das Produktfeld gefordert ist
Zuverlässigkeit	Messkriterien sind relevant, wenn die Identifizierung und/oder Analyse quantitativer oder qualitativer zuverlässigkeitstechnischer Informationen gefordert ist[18]
Funktionsnachweis	Messkriterien sind relevant, wenn die Durchführung eines produktbezogenen Funktionsnachweises gefordert ist

Die Relevanz wird hierbei in den gleichen Abstufungen bewertet wie beim CMMI-DEV. Beim Bezug wird, wie auch beim CMMI-DEV, auf der untersten Ebene das Vorhandensein von Forderungen zum jeweiligen Kriterium abgefragt, vgl. 3.1.1.2.1. Analog hierzu erfolgt die Bewertung auf Ebene der Indikatoren. Sofern ein Messkriterium für einen Reifegrad RG0 bis RG3 mindestens mit einer geringen Relevanz bewertet ist, wird der zugehörige Indikator mit einer geringen Relevanz bewertet. Sind mindestens zwei Messkriterien eines Reifegrads mindestens mit einer geringen Relevanz bewertet, besitzt der Indikator eine hohe Relevanz.

Tab. 12 fasst die Bewertungsstufen zusammen und beschreibt die Einstufung der Relevanz jeweils für die Messkriterien und Indikatoren.

Tab. 12: Bewertungsstufen der Indikatoren und Messkriterien

Bedeutung	Bezug	Beschreibung
Keine Relevanz	Messkriterien	Die Messkriterien enthalten keine Forderung bezüglich der Bewertungskriterien für einen Reifegrad
	Indikator	Es existiert kein relevantes Messkriterium für den zugehörigen Reifegrad
Geringe Relevanz	Messkriterien	Die Messkriterien enthalten eine Forderung bezüglich der Bewertungskriterien für einen Reifegrad
	Indikator	Es existiert mindestens ein Messkriterium mit geringer Relevanz für einen Reifegrad
Hohe Relevanz	Messkriterien	Die Messkriterien enthalten mehr als eine Forderung bezüglich der Bewertungskriterien für einen Reifegrad
	Indikator	Es existiert mindestens ein Messkriterium mit geringer Relevanz für mehrere Reifegrade

3.1.2.5 Bewertung des RGA-Modells

Für die Bewertung des RGA-Modells nach VDA wurden die Messkriterien und Indikatoren der Reifegrade RG0 bis RG3 hinsichtlich der identifizierten Kriterien analysiert und bewertet. Dies

[18] Unter dem Bewertungskriterium zur Zuverlässigkeit werden ebenfalls Forderungen aufgenommen, welche eine Ursachenanalyse fordern.

wird exemplarisch für den Indikator Innovation/Konzept/Zuverlässigkeit aufgezeigt. Dieser besitzt Messkriterien für die Reifegrade RG0 und RG1; für die nachfolgenden Reifegrade existieren bis auf den RG7 Wiederholfragen. Beim RG0 wird bei der Messkriteriumsnummer 0.2.2 für den Indikator gefordert, dass „Wechselwirkungen mit Schnittstellen im Gesamtfahrzeug bekannt [sind]“, ergänzt um die Bemerkung, dass dies „inkl. der Systeme, Subsysteme und Komponenten, elektrischen, mechanischen und geometrischen Systemgrenzen etc.“ gelte [VDA 2009, S. 41]. Der RG0 des Indikators Innovation/Konzept/Zuverlässigkeit wurde daher für das Kriterium Schnittstellenidentifizierung und -analyse als relevant bewertet. Da er für andere Kriterien nicht relevant ist, resultiert eine geringe Relevanz des RG0 für den Indikator.

Analog hierzu erfolgte die Bewertung des RG1 für den Indikator Innovation/Konzept/Zuverlässigkeit. Hierbei konnte eine Relevanz für das Kriterium Zuverlässigkeit festgestellt werden. Da keine weitere Relevanz für die anderen Kriterien vorlag, resultiert eine geringe Relevanz des RG1 für den Indikator. Der Indikator als Oberelement erhält somit eine hohe Relevanz, da seine beiden ihn untersetzenden Reifegrade relevant sind.

Dieses Bewertungsschema wurde für alle Indikatoren für ihre jeweils existierenden Reifegrade durchgeführt. Von den zwölf analysierten Indikatoren besitzen in Summe vier Indikatoren eine geringe und zwei Indikatoren eine hohe Relevanz. Diese sechs relevanten Indikatoren enthalten insgesamt zehn relevante Messkriterien, die sich auf die Bewertungskriterien aufteilen, vgl. Tab. 13. Dabei führen acht Messkriterien, die sich jeweils auf einen Indikator beziehen, zu einer geringen Relevanz, während zwei Messkriterien, die sich auf den gleichen Indikator beziehen, zu einer hohen Relevanz. Die genaue Bewertung der Messkriterien bezogen auf die Bewertungskriterien kann dem Anhang entnommen werden.

Tab. 13: Bewertung der Indikatoren

Indikator	Relevanz Indikator	Relevanz RG0	Relevanz RG1	Relevanz RG2	Relevanz RG3
Innovation/Konzept/Zuverlässigkeit	hoch	gering	gering	n/a	n/a
Projektmanagement	gering	keine	keine	gering	n/a
Risikomanagement	gering	keine	gering	keine	keine
Beschaffungsprozess	gering	keine	gering	keine	n/a
Lieferkette/Teileversorgung	keine	n/a	keine	keine	keine
Änderungsmanagement	keine	n/a	n/a	keine	n/a
Produktentwicklung	gering	n/a	keine	keine	gering
Produktabsicherung	hoch	n/a	gering	hoch	gering
Prozessentwicklung	keine	n/a	n/a	keine	keine
Prozessabsicherung	keine	n/a	n/a	keine	keine
Produkt- und Prozessfreigabe (PPF)	keine	n/a	n/a	keine	n/a
Absicherung Serie	keine	n/a	n/a	n/a	keine

Von den zehn relevanten Messkriterien entfallen fünf auf das Bewertungskriterium Schnittstellenidentifizierung und -analyse, ein Messkriterium auf das Themenfeld Zuverlässigkeit und vier

auf den Funktionsnachweis. Für das Kriterium der Komplexität konnten keine Anforderungen identifiziert werden. Abb. 18 fasst dieses Ergebnis graphisch zusammen.

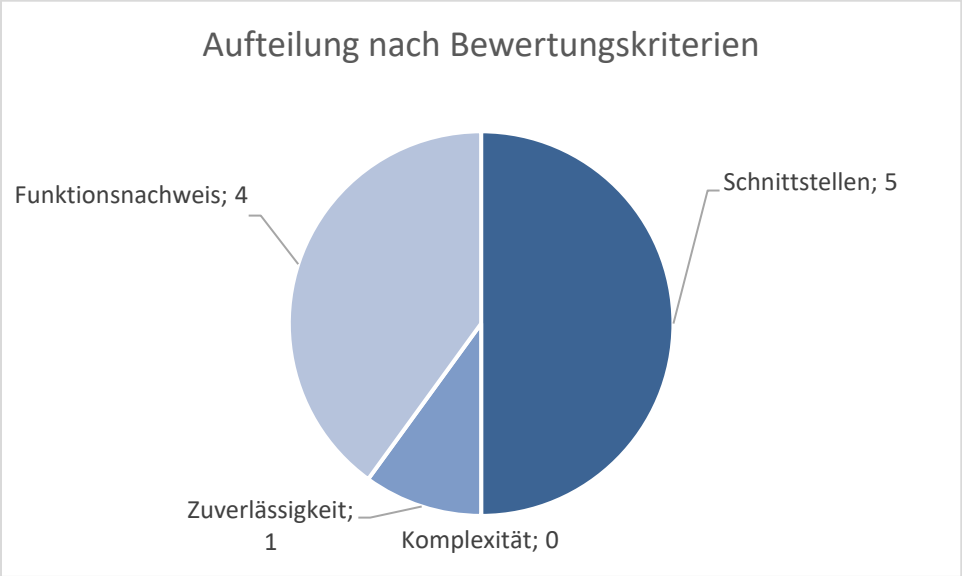


Abb. 18: Aufteilung der Messkriterien nach Bewertungskriterien

Wie bereits bei den spezifischen Zielen und Praktiken des CMMI-DEV, wurden auch aus den Forderungen der als relevant identifizierten Messkriterien des RGA-Modells Anforderungen abgeleitet. Hierbei wurde für die Indikatoren Risikomanagement beim RG1 und Projektmanagement beim RG2 für das Bewertungskriterium Schnittstellenidentifizierung und -analyse eine gemeinsame Anforderung abgeleitet. Gleichzeitig wurde für den Indikator Produktabsicherung beim RG1 für das Bewertungskriterium Funktionsnachweis die Forderung des Messkriteriums in zwei einzelne abgeleitete Anforderungen aufgeteilt, sodass in Summe zehn Anforderungen abgeleitet werden konnten, vgl. Tab. 14.

Tab. 14: Abgeleitete Anforderungen des RGA-Modells

Kriterium	Indikator	Reifegrad	Abgeleitete Anforderung
Schnittstellenidentifizierung und -analyse	Innovation/ Konzept/ Zuverlässigkeit	RG0	Eine Systemgrenze muss definiert und Schnittstellen und Wechselwirkungen identifiziert werden
	Beschaffungs-Prozess	RG1	Schnittstellen und Wechselwirkungen müssen mit Lieferanten von Subsystemen besprochen werden
	Risiko-management	RG1	Bei modularen Anwendungen müssen Schnittstellen mit anderen Projekten identifiziert und bewertet werden
	Projekt-management	RG2	
	Produkt-absicherung	RG2	Schnittstellen müssen analysiert werden
Kom-plexität	n/a	n/a	n/a
Zuver-lässigkeit	Innovation/ Konzept/ Zuverlässigkeit	RG1	Zuverlässigkeitsprognosen müssen erstellt werden
Funktionsnachweis	Produkt-absicherung	RG1	Ein Funktionsnachweis muss auf Ebene des Gesamtsystems erbracht werden
			Eine Planung der erforderlichen Simulationen und Tests für den Nachweis der Anforderungserfüllung muss erstellt werden
	Produkt-absicherung	RG2	Tests zum Nachweis der Anforderungserfüllung müssen alle vereinbarten Merkmale umfassen
	Produkt-entwicklung	RG3	Der Systementwurf muss hinsichtlich Anforderungserfüllung überprüft werden
	Produkt-absicherung	RG3	Für das System und seine Subsysteme müssen Funktionsnachweise auf ihrer jeweiligen Ebene erbracht werden

3.1.2.6 Fazit zum RGA-Modell

Das Reifegradabsicherungs-Modell des VDA dient der produktbezogenen Reifegradabsicherung entlang der Produktentwicklungsphasen und bietet sich insbesondere für komplexe Entwicklungsvorhaben an, bei der die Entwicklung zwischen einem Gesamtsystemverantwortlichen und (Sub-)Systemlieferanten sichergestellt werden muss. Ein besonderer Fokus liegt daher auf der Erfassung, Analyse und Überwachung von Schnittstellen zwischen Systembestandteilen sowie dem Nachweis der Anforderungserfüllung in späteren Phasen.

Bezogen auf die Bewertungskriterien zeigt sich, dass die Schnittstellenidentifizierung und -analyse sowie der Funktionsnachweis wesentliche Bestandteile des RGA-Modells bilden. Entsprechend konnten Anforderungen abgeleitet werden, welche den Nachweis dieser Kriterien einfordern. Mit Blick auf die beschriebene Lücke zwischen Modellen mit generischem Prozessbezug wie dem CMMI-DEV und Modellen mit konkretem Produktbezug kann das RGA-

Modell jedoch nicht zu deren Schließung beitragen. Das RGA-Modell liefert ebenso wie das Fähigkeits- und Reifegradmodell CMMI-DEV Anforderungen, welche Kriterien zur Erreichung eines produktbezogenen Reifegrads in den frühen Phasen der Produktentwicklung erfüllt werden müssen. Einen Ansatz für die Produktentwicklung selbst, d.h. wie die Entwicklung in frühen PEP-Phasen umgesetzt werden kann, liefert das RGA-Modell nicht. Somit bleibt die methodische Lücke, einen Ansatz für die spezifische Umsetzung in frühen Phasen der Produktentwicklung anzubieten, bestehen.

3.1.3 Fazit zu Fähigkeits- und Reifegradmodellen

Die Entwicklung komplexer technischer Systeme stellt die Produktentwicklung vor Herausforderungen, deren Nichtbeherrschung zu Ausfällen, nicht erfüllten Kundenanforderungen oder der mangelnden Erreichung zeitlicher und monetärer Projektziele führt. Es konnte gezeigt werden, dass Fehler insbesondere an Schnittstellen zwischen den beteiligten Domänen auftreten [vgl. Schlund et al. 2009]. Während die Fehlerentstehung oftmals den frühen Entwicklungsphasen zugeordnet werden kann, liegt der Schwerpunkt der Fehlerentdeckung in nachgelagerten Phasen [vgl. Westkämper 2006, S. 131]. Dies stellt bezüglich der möglichen Fehlerbehebung sowie der hierfür notwendigen monetären Aufwendungen ein wesentliches Problem der Produktentwicklung dar, welches mittels der *Rule of Ten* beschrieben wird [vgl. Bertsche et al. 2009, S. 200, Pfeifer 2001, S. 188f, von Regius 2006, S. 9]. Es konnte gefolgert werden, dass diese Probleme letztlich auf nicht vorhandene Informationen über ein zu entwickelndes System oder mangelnde methodische Unterstützung im Umgang mit diesen Informationen zurückgeführt werden können [vgl. Crostack/Klute 2008, S. 404, Dorociak 2012]. Mit zunehmendem Informationsgehalt über ein System steigt jedoch die Komplexität, deren unzureichendes Management neue Probleme generiert [Lindemann et al. 2009].

Die Etablierung von Produktentwicklungsprozessen, bei denen systematisch gute Praktiken implementiert werden, dieses Vorgehen standardisiert und im konkreten Projekt kontinuierlich die Produktreife nachgewiesen wird, ist ein wesentliches Erfolgsmerkmal von Unternehmen. Zur Erreichung dieses Ziels wurden Modelle entwickelt, welche entweder die Fähigkeit und den Reifegrad der Prozesse fokussieren oder die die Anforderungserfüllung vom Konzept über den Systementwurf bis zum fertigen Produkt absichern. Während Modelle mit generischem Prozessbezug Forderungen an die Entwicklungsprozesse stellen, setzen Modelle der Reifegradabsicherung auf den konkreten Produktbezug und sichern die Qualität entwicklungsbegleitend ab.

Fähigkeits- und Reifegradmodelle wie das CMMI-DEV [SEI 2011] ermöglichen dabei eine inkrementelle Entwicklung von Organisationen. Die hierfür definierten Stufen der Fähigkeits- und Reifegrade zeigen auf, welche Prozesse zu implementieren sind und welche Ergebnisse erwartet werden. Diese Vorgaben sind jedoch so generisch, dass kein Vorgehen für die Produktentwicklung selbst abgeleitet werden kann. Das *Wie* bleibt damit unbeantwortet.

Ansätze mit konkretem Produktbezug wie das RGA-Modell des VDA ermöglichen die Bewertung des Entwicklungsfortschritts zu definierten Meilensteinen, den sogenannten Quality Gates [VDA 2009]. Weicht beispielsweise der zu einem bestimmten Quality Gate erfasste von dem zu erwartenden Produktreifegrad ab, sind Maßnahmen erforderlich, um das Delta auszugleichen. In Entwicklungsprojekten resultieren hieraus üblicherweise Iterationen und Rücksprünge. Während Entwicklungsiterationen als positiv und erforderlich angesehen werden, stellen

Rücksprünge einen zusätzlichen, unerwünschten Entwicklungsaufwand dar. Als Gründe hierfür werden unklare Anforderungen und Informationsdefizite genannt, beides insbesondere in frühen Entwicklungsphasen [Krehmer et al. 2009, S. 181]. Ansätze der Reifegradabsicherung eignen sich demzufolge für eine entwicklungsbegleitende Qualitätssicherung. Die Frage, wie eine anforderungsgerechte Produktentwicklung umzusetzen ist, beantwortet das RGA-Modell allerdings ebenso wenig wie das CMMI-DEV.

Damit bleibt die spezifische Umsetzung in der Produktentwicklung als Lücke zwischen dem generischen Prozessbezug und dem konkreten Produktbezug bestehen. Die vorgestellten Ansätze definieren hierfür Vorgaben und Ergebnisse bezogen auf den Prozess und das Produkt, welche in Form von Anforderungen herausgearbeitet wurden. Es bedarf jedoch eines Ansatzes zur Synchronisierung von Fähigkeits- und Reifegradmodellen und Methoden der Reifegradabsicherung. Für die Umsetzung müssen insbesondere die Herausforderungen im Kontext der Entwicklung komplexer technischer Systeme in den frühen Produktentwicklungsphasen fokussiert werden.

3.2 Komplexitätsmanagement

Bezogen auf die Komplexität wurde bereits ihre Bedeutung als eine der drei Herausforderungen der Produktentwicklung herausgestellt (Kap. 1.2) sowie eine dieser Arbeit zugrundeliegende Definition vorgestellt (Kap. 2.3). Auf dieser Basis ist nun festzulegen, welche Optionen das Komplexitätsmanagement prinzipiell bietet, welche Zielrichtung im Umgang mit Komplexität verfolgt wird und wie das Komplexitätsmanagement methodisch unterstützt werden kann.

Um Komplexität im Produktentwicklungsprozess handhabbar zu machen, sind geeignete Ansätze zum Umgang mit ihr erforderlich. Eine wesentliche Dimension stellt die Zielrichtung dar. So identifiziert MAURER als übliche Lösungsansätze die Vermeidung oder Reduzierung von Komplexität, erweitert diesen Lösungsraum jedoch dadurch, dass auch die Steigerung von Komplexität zur Problemlösung beitragen kann [Maurer 2007, S. 3]. Basierend auf der Erkenntnis, dass die aus der Komplexität resultierenden Probleme durch bisherige Ansätze nicht beherrscht werden, entwickeln LINDEMANN ET AL. einen neuen Ansatz für das Komplexitätsmanagement. Wesentliches Merkmal des Ansatzes ist die Identifizierung von Strukturen, d.h. die Beschreibung der Abhängigkeiten zwischen Systemelementen. Die hierfür wesentlichen Begriffsdefinitionen von System[19], Struktur[20] und Komplexität[21] heben das Verständnis hervor, dass Komplexitätsmanagement nicht auf die Reduzierung, sondern das Beherrschen von Komplexität über Systemstrukturen abzielt. Für die Analyse von Systemstrukturen wird bereits an

[19] „A system is created by compatible and interrelated parts that form a system structure, possess individual properties, and contribute to fulfill the system's purpose. Systems are delimitated by a system border and connected to their surroundings by inputs and outputs. Changes to parts of a system can be characterized by dynamic effects and result in a specific system behavior." [Lindemann et al. 2009, S. 23f]

[20] „The network formed by dependencies between system elements and which represents a basic attribute of each system. Structures can be characterized by the specific compilation of implied linkages between system elements and can be divided into subsets. Structures and their subsets can be analyzed by means of computational approaches, primarily provided by the graph theory." [Lindemann et al. 2009, S. 24f]

[21] „Complexity represents an attribute of systems and can be divided into several aspects: numerical, relational, variational, disciplinary and organizational complexity. These aspects include the number of components, dependencies and variants as important characteristics of complexity. These aspects also address the number of disciplines involved and the distribution of work for the degree of complexity. Furthermore, complexity can result from internal or external sources, but only the internal ones can be actively managed." [Lindemann et al. 2009, S. 29f]

dieser Stelle auf die Graphentheorie verwiesen, welche eine notwendige Grundlage für das Komplexitätsmanagement bildet. [Lindemann et al. 2009, S. 1ff, S. 24f]

Werden die Definition von LINDEMANN ET AL. bei der Beschreibung eines Systems zugrunde gelegt, steigt die Komplexität nicht nur mit steigendem Reifegrad des Systems, sondern ebenfalls über eine präzisere Beschreibung eines technisch-funktional identischen Systementwurfs. Diese Erkenntnis verdeutlicht, dass die Zielrichtung im Umgang mit Komplexität nicht ausschließlich auf der Vermeidung liegen darf, da ihr Ziel andernfalls bei einer ungenauen und weniger komplexen Systembeschreibung erreicht wäre. Vielmehr verdeutlicht sie, dass nicht die Komplexität an sich, sondern ihr unzureichendes Management für Probleme in der Produktentwicklung sorgt: „Complexity in product design does not represent a problem per se; rather it is the lack of ability of users to control complexity that leads to severe consequences“ [Lindemann et al. 2009, S. 31].

Für den Umgang mit Komplexität nennen LINDEMANN ET AL. die Erfassung und Bewertung von Strukturen komplexer Systeme, die Vermeidung und Reduzierung sowie das Management und die Beherrschung von Komplexität als mögliche Strategien. Die Strukturerfassung stellt die Basis im Umgang mit Komplexität dar. Hierbei haben sich matrizenbasierte Ansätze als vorteilhaft bewiesen, die auf der Design Structure Matrix (DSM) basieren. Die Bewertung der Komplexität kann anhand identifizierter Systemstrukturen der Elemente erfolgen [ebd., S. 31ff]. Das Management und die Beherrschung von Komplexität ist im Kern oftmals auf ein Variantenmanagement begrenzt, was nur eine Teilmenge des Komplexitätsmanagements darstellt. Zudem mangelt es bislang an der Berücksichtigung struktureller Aspekte [ebd., S. 35f].

Für die vorliegende Arbeit stellt die Strukturerfassung und Bewertung von Schnittstellen die Zielrichtung dar. Die Berücksichtigung von Systemstrukturen als Stellhebel der Komplexität ermöglicht somit Vorteile, die sich insbesondere bei Änderungen am System im Rahmen des Engineering Change Managements (ECM) zeigen. Dabei ist die Visualisierung von Systemstrukturen über Graphen und Matrizen von zentraler Bedeutung. Gleichzeitig können neben den unmittelbar in Matrizen oder Graphen enthaltenen Informationen (Elemente und Knoten) weitere Informationen enthalten sein, welche nicht ohne weiteres transformierbar sind, wie z.B. eine Clusterung innerhalb einer Matrix. Matrix und Graph werden daher für das Vorgehenskonzept parallel genutzt.

Bei der Informationsaufbereitung mittels Matrizen haben sich daher sogenannte Design Structure Matrix-Ansätze durchgesetzt [Maurer/Lindemann 2007, Lindemann et al. 2009]. In Anlehnung an MAURER/LINDEMANN[22] unterscheiden LINDEMANN ET AL. zwischen der *Intra-domain matrix*, der *Inter-domain matrix* und den zusammengesetzten Formen *Combinated intra-domain and inter-domain matrix* sowie *Multiple-domain matrix* (MDM). Abb. 19 stellt diese Matrizen dar und zeigt auf, in welcher Matrix Relationen zwischen Domänen betrachtet werden.

[22] Maurer, M.; Lindemann, U.: *Facing Multi-Domain Complexity in Product Development.* Cidad Working Paper Series 3 (2007) 1, pp 1-12.

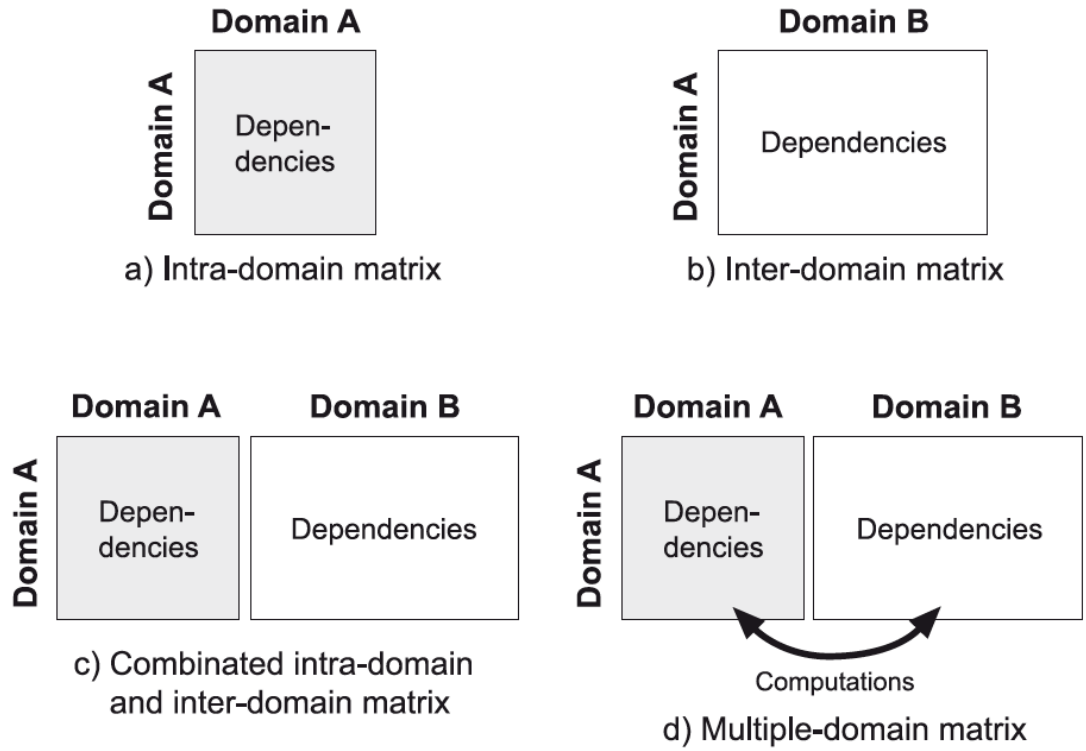


Abb. 19: Classification of matrix-based methods [Lindemann et al. 2009, S. 50]

Die *Intra-domain matrix* beschreibt als Quadratmatrix die Wechselbeziehungen zwischen Elementen innerhalb einer Domäne und entspricht einer Design Structure Matrix (DSM). Die Wechselbeziehungen können entweder binär oder numerisch sowie statisch oder dynamisch beschrieben werden [ebd., S. 49ff]. Die *Inter-domain matrix* beschreibt in gleicher Weise Wechselbeziehungen zwischen Elementen zweier verschiedener Domänen. Für diese hat sich der Begriff Domain Mapping Matrix (DMM) durchgesetzt [ebd., S. 54]. Als Mischform der DSM und DMM können über die *Combinated intra-domain and inter-domain matrix* Informationen dargestellt werden, welche sowohl innerhalb als auch zwischen Domänen existieren. Ein bekanntes Beispiel hierfür ist das House of Quality (HoQ) des Quality Function Deployments (QFD) [Linß 2011, S. 194ff], bei dem DSM und DMM parallel verwendet werden [Lindemann et al. 2009, S. 54ff]. Die *Multiple-domain matrix* (MDM) kann über die Informationen der DSM und DMM berechnet werden und formt eine eigene Matrix [ebd., S. 56ff]. Sie ist damit in der Lage, Informationen aus Graphen abzubilden, die sowohl domäneninterne als auch domänenübergreifende Informationen enthalten.

Für die vorliegende Arbeit besitzt die MDM keine Relevanz, da die betrachteten Relationen mit der DSM und der DMM hinreichend genau abbildbar sind. Stattdessen liegt der Fokus im Umgang mit komplexen Systemen auf einer eindeutigen und reproduzierbaren Systembeschreibung sowie der Identifizierung und Darstellung aller relevanten Wechselbeziehungen und Schnittstellen. Für die Umsetzung eines strukturellen Komplexitätsmanagements müssen frühzeitig alle Informationen zum System (Elemente und Relationen) identifiziert werden. Dies ermöglicht die Erfassung von Komplexität und muss transparent und rückverfolgbar geschehen. Erst im Anschluss kann die Analyse der strukturellen Komplexität erfolgen, an die sich die von MAURER definierten Möglichkeiten des Komplexitätsmanagements, Vermeidung, Reduzierung oder Steigerung [Maurer 2007, S. 3], anschließen. Hinzu kommt als weitere Option die Belassung der Komplexität auf dem festgestellten Niveau.

Den Fokus der vorliegenden Arbeit bilden die Identifizierung, Darstellung und die Analyse von Schnittstellen und Wechselbeziehungen. Dies stellt zwar keinen Ansatz zur Umsetzung des Managements struktureller Komplexität im Produktfeld dar, es ist jedoch eine notwendige Bedingung, um nachfolgend ein Komplexitätsmanagement zu ermöglichen. Die Zielrichtung, d.h. die Frage, ob Komplexität vermieden, reduziert, belassen oder erhöht werden sollte, geht über die hier thematisierte Fragestellung hinaus und wird nicht betrachtet.

3.3 Systemmodellierung

Zur Beherrschung der Herausforderungen der Produktentwicklung stellt die entwicklungsbegleitende Systemmodellierung einen zentralen Bestandteil des Konzepts dar. Sie ist zudem ein geeignetes Werkzeug, um strukturelle Komplexität zu visualisieren. Hinsichtlich des Stands der Wissenschaft und Forschung werden daher zunächst die theoretischen Hintergründe der Graphentheorie dargelegt, welche unabhängig vom gewählten Systemmodell Allgemeingültigkeit besitzen. Im Anschluss wird die Systemmodellierung gemäß dem Systemverständnis des Demand Compliant Designs (DeCoDe) aufgezeigt.

Während einer Systemmodellierung immer ein Systemverständnis zugrunde liegen muss bezüglich existierender Systemsichten und dem Verhältnis der zugehörigen Systemelemente zueinander, ist das verwendete Modell nicht per se auf ein konkretes Systemverständnis – wie dem DeCoDe – festgelegt. Vielmehr stellt die Entscheidung zugunsten von DeCoDe eine notwendige Festlegung im Rahmen der Arbeit dar, ohne dass der eigentliche Kern – die Entwicklung eines Vorgehenskonzepts – damit eingeschränkt wird. Es ist erforderlich, dass das Systemverständnis übliche Systemsichten wie Anforderungen, Funktionen und Komponenten umfasst. Gleiches gilt auch für deren Modellierung: die Visualisierung mittels eines Werkzeugs, in diesem Fall LOOMEO, stellt eine mögliche Modellierungsoption dar, ohne dass diese Auswahl das Vorgehenskonzept beeinflusst. Zur Visualisierung der Systemmodellierung ist es daher erforderlich, dass Systemsichten und deren Elemente beschrieben und sowohl in der Graphen- als auch in der Matrizendarstellung visualisierbar sind. Darüberhinausgehende Funktionen eines Modellierungstools sind nicht Bestandteil des Fokus der vorliegenden Arbeit.

3.3.1 Graphentheorie

Wie für das Komplexitätsmanagement aufgezeigt, besitzt die Darstellung von Systemen über DSM und DMM eine hohe Bedeutung, um Relationen zwischen Systemelementen transparent darzustellen. Zur Erreichung der geforderten Transparenz und Reproduzierbarkeit muss somit eine Systemmodellierung in der Produktentwicklung implementiert werden. Dies knüpft an die Erkenntnis an, dass in der Produktentwicklung oftmals nicht klar ist, wie sich eine Relation oder die Änderung eines Systembestandteils auf das System auswirkt. Auch bezugnehmend auf die Reifegradbewertung für ein konkretes Produkt ist es schwierig, die generierten Informationen ohne die Nutzung eines Systemmodells transparent darzustellen. WINZER stellt hierzu fest: „Bei den definierten Quality-Gates [...] erfolgt die Prüfung des Grades der Anforderungserfüllung häufig über Checklisten oder protokollierte Workshops. Damit entsteht eine Vielzahl von Informationen, die nicht zu verbinden und die nicht – oder nur schwer – dem Denkmodell des jeweiligen betrachteten Systems zuzuordnen sind“ [Winzer 2016, S. 264]. Dieses Denkmodell, d.h. eine abstrakte Darstellung des Systems unter der Berücksichtigung von Regeln und

Gesetzmäßigkeiten der Modellierung, kann generierte Informationen darstellen und trägt somit auch zur Komplexitätsbeherrschung bei.

Unabhängig vom Inhalt eines Systemmodells können Gesetzmäßigkeiten der Systemmodellierung identifiziert werden, welche Auswirkungen auf den Aufbau des Systemmodells und auf das Vorgehenskonzept besitzen. Hierbei finden insbesondere Erkenntnisse und Gesetzmäßigkeiten der Graphentheorie Anwendung [Steger 2007, Turau 2009]. In deren Sinne lassen sich Systeme als Netzwerk aus Knoten (Elemente) und Kanten (Relationen) beschreiben.

Während in der Graphentheorie über Netze alle Relationen zwischen Elementen über Knoten und Kanten darstellbar sind, bietet die Matrizendarstellung eine weitere Art der Modellierung. Graphen- und Matrizendarstellungen unterscheiden sich dabei grundsätzlich nur in der Aufbereitungsart verfügbarer Informationen und sind in die jeweils andere Form transformierbar: „Graphs can represent the same information as matrices, and both forms can be translated from one to the other“ [Andrásfai 1991, S. 133].

In der Systemmodellierung werden Elemente über Relationen miteinander in Beziehung gesetzt, wobei für die Matrizendarstellung die Richtung der Beziehung durch Gesetzmäßigkeiten bei der Modellierung definiert ist (*Zeile beeinflusst Spalte*). Systembestandteile können über Klassen oder Domänen zusammengefasst werden und spezifische Attribute besitzen. Attribute sind zugewiesene Eigenschaften, die für Klassen, Elemente und Relationen festlegbar sind. Demnach sind Systemmodelle in der Graphen- und der Matrizendarstellung über die definierten Bestandteile *Elemente*, *Relationen*, *Klasse* bzw. *Domäne* sowie *Attribute* beschreibbar.

Für die Darstellung von Systemmodellen über Graphen und Matrizen ist es von Bedeutung, welche Richtung eine Relation zwischen Elementen besitzt. Die Richtung stellt ein Attribut der Relation dar und gibt an, ob die Beziehung zwischen Elementen ungerichtet oder gerichtet ist. Weiterhin können Elementen und Relationen unterschiedliche Klassen zugewiesen sein oder es bestehen mehrere Beziehungen zwischen Elementen, so dass diese über mehrere Relationen zu beschreiben sind. Da es sich bei Graphen und Matrizen lediglich um unterschiedliche Darstellungsvarianten handelt, müssen sich alle Informationen über ein System in beiden Varianten wiederfinden. Im Folgenden wird über unterschiedliche Relationen jeweils die Graphen- und Matrizendarstellung abgeleitet und für einfache und fortführend für Systeme mit erhöhter Informationsverfügbarkeit hinsichtlich Richtung (ungerichtet/gerichtet) und Anzahl der Relationen (Einfachkante/Mehrfachkante) sowie Klasse der Elemente beschrieben.

3.3.1.1 Ungerichtete Relationen

Ein System mit ungerichteten Relationen enthält Elemente, die zueinander in Wechselbeziehung stehen, ohne dass dieser das Attribut „Richtung“ zugewiesen wird. Systemmodelle mit ungerichteten Relationen stellen die einfachste Form von Modellen dar.

3.3.1.1.1 Ungerichtete Relationen ohne Mehrfachkanten

Systemmodelle ohne Mehrfachkanten stellen Relationen zwischen Elementen nur auf eine Art und Weise dar. Dementsprechend können sie nicht zwischen unterschiedlichen Relationen, die zwischen Elementen vorliegen könnten, differenzieren. Im Beispiel besitzt ein aus den Elementen α, β und γ bestehendes System Relationen zwischen allen Elementen, die zueinander in

ungerichteter Relation stehen, d.h. jedes Element beeinflusst die anderen Elemente und wird gleichzeitig von ihnen beeinflusst. Abb. 20 stellt dies in Graphen- und Matrizendarstellung dar.

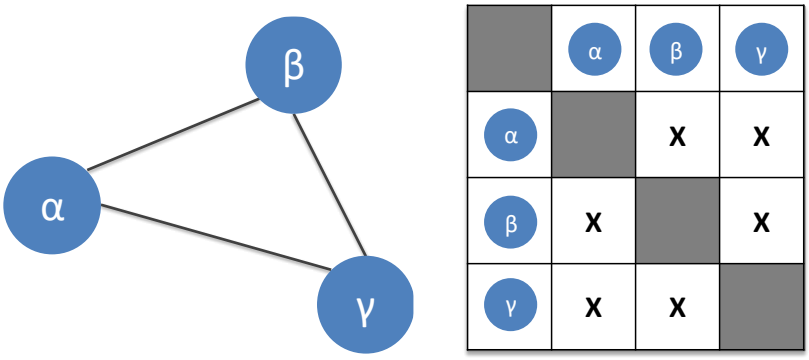


Abb. 20: Ungerichtete Relation ohne Mehrfachkanten

In der Matrizendarstellung werden aufgrund der Gesetzmäßigkeiten von Matrizen (*Zeile beeinflusst Spalte*) grundsätzlich gerichtete Relationen beschrieben. Um eine ungerichtete Relation zwischen Elementen in einer Matrix zu beschreiben, ist es erforderlich, eine gerichtete Relation in beide Richtungen einzutragen, wodurch ihre Richtung aufgehoben bzw. bidirektional wird. Dementsprechend existiert kein Unterschied zwischen einer ungerichteten und einer bidirektionalen, d.h. in beide Richtungen wirkenden Relation.

3.3.1.1.2 Ungerichtete Relation mit Mehrfachkanten

Existieren zwischen Elementen mehrere Relationsarten, werden diese im Graphen über Mehrfachkanten dargestellt. Jede Relationsart beschreibt eine andere Beziehung zwischen Elementen. Dadurch erhält die Relation das Attribut „Art“, welche den Inhalt der Relation beschreibt. Dies kann erforderlich sein, wenn in einem technischen System Komponenten über mehr als nur eine Art miteinander in Relation stehen, wie z.B. Stoff- und Signalfluss. Jede Relationsart wird in diesem Fall im Graphen mit einer Kante dargestellt. Im Beispiel (vgl. Abb. 21) existieren zwischen den Elementen α und β sowie α und γ zwei unterschiedliche Relationsarten (Mehrfachkanten). In der Graphendarstellung wird dies durch die durchgezogene Linie für die Relationsart „A“ und mittels gestrichelter Linie für die Relationsart „B“ beschrieben.

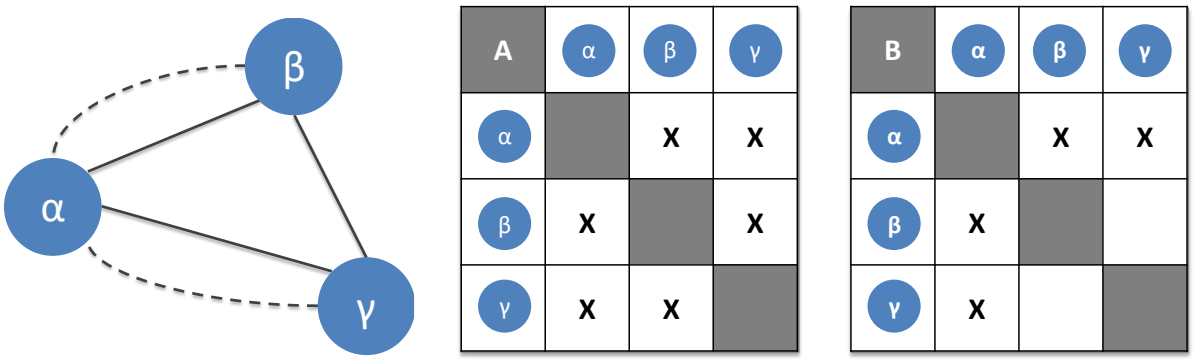


Abb. 21: Ungerichtete Relation mit Mehrfachkanten

Für die Matrizendarstellung bewirkt das Vorhandensein von Mehrfachkanten, dass die Relationen nicht mehr in einer Matrix darstellbar sind, da innerhalb einer Matrix nicht zwischen unterschiedlichen Relationen differenziert werden kann. Entsprechend erfolgt die Matrixdarstellung des Modells in zwei Matrizen, welche die Relation A bzw. B darstellen.

3.3.1.2 Gerichtete Relationen

Modelle mit gerichteten Relationen enthalten das Attribut „Richtung“. Die Richtung einer Relation zeigt an, in welchem Verhältnis Elemente zueinanderstehen. Eine Richtung kann beispielsweise für eine Hierarchie oder die Wirkrichtung einer Beeinflussung stehen.

3.3.1.2.1 Gerichtete Relationen ohne Mehrfachkanten

Im aufgeführten Beispiel existiert eine gerichtete Relation des Elements α auf die Elemente β und γ sowie vom Element γ auf das Element β. Die in der Graphendarstellung in Abb. 22 gezeigten Relationen kennzeichnen die Richtung der Relation über einen Pfeil.

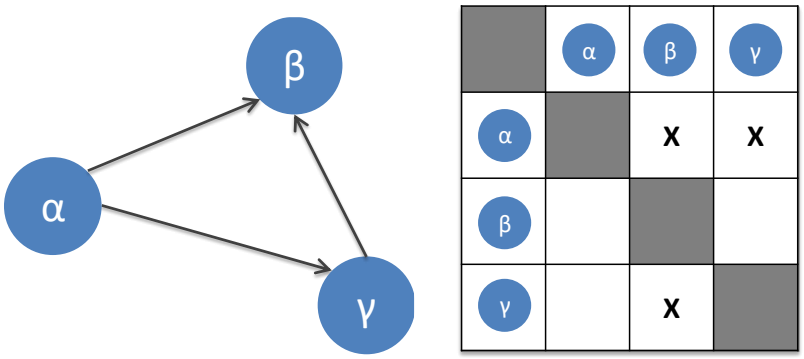


Abb. 22: Gerichtete Relation ohne Mehrfachkanten

Die gerichtete Relation ohne Mehrfachkanten lässt sich in einer Matrix darstellen. Da gilt, dass in der Matrix eine Zeile die Spalte beeinflusst, ist die Richtung der Relation direkt aus dieser ersichtlich.

3.3.1.2.2 Gerichtete Relationen mit Mehrfachkanten

In Modellen mit gerichteten Relationen mit Mehrfachkanten können Relationen zwischen Elementen dargestellt werden, die sowohl die Attribute „Richtung“ als auch „Art“ besitzen. Während es für das Attribut „Richtung“ nur drei mögliche Ausprägungen gibt (von einem Element 1 zum Element 2, vom Element 2 zum Element 1 oder ungerichtet bzw. bidirektional zwischen Element 1 und Element 2), kann das Attribut „Art“ beliebig viele Ausprägungen annehmen. Dabei muss jede Ausprägung durch eine eigene Relationsart dargestellt werden.

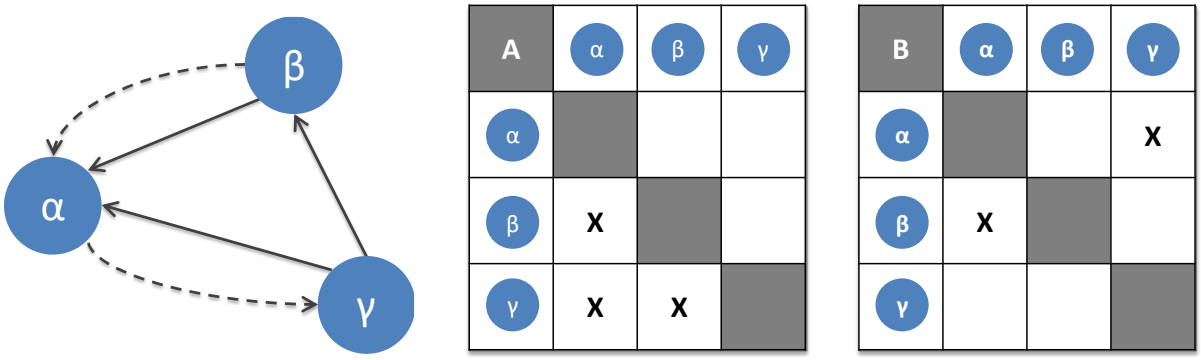


Abb. 23: Gerichtete Relation mit Mehrfachkanten

Im aufgeführten Beispiel in Abb. 23 existieren zwei unterschiedliche Relationsarten, die durch Linien gekennzeichnet sind. Typ A steht für Relationen mit durchgezogener Linie und Typ B für Relationen mit gestrichelter Linie. Indirekte Relationen, im Beispiel bei der Relation B zwischen β und γ (durch die direkten Relationen von β über α auf γ), werden im

Komplexitätsmanagement zum Teil genutzt bzw. gezielt identifiziert. Für die Systemmodellierung im Kontext des Vorgehenskonzepts besitzen indirekte Relationen jedoch keine Relevanz und werden nicht gesondert hervorgehoben oder in der Matrizendarstellung modelliert, d.h. es existiert keine Relation vom Typ B zwischen β und γ im Graphen oder der Matrix.

3.3.1.3 Klassenbildung

Eine wichtige Eigenschaft von Modellen ist die Möglichkeit, Elemente auf Basis zugewiesener Attribute in Klassen einzuteilen bzw. zu clustern. Auf diese Weise lassen sich Elemente zusammenführen, denen eine über das Attribut beschriebene gemeinsame Eigenschaft zugrunde liegt. In Modellen technischer Systeme beziehen sich Klassen oftmals auf die logische oder bauliche Anordnung von Bauteilen oder beschreibender Systemmerkmale.

Zur Veranschaulichung dient ein System, welches aus zwölf Elementen besteht, die sich in den Attributen Form, hier Quadrat (Q), Dreieck (D), Raute (R), Kreis (K), und Farbe, hier blau, grün, rot, orange, unterscheiden und zwischen denen gerichtete Relationen bestehen, vgl. Abb. 24. Die Elemente sind nach ihrem Attribut Form klassiert (geclustert), wobei eine Klassenbildung auch entsprechend eines anderen Attributs wie beispielsweise Farbe der Elemente hätte erfolgen können. Es handelt sich demnach um einen in Klassen eingeteilten, gerichteten Graphen ohne Mehrfachkanten.

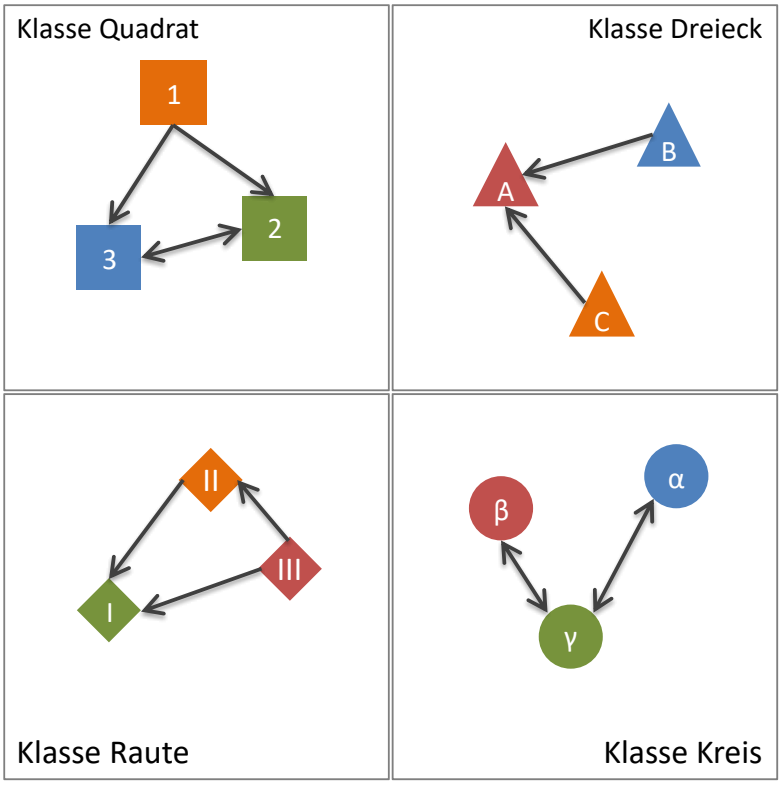


Abb. 24: Graphendarstellung klassierter Elemente mit gerichteten Relationen ohne Mehrfachkanten

Diese klassierten Graphen ohne Mehrfachkanten lassen sich wie in Kap. 3.3.1.2.1 jeweils über eine Matrix darstellen, siehe Abb. 25. Hierbei handelt es sich um eine DSM, welche die enthaltenen Elemente je Klasse nur sich selbst gegenüberstellen. Bei klassierten Elementen mit gerichteten Relationen, die nur innerhalb einer Klasse und ohne Mehrdachkanten existieren, ist die Matrizendarstellung eine mögliche Alternative zur Graphendarstellung, ohne dabei Transparenz- und Rückverfolgbarkeitseinbußen zu verursachen.

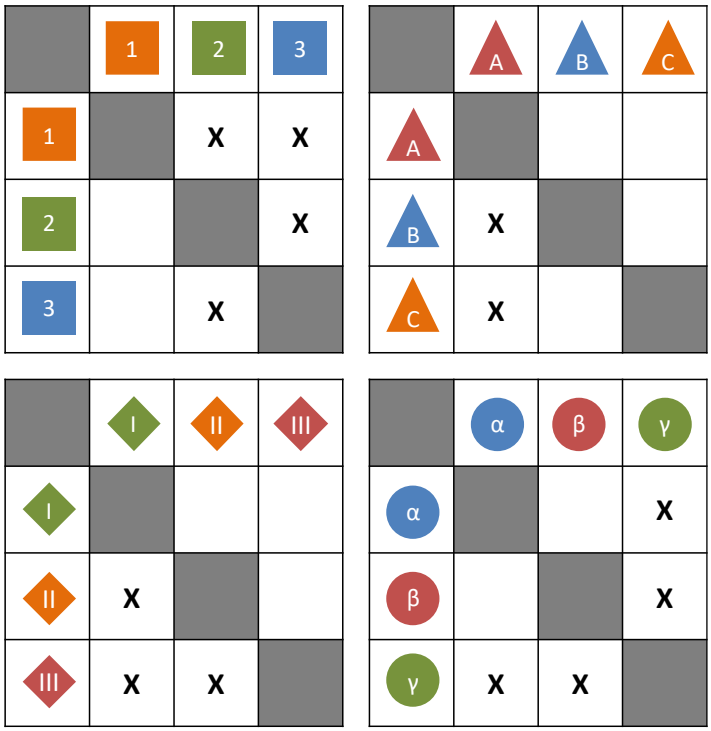


Abb. 25: Matrizendarstellung klassierter Elemente mit gerichteten Relationen ohne Mehrfachkanten

Werden gerichtete Relationen zusätzlich klassenübergreifend modelliert, wie in Abb. 26 dargestellt, bleibt die Übersichtlichkeit in der Graphendarstellung erhalten.

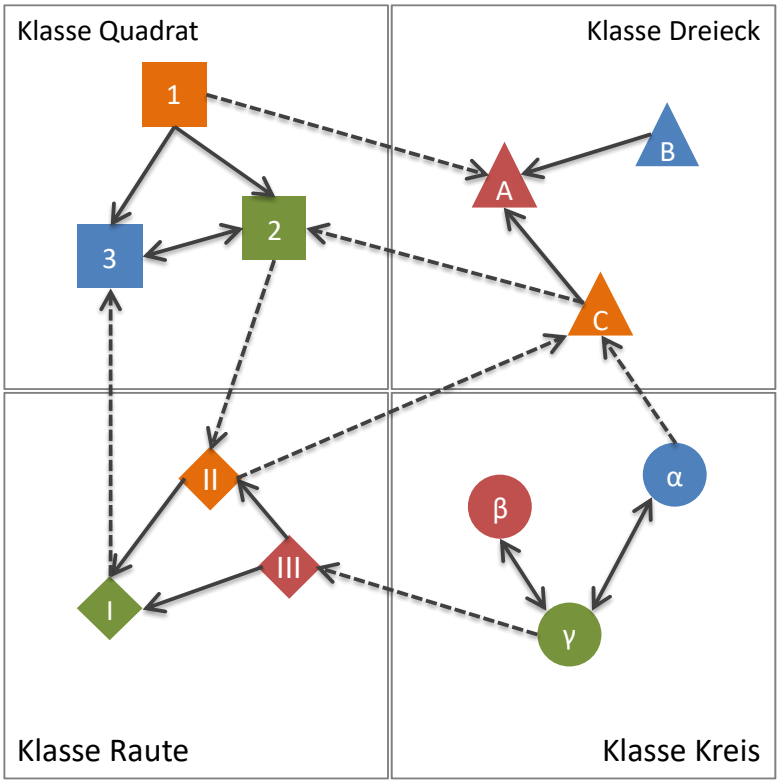


Abb. 26: Graphendarstellung klassierter Elemente mit gerichteten, klassenübergreifenden Relationen ohne Mehrfachkanten

In der Matrizendarstellung steigt die Anzahl benötigter Matrizen auf sechzehn, da neben den vier DSM nun auch zwölf DMM modelliert werden müssen, um auch die enthaltenen klassenübergreifenden Relationen abzubilden. Abb. 27 enthält die gleichen Informationen wie die

zugehörige Graphendarstellung. Die DSM, bei denen die Diagonale ausgegraut dargestellt ist, da sich Elemente nicht selbst beeinflussen können, sind in der linken Spalte dargestellt. Bei den DMM sind alle Relationen prinzipiell beschreibbar. Durch die Benennung der Einzelmatrizen, bei der die miteinander in Beziehung gesetzten Klassen als Kürzel jeweils oben links in der DSM bzw. DMM angegeben sind, können zwar alle Relationen rückverfolgt werden, die Transparenz ist jedoch kaum noch gegeben. Aus Gründen der vereinfachten Darstellung existieren nur sieben klassenübergreifende Relationen, weshalb in der Matrizendarstellung auch nur sieben DSM modellierte Relationen enthalten sind.

Abb. 27: Matrizendarstellung klassierter Elemente mit gerichteten, klassenübergreifenden Relationen ohne Mehrfachkanten

Obwohl die Matrizendarstellung im Vergleich zur Graphendarstellung deutlich umfangreicher wirkt, ist in beiden Fällen das gleiche System modelliert und die vorhandenen Informationen sind identisch. Dabei besteht für gerichtete Relationen zwischen Elementen unterschiedlicher Klasse die Notwendigkeit, die Matrizen regulär und mit getauschten Achsen darzustellen, während gerichtete Relationen innerhalb der gleichen Elementart über eine Matrix darstellbar sind, bei der beide Achsen identisch sind.

Die Ergänzung des beschriebenen Systemmodells um Mehrfachkanten würde sich in der Graphendarstellung durch mehrfache statt einfacher Relationsarten zwischen Elementen auswirken und ließe sich beispielsweise durch gestrichelte Linien visualisieren. Für die Matrizendarstellung würde dies in einer Potenzierung notwendiger Matrizen resultieren. Da für jede Relationsart eine eigene Matrix erforderlich wäre, ließe sich das Modell als n-dimensionale Matrix visualisieren, bei der jede Dimension eine Relationsart enthält. Dabei stellt jede Matrixebene auf der z-Achse eine geclusterte, gerichtete Matrixdarstellung einer Relationsart dar, während jede weitere Ebene der z-Achse eine andere Ausprägung des Attributs „Art“ einer Relation zwischen Elementen beschreibt. Für drei unterschiedliche Relationsarten würden somit bereits 48 Matrizen erforderlich sein, vgl. Abb. 28.

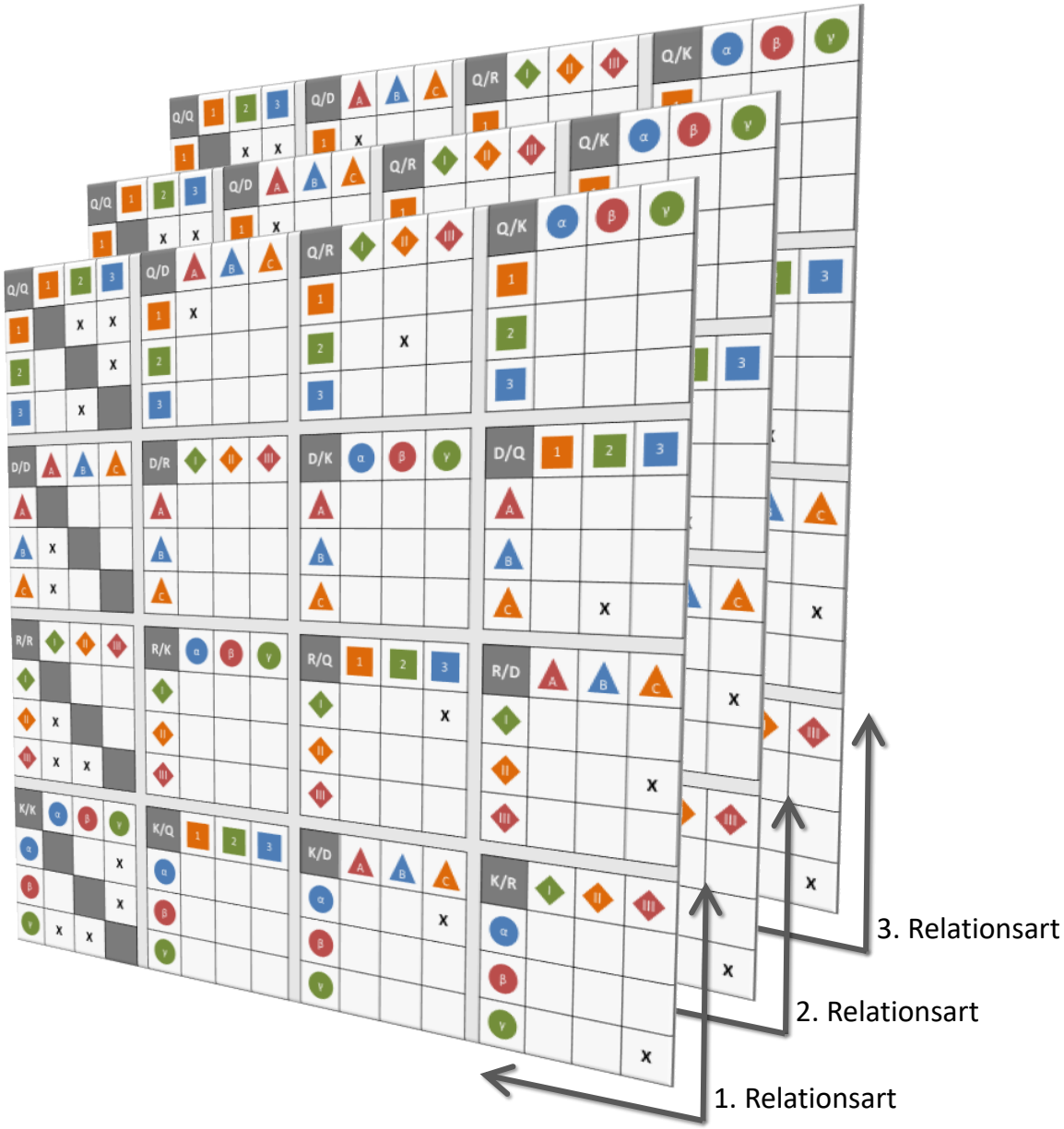


Abb. 28: Gerichtete und klassierte Matrizendarstellung mit Mehrfachkanten

Bei komplexen technischen Systemen bestehen oftmals genau solche Relationen und Klassen wie in Abb. 28 dargestellt. Die Notwendigkeit der Darstellung über gerichtete Mehrfachkanten sowie klassierte Elemente resultiert dabei aus der Vielzahl und Unterschiedlichkeit benötigter

Informationen. Um derartige Modelle nicht nur handhabbar, sondern auch transparent und reproduzierbar zu machen, ist neben einer geeigneten EDV-Umsetzung insbesondere die Festlegung eines einheitlichen Systemverständnisses, d.h. ein Denkmodell erforderlich. Dieser Aspekt wird in dem folgenden Kapitel behandelt.

3.3.2 Systemmodellierung nach DeCoDe

Während die zuvor dargestellten Grundlagen der Graphentheorie Allgemeingültigkeit besitzen, muss ein Systemmodell festgelegten Definitionen folgen, die sich auf ein konkretes Systemverständnis beziehen. Ein solches Systemmodell stellt eine Konvention im Umgang mit Systemen dar und kann im Aufbau und Inhalt von anderen Systemmodellen abweichen. Demnach können mehrere Systemmodelle unterschiedlicher Konventionen koexistieren, ohne im gegenseitigen Widerspruch zu stehen. In Entwicklungsprojekten ist es zur Erreichung der geforderten Eindeutigkeit und Transparenz jedoch erforderlich, Systeme gemäß einer einheitlichen Konvention zu modellieren, um reproduzierbare Ergebnisse zu generieren.

Für die Zielstellung der vorliegenden Arbeit wird im Wesentlichen den Konventionen des Demand Compliant Design (DeCoDe) gefolgt. DeCoDe legt Elemente und Relationen fest, welche innerhalb eines durch eine Systemgrenze aufgespannten Systemmodells existieren und betrachtet Systeme über die vier Sichten Anforderungen, Funktionen, Komponenten und Prozesse, die miteinander in Verbindung stehen [Mamrot et al. 2012, S. 22ff].

Obgleich das DeCoDe-Systemmodell den Fokus der anforderungsgerechten Produktentwicklung verfolgt, ist es im Kern nicht an einen konkreten Zweck gebunden. Ein zweckgebundener Ansatz baut daher auf zwei Bestandteilen auf, dem DeCoDe-Systemmodell und einem zweckgebundenen Vorgehenskonzept [Mamrot et al. 2012, S. 23]. Dies ermöglicht eine Kopplung verschiedener Ansätze, die sich aufgrund einer einheitlichen Systemmodellbasis beliebig kombinieren und erweitern lassen. Beispiele für die Kopplung von DeCoDe und einem Vorgehenskonzept bilden das Generic Systems Engineering [Winzer 2016], der Methodenworkflow zur Entwicklung mechatronischer Systeme [Winzer et al. 2009] oder der Ansatz zur modellbasierten Felddatenrückführung in die Produktentwicklung [Mamrot 2014].

Das DeCoDe-Systemmodell wird über die Sichten Anforderungen, Funktionen, Komponenten und Prozesse gebildet. Hierfür werden die Sichten über Matrizen zueinander in Beziehung gesetzt und die jeweils betrachtete Relation in einer Matrix dargestellt. Das Verständnis dieser Arbeit erweitert dabei das DeCoDe-Grundschema, bei dem eine Halbmatrix betrachtet wird [Winzer et al. 2007] und stellt die DeCoDe-Sichten als Vollmatrix dar. Dies ermöglicht insbesondere die Berücksichtigung gerichteter Relationen. Aus dem erweiterten DeCoDe-Grundschema ergeben sich 16 Matrizen, vgl. Abb. 29. Auf der von oben links nach unten rechts verlaufenden Diagonalen sind die Systemmatrizen S beschrieben, welche im Sinne einer DSM Relationen zwischen Elementen der gleichen Art abbilden. Alle anderen Systemmatrizen beschreiben gerichtete Relationen zwischen zwei unterschiedlichen Elementen und entsprechen damit einer DMM. Die Benennung als $S_{X,Y}$ zeigt die Richtung der beschriebenen Relation in der Systemmatrix von X nach Y an.

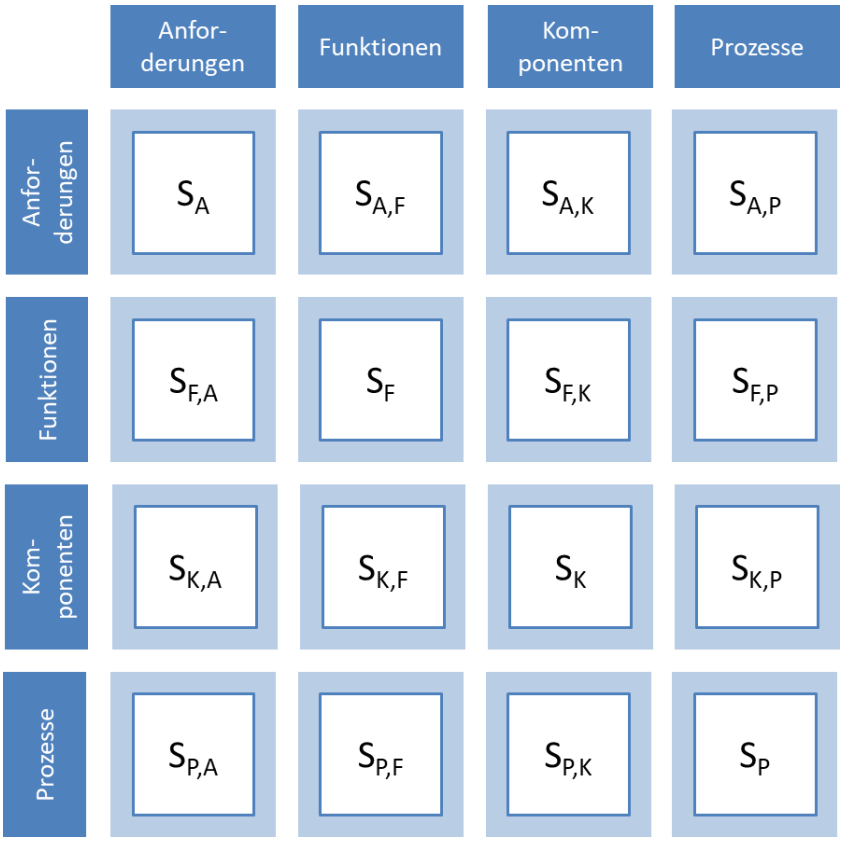


Abb. 29: Erweitertes DeCoDe-Grundschema[23]

Während das erweiterte DeCoDe-Grundschema den prinzipiellen Aufbau des DeCoDe-Modells widerspiegelt, enthält es noch keine Informationen über die enthaltenen Sichten. Diese sind ein wesentlicher Bestandteil der Konventionen von DeCoDe. Zur genaueren Beschreibung werden die Sichten nachfolgend definiert und in das Modell eingeordnet, wobei ein besonderer Fokus auf Funktionen liegt.

3.3.2.1 Anforderungssicht

Anforderungen stellen einen wesentlichen Bestandteil aller Produktentwicklungsansätze dar. Gemäß dem V-Modell bilden sie die Eingangsgröße der eigentlichen Entwicklungstätigkeit. Anforderungen sind ein Bestandteil der Vertragskonditionen und werden im technischen Kontext üblicherweise in Form von Zeichnungen, Lastenheften, Spezifikationen oder Vorschriften gestellt. Unter Berücksichtigung dieser Aspekte und in Anlehnung an die ISO 9001 definieren MAMROT ET AL. Anforderungen an technische Systeme als „Erfordernisse oder Erwartungen von Stakeholdern an ein System, welche festgelegt, üblicherweise vorausgesetzt oder verpflichtend sind“ [Mamrot et al. 2012, S. 24].

Um Anforderungen an technische Systeme strukturiert darzustellen und ihren gegenseitigen Einfluss zu beschreiben, wird in DeCoDe eine Anforderungsstruktur in Form einer DSM angelegt. Über die Systemmatrix S_A lassen sich entsprechend Informationen zu verschiedenen Relationsarten wie der Hierarchie darstellen. Dieser erste Schritt zur Schaffung einer Anforderungsbasis für die Produktentwicklung ist durchaus komplex, sodass eigene Ansätze wie beispielsweise das Requirements Engineering diesen Prozess unterstützen.

[23] in Anlehnung an [Winzer et al. 2007]

Neben dem Bezug zum System hebt die Definition den verpflichtenden Charakter von Anforderungen hervor. Entsprechend muss die Umsetzung in technische Merkmale eines Systems nachgewiesen werden, um die Anforderungserfüllung gegenüber den Stakeholdern sicherzustellen. Im V-Modell ist hierfür eine Eigenschaftsabsicherung in der Phase der Systemintegration vorgesehen. Insbesondere fokussiert aber auch das RGA-Modell den Abgleich von Systemmerkmalen und -eigenschaften mit definierten Anforderungen. Zudem muss der Einfluss auf die Anforderungserfüllung analysierbar sein, wenn es zu Änderungen während der Produktentwicklung oder der Serienfertigung kommt. Diesem Themenkomplex widmet sich das Engineering Change Management, welches idealerweise auch direkt ein Anforderungsmanagement umfasst. Auf diese Weise wird sichergestellt, dass der Einfluss von Änderungen auf Anforderungen bewertet werden kann.

Um den Nachweis der Anforderungserfüllung führen zu können und den Einfluss von Änderungen darzustellen, ist es zweckdienlich, den Zusammenhang zwischen Anforderungen und Elementen anderer Sichten im Modell darzustellen. Dies ermöglichen die DMM, welche aus Anforderungen und den mit ihnen verknüpften Elementen der Funktion-, Komponenten- und Prozesssicht generiert werden, d.h. $S_{A,F}$, $S_{A,K}$ und $S_{A,P}$ bzw. $S_{F,A}$, $S_{K,A}$ und $S_{P,A}$.

3.3.2.2 Funktionssicht

Funktionen besitzen als Bindeglied zwischen lösungsneutralen Anforderungen und lösungsgebundenen Komponenten und Prozessen eine zentrale Bedeutung für die Produktentwicklung. Um bei der Produktentwicklung zielgerichtet Anforderungen in Merkmale umzusetzen und gleichzeitig zu verhindern, dass die Lösung der Aufgabe vorzeitig festgelegt wird, nutzen Vorgehensmodelle wie das V-Modell [VDI 2206] oder das Pyramidenmodell der Produktkonkretisierung [Ponn/Lindemann 2011] Funktionen als Übergang zwischen den Anforderungen als Ausgangspunkt der Produktentwicklung und der physischen Realisierung mittels Komponenten und ihrer Umsetzung in Prozessen. Funktionen besitzen somit insbesondere für die frühen Phasen der Produktentwicklung eine besondere Bedeutung.

Ein notwendiger Bestandteil des Begriffsverständnisses ist der Zweck von Funktionen. Gemäß VDI 2221 stellen Funktionen „lösungsneutral beschriebene Beziehungen zwischen Eingangs-, Ausgangs- und Zustandsgrößen eines Systems“ [VDI 2221] dar, wobei kein expliziter Zweck zugrunde liegt. Hingegen beschreibt GAUSEMEIER eine Funktion als „der allgemeine und gewollte Zusammenhang zwischen Eingangs- und Ausgangsgrößen mit dem Ziel, eine Aufgabe zu erfüllen“ [Gausemeier 2010, S. 57f]. Nach EHRLENSPIEL beschreiben Funktionen „allgemein in den Natur- und Ingenieurwissenschaften die Darstellung eines physikalischen oder mathematischen Zusammenhangs“ [Ehrlenspiel 2003, S. 374]. Weitere Autoren definieren Funktionen als „das Vermögen, bestimmte Eingangsgrößen in bestimmte Ausgangsgrößen überzuführen“ [Patzak 1982, S. 64], „den gewollten Zusammenhang zwischen Eingang und Ausgang eines Systems mit dem Ziel, eine Aufgabe zu erfüllen“ [Pahl/Beitz 2007, S. 44] oder den „Zweck bzw. die Aufgabe, die ein System zu erfüllen hat“ [Mamrot et al. 2012, S. 24]. Die Definitionen verdeutlichen die Uneinigkeit über den Zweck bzw. das Ziel von Funktionen. Im Gegensatz zum Verständnis von EHRLENSPIEL, GAUSEMEIER und PATZAK enthält die VDI 2221 keine Aussage zum Zielbezug von Funktionen oder gewollten Zusammenhängen zwischen Eingangs- und Ausgangsgrößen.

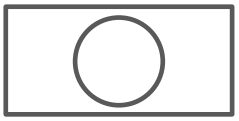
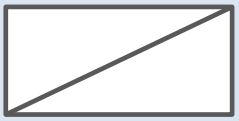
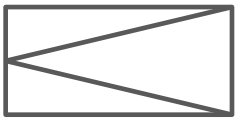
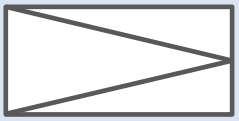
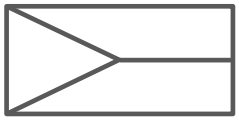
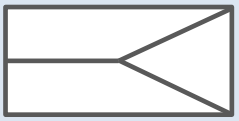
Unter Berücksichtigung der vorangegangenen Definitionen zum Zweck von Funktionen, der Bedeutung von Eingangs- und Ausgangsgrößen und der Lösungsneutralität werden Funktionen für die vorliegende Arbeit nachfolgend definiert:

> *Funktionen beschreiben den lösungsneutralen technisch-physikalischen Zusammenhang zwischen Eingangs- und Ausgangsgrößen technischer Systeme.*

In Kap. 4.2.1.5 wird dieses Funktionsverständnis in den Kontext der Produktentwicklung eingeordnet und aufgezeigt, welche Bedeutung die Lösungsneutralität bei der Nutzung von Lösungsebenen besitzt bzw. für welche Fälle sie eingeschränkte Gültigkeit besitzt.

Zur Schaffung eines einheitlichen Funktionsverständnisses wurden ursprünglich zunächst in der Konstruktionsmethodik Funktionskategorien definiert, mit denen Funktionen standardisiert beschreibbar sind. Diese Funktionskategorien basieren auf den Überlegungen zu Eingangs- und Ausgangsgrößen von Funktionen. Werden Funktionen als Umsetzung einer technischen Grundoperation verstanden, lassen sich für Energie-, Stoff- und Signalflüsse diese Grundoperationen auf wenige, grundlegende Funktionen reduzieren. In Anlehnung an EHRLENSPIEL und KOLLER/KASTRUP [Ehrlenspiel 2009, Koller/Kastrup 1998] werden daher die Funktionskategorien Leiten, Speichern/Isolieren, Wandeln, Vergrößern, Verkleinern, Vereinigen und Aufteilen genutzt, über die alle Zusammenhänge von Energie-, Stoff- und Signalflüssen beschreibbar sind, vgl. Tab. 15. Die Symbole veranschaulichen die Funktionskategorie und können für die Systemmodellierung mittels Symboldarstellung genutzt werden, siehe Kap. 4.2.1.4. Zudem wird in der Tabelle die Bedeutung der jeweiligen Funktionskategorie erläutert und Anmerkungen zur etwaigen Änderung der Größe (Wert und Einheit) sowie Beziehung zu anderen Funktionskategorien erläutert.

Tab. 15: Funktionskategorien und -symbole[24]

Funktions-kategorie	Symbol	Bedeutung	Anmerkung
Leiten		Die Eingangsgröße wird geleitet	Keine Änderung der physikalischen Größe. Antonym zu Speichern/Isolieren
Speichern/ Isolieren		Die Eingangsgröße wird isoliert oder gespeichert	Keine Änderung der physikalischen Größe Antonym zu Leiten
Wandeln		Die Eingangsgröße wird gewandelt	Änderung der physikalischen Größe
Vergrößern		Die Eingangsgröße wird vergrößert	Änderung des Werts, keine Änderung der Einheit Antonym zu Verkleinern
Verkleinern		Die Eingangsgröße wird verkleinert	Änderung des Werts, keine Änderung der Einheit Antonym zu Vergrößern
Vereinigen		Mehrere Eingangsgrößen werden vereinigt	Änderung des Werts, keine Änderung der Einheit Antonym zu Aufteilen
Aufteilen		Eine Eingangsgröße wird in mehrere Ausgangsgrößen aufgeteilt	Änderung des Werts, keine Änderung der Einheit Antonym zu Vereinigen

Beim Leiten wird ein Fluss ohne die Änderung der Größe[25] von einem Systembestandteil zu einem oder mehreren folgenden Systembestandteilen übergeben, wobei Leitungsverluste o.ä. keine Berücksichtigung finden. Auch der für den Hauptfluss insbesondere von Stoff- und Signalflüssen benötigte Energiefluss für das Leiten [Feldhusen/Grote 2013, S. 241f, Pahl et al. 2021, S. 13] wird nicht erfasst und muss bei Bedarf gesondert dargestellt werden.

Beim Speichern/Isolieren wird eine Eingangsgröße nicht unmittelbar bzw. nicht weitergeleitet. Während beim Speichern ein Fluss zwischen Systemelementen bis zur späteren Freigabe gestoppt wird, ist beim Isolieren der Fluss komplett unterbrochen, sodass aus der Eingangsgröße keine Ausgangsgröße resultiert. Eventuell auftretende Restflüsse werden beim Isolieren nicht berücksichtigt. Wird der Fluss jedoch nur reduziert und nicht isoliert, muss die Funktionskategorie Ändern genutzt werden, d.h. eine Reduzierung der Eingangsgröße.

[24] Eigene Darstellung in Anlehnung an [Riekhof et al. 2013b], nach [Ehrlenspiel 2009, Koller/Kastrup 1998]
[25] Eine physikalische Größe setzt sich zusammen aus Wert und Einheit, beispielsweise 100 Newton

Beim Wandeln wird eine Eingangsgröße hinsichtlich ihrer Größe gewandelt, dementsprechend ändern sich Wert und Einheit. Aufgrund von Wandlungsverlusten können beim Realisieren der Funktion Störausgangsgrößen auftreten, welche sich aus der prinzipiell-physikalischen Lösung einer Funktion ergeben. Wird beispielsweise in einem Verbrennungsmotor chemische in mechanische Energie umgewandelt, resultiert als Nebengröße thermische Energie, vgl. [Feldhusen/Grote 2013, S. 240, Pahl et al. 2021, S. 12], welche im Rahmen der vorliegenden Arbeit nachfolgend als Störgröße beschrieben wird.

Die Funktionskategorie Vergrößern beschreibt eine Änderung der Ausgangsgröße im Verhältnis zur Eingangsgröße. Hierbei wird der Wert vergrößert, die Einheit bleibt unverändert. Etwaige Änderungen des SI-Präfixes, beispielsweise von Kilowatt [kW] in Megawatt [MW], beziehen sich auf den Wert der physikalischen Größe und haben keinen Einfluss auf die Einheit.

Die Funktionskategorie Verkleinern stellt das Antonym zum Vergrößern dar. Auch hier erfolgt eine Änderung des Werts, allerdings in entgegengesetzter Richtung. Folglich wird der Wert der Ausgangs- im Verhältnis zur Eingangsgröße verkleinert, während die Einheit identisch bleibt.

Die Funktionskategorien Vereinigen und Aufteilen sind abhängig von der Richtung. Beim Vereinigen werden mindestens zwei Eingangsgrößen in eine neue Ausgangsgröße zusammengeführt, wobei auch mehrere Ausgangsgrößen resultieren können, solange die Anzahl der Eingangsgrößen die Anzahl der Ausgangsgrößen übersteigt. Beim Trennen werden genau entgegengesetzt zum Vereinigen aus einer oder mehreren Eingangsgrößen mehrere Ausgangsgrößen erzeugt, wobei die Anzahl der Eingangsgrößen kleiner als die der Ausgangsgrößen sein muss.

3.3.2.3 Komponentensicht

Als Komponenten werden physische Systembestandteile bezeichnet, wobei diese auch logische Bestandteile wie Software umfassen können [Mamrot et al. 2012, S. 24, Schlund/Winzer 2010, S. 282]. Gemäß dem DeCoDe-Systemverständnis dienen Komponenten der Realisierung technischer Funktionen, schränken jedoch im Gegensatz zu Funktionen den Lösungsraum zugunsten einer Produktkonkretisierung ein [Riekhof/Winzer 2011].

Komponenten stellen bei der schrittweisen Produktkonkretisierung ein lösungsgebundenes Systemelement dar. Während Funktionen lösungsneutral sind, ermöglichen Komponenten eine technisch spezifizierte Festlegung der Realisierung der Funktion.

3.3.2.4 Prozesssicht

Prozesse beschreiben „einen Satz zusammenhängender oder sich gegenseitig beeinflussender Tätigkeiten […]“ [DIN EN ISO 9000:2015, S. 33], bei dem Ressourcen zur „Umwandlung der Eingaben eines Systems in Ausgaben“ [Mamrot et al. 2012, S. 24] genutzt werden. Prozesse setzen dabei Funktionen des Systems unter Nutzung von Komponenten und anderen Ressourcen, wie Energie, um und sind insbesondere über die Abfolge der Prozessschritte beschreibbar. Dabei sind Prozesse keine Beitragsleister der Produktkonkretisierung und schränken demzufolge den Lösungsraum nicht ein. Stattdessen nutzen Prozesse ein gegebenes technisches System, um Eingaben unter Zuhilfenahme der Systembestandteile in Ausgaben zu wandeln.

Da Prozesse die Umwandlung von Eingangs- in Ausgangsgrößen durch die Nutzung bestehender technisch-logischer Systembestandteile beschreiben, ohne selbst die Umwandlung

vorzunehmen oder vorzugeben, resultieren aus Prozessen keine neuen Eingangs- oder Ausgangsgrößen. Für die Schnittstellenanalyse müssen Prozesse demnach nicht bezogen auf diese Größen berücksichtigt werden. Allerdings ist zu beachten, dass Prozesse bedingt durch ihren zeitlichen Charakter Funktionen dauerhaft oder mehrfach ausführen können, sodass aus der Funktion oder den realisierenden Komponenten resultierende Ausgangsgrößen hinsichtlich ihrer Beeinflussung von anderen Systembestandteilen durchaus kritisch sein können. Dies kann der Fall sein, wenn beim Betrieb eines Systems als Störausgangsgröße Wärme entsteht. Wird das System nur periodisch betrieben, könnte die resultierende Wärme beispielsweise über die Oberfläche abgeführt werden. Sofern über den Prozess das System jedoch permanent angesteuert wird, könnte durch die unzureichende Entwärmung ein Wärmestau resultieren, welcher negative Auswirkungen auf das System hat. Zur Vermeidung dieses Einflusses könnte somit die Integration eines Kühlkörpers oder eine aktive Kühlung notwendig sein.

Das Beispiel zeigt, dass die zeitlich-logische Abfolge von Prozessen auch bei der Systemauslegung berücksichtigt werden muss, um keinerlei negative Auswirkungen auf das System hervorzurufen. Die zu berücksichtigenden Eingangs- und Ausgangsgrößen resultieren jedoch nicht aus dem Prozess, sondern sind durch die schrittweise Produktkonkretisierung über Funktionen und Komponenten festgelegt worden. Prozesse beschreiben demnach den bei der Realisierung einer bestimmten Funktion oder der Nutzung einer Komponente erforderlichen Ablauf. Diese müssen jedoch nicht zu jedem Zeitpunkt parallel realisiert bzw. genutzt werden. Für diese Beschreibung wird auf die Prozesssicht zurückgegriffen. Der zeitlich-logische Ablauf wird jedoch nicht auf die Logik der Lösungsebenen angewendet, da Prozesse nicht sukzessiv je Lösungsebene umgesetzt werden, sondern im Gesamt- bzw. Teilsystem. Entsprechend wird die Prozesssicht für das genutzte Systemmodell nicht betrachtet.

3.4 Fazit zum Stand der Wissenschaft und Technik

Für die Entwicklung komplexer technischer Systeme besitzen frühe Entwicklungsphasen einen wesentlichen Einfluss auf die spätere Anforderungserfüllung des Produkts. Der guten Beeinflussbarkeit des Systems in diesen Phasen bei erkannten Problemen [Bertsche et al. 2009, Pfeifer 2001, von Regius 2006, Westkämper 2006] steht jedoch prinzipiell die geringe Informationsverfügbarkeit über das System gegenüber [Crostack/Klute 2008, Dorociak 2012]. Bestehende Ansätze fokussieren daher oftmals eine Informationsrückführung mittels Felddaten [Delonga 2007, Mamrot 2014] oder die Nutzung von Daten aus quantitativen und qualitativen Methoden zur bedarfsgerechten Systemauslegung und damit zur Erreichung geforderter Kenngrößen [Bertsche et al. 2009]. Aus Sicht des Komplexitätsmanagements ist zudem eine Systemmodellierung erforderlich, um Systemstrukturen zu visualisieren und ein Komplexitätsmanagement bzw. die Komplexitätsbeherrschung zu ermöglichen [Lindemann et al. 2009].

Einen Ansatz zur Optimierung von Entwicklungsprozessen aus der generischen Prozessperspektive bieten Fähigkeits- und Reifegradmodelle. Sie fordern von Unternehmen die Implementierung bestimmter Praktiken, welche die Umsetzung fähiger Prozesse sicherstellen und die Entwicklung reifer Produkte ermöglichen sollen. Aus diesen Vorgaben konnten Anforderungen abgeleitet werden, welche dem zu entwickelnden Vorgehenskonzept als Input dienen. Eine Beschreibung der Umsetzung enthält das analysierte CMMI-DEV nicht.

Bezogen auf die Produktperspektive geben Methoden der Reifegradabsicherung vor, welche Kriterien zu bestimmten Meilensteinen zu erfüllen sind. Hierüber soll der Reifegrad bereits während der Produktentwicklung über die Bewertung von Produkteigenschaften sichergestellt werden. Auch das RGA-Modell erlaubt die Ableitung von Anforderungen an den eigenen Ansatz, beschreibt jedoch kein implementierbares Vorgehen.

Ansätze aus dem Bereich der Zuverlässigkeitstechnik, d.h. qualitative und quantitative Methoden, sind prinzipiell geeignet, um Zuverlässigkeitskennwerte, Zuverlässigkeitsprognosen oder Ursache-Wirkungs-Zusammenhänge zu determinieren. Ihre Anwendung setzt das Vorhandensein von Informationen voraus, welche in den frühen Phasen der Produktentwicklung oftmals nicht oder nur eingeschränkt zur Verfügung stehen. Um eine Anwendung zu ermöglichen, muss die Informationsverfügbarkeit über das System somit verbessert werden.

Das Komplexitätsmanagement bildet in enger Verknüpfung mit den Ansätzen der Graphentheorie und der Systemmodellierung die Möglichkeit, technische Systeme strukturell zu beschreiben, hieraus Erkenntnisse für die Produktentwicklung abzuleiten und so die Informationsverfügbarkeit zu verbessern und die inhärente Komplexität beherrschbar zu machen.

Bezugnehmend auf die Anforderungen, die für ein Vorgehenskonzept definiert wurden (vgl. Tab. 2), können die diskutierten Ansätze dahingehend verglichen werden, welche Aspekte berücksichtigt sind oder für welche Aspekte eine Adaption für die Ansätze möglich ist. So ermöglicht das Fähigkeits- und Reifegradmodell CMMI-DEV eine Bewertung des Prozessreifegrads, kann jedoch nicht genutzt werden, um die Zuverlässigkeit technischer Systeme zu bewerten oder zu verbessern, systemrelevante Informationen darzustellen, Komplexität zu beherrschen, Systemstrukturen zu erfassen, Meilensteine zu definieren oder eine Schrittfolge abzuleiten, wie diese Tätigkeiten in Rahmen der Produktentwicklung zu implementieren sind. Im direkten Vergleich bietet das RGA-Modell des VDA eine Bewertung des Produktreifegrads anstelle des Prozessreifegrads sowie zusätzlich die Definition von Meilensteinen, an denen eine Bewertung erfolgen kann. Allerdings bietet auch das RGA-Modell keine Anhaltspunkte, wie Anforderungen umgesetzt werden können, welche die Erreichung einer hohen Produktreife unmittelbar unterstützen. Dies offenbart die methodische Lücke zwischen Ansätzen mit generischem Prozess- und konkretem Produktbezug.

Ansätze im Bereich des Komplexitätsmanagements und der Systemmodellierung, welche einen Beitrag zur Beherrschung der Herausforderungen (Informationsverfügbarkeit, Komplexität, Zuverlässigkeit) liefern, fokussieren die erforderliche Erfassung bzw. die Modellierung von Systemstrukturen, ihre Kategorisierung und die Vorgabe einer generischen Systemsicht zur Beschreibung technischer Systeme. Ihr Fokus liegt jedoch nicht bzw. nur anteilig auf einer Funktionsorientierung oder der Vorgabe einer Schrittfolge für die Produktentwicklung.

Tab. 16 stellt die Ansätze den Anforderungen gegenüber und zeigt, welche Aspekte bereits integraler Bestandteil der Ansätze sind (dargestellt mit einem schwarzen Kreis) bzw. bei welchen sich eine Erweiterung um diese Aspekte anbietet (dargestellt mit einem schwarz umrandeten Kreis), d.h. deren Umsetzung nicht im Ansatz selbst enthalten ist, sich aufgrund der inhaltlichen Nähe jedoch integrieren lässt. Hierbei wird jeweils die Referenz zur Anforderung angegeben (A01…A09, vgl. Tab. 2), auf welche sich bezogen wird.

Tab. 16: Umsetzung der Aspekte in den Ansätzen

Aspekt	Fähigkeits- und Reife-gradmodell	RGA-Modell	Komplexitäts-management	DeCoDe-System-modell
Erfassung von Systemstrukturen (A01, A02)			●	○
Modellierung von System-strukturen (A02)			○	●
Identifizierung von Störgrößen (A03)				○
Bereitstellung eines Systemmodells mit generischen Systemsichten (A04, A08)				●
Fokussierung auf Funktionen (A05)				○
Integration von Lösungsebenen (A06)				○
Definition von Meilensteinen (A06)		●		
Prozess-Reifegradbewertung	●			
Produkt-Reifegradbewertung (A06)		●		
Vorgabe einer Schrittfolge für die Pro-duktentwicklung (A07)				

Mit Ausnahme der Anforderung A09[26] werden alle Anforderungen durch mindestens einen Ansatz berücksichtigt. Da sich die Anforderung A09 auf das entwickelte Vorgehenskonzept selbst bezieht, kann diese nicht durch die Ansätze umgesetzt werden. In der umgekehrten Betrachtung zeigt sich, dass der Prozess-Reifegradbewertung keine abgeleitete Anforderung gegenübersteht. Die Prozess-Reifegradbewertung (konkreter Produktbezug) stellt jedoch das Pendant zur Produkt-Reifegradbewertung für den generischen Prozessbezug dar und wurde entsprechend berücksichtig. Entsprechend werden durch die Ansätze alle Anforderungen adressiert.

Es zeigt sich somit, dass die vorgestellten Ansätze alle Aspekte bis auf die Vorgabe einer Schrittfolge für die Produktentwicklung umfassen oder sie zumindest um diese erweitert werden können. Dies untermauert die These, dass die identifizierte methodische Lücke zwar die Integration der Ansätze erfordert, durch sie alleine jedoch nicht geschlossen werden kann. Im Folgenden wird daher ein Ansatz vorgestellt, wie die Lücke durch ein auf die frühen Entwicklungsphasen ausgerichtetes, funktionsorientiertes Vorgehenskonzept geschlossen werden kann. Hierfür werden Elemente bestehender Ansätze implementiert und bedarfsgerecht erweitert, etwa durch Erkenntnisse aus dem Komplexitätsmanagement, der Graphentheorie und der Systemmodellierung sowie durch Forderungen des CMMI-DEV und des RGA-Modells. Der Ansatz soll dabei nicht in Konkurrenz zu den besprochenen Ansätzen stehen, sondern ihre Bestandteile synergetisch vereinen.

[26] „Best Practices des Reifegradmodells sind herauszuarbeiten und im Vorgehenskonzept umsetzen“

4 Entwicklung eines Vorgehenskonzepts zur modellbasierten Systemkonkretisierung und -analyse in den frühen Phasen der Produktentwicklung

Zur Schließung der methodischen Lücke zwischen Fähigkeits- und Reifegradmodellen sowie Methoden der Reifegradabsicherung wird ein Vorgehenskonzept benötigt, welches unter den Rahmenbedingungen der frühen Phasen der Produktentwicklung einen Ansatz zur Beherrschung der skizzierten Herausforderungen bietet. Diese bestehen in der geringen Informationsverfügbarkeit, einer nicht beherrschten Komplexität und hieraus resultierend einer unzureichenden Zuverlässigkeit und Anforderungserfüllung des Systems. Die frühen Phasen der Produktentwicklung verschärfen den Informationsmangel, da hier üblicherweise keine oder nur wenige Informationen aus entwicklungsbegleitenden Versuchen oder sogar Felddaten vorliegen. Gleichzeitig bieten die frühen Entwicklungsphasen die beste Beeinflussbarkeit des Systems. Wird in diesen Phasen ein Ansatz implementiert, der die Informationsverfügbarkeit verbessert, die Integrierbarkeit von Methoden der Zuverlässigkeitstechnik bzw. Simulationen ermöglicht und sicherstellt, dass die Komplexität erfasst und beherrscht wird, können die beschriebenen Herausforderungen über die frühen Phasen der Produktentwicklung hinaus beherrscht werden. Hierfür soll auf vorhandenen Vorarbeiten aufgebaut werden, um einerseits die Adaptierbarkeit des Vorgehenskonzepts sicherzustellen und andererseits „Best Practices“-Ansätze zu implementieren, was einem der Kerngedanken des CMMI-DEV entspricht.

Zur Erläuterung der Rahmenbedingungen wird der Ansatz in Kap. 4.1 zunächst in den Produktentwicklungsprozess eingeordnet. Hierbei wird aufgezeigt, welche Informationen bereits aus vorhergehenden Entwicklungsschritten vorliegen und welche an nachfolgende Entwicklungsphasen übergeben werden. Kap. 4.2 stellt das entwickelte Vorgehenskonzept umfassend vor, welches aus drei miteinander interagierenden Bausteinen besteht: der Schrittfolge des Vorgehenskonzepts zur Systemkonkretisierung und -analyse (i), dem Systemmodell (ii) und der Schrittfolge zur Ermittlung des Produktreifegrads (iii).

Zur Erreichung eines besseren Verständnisses des Vorgehenskonzepts erfolgt die Vorstellung des Systemmodells vor der Schrittfolge zur Systemkonkretisierung und -analyse und der Schrittfolge zur Ermittlung des Produktreifegrads. Dies ändert jedoch nicht die generelle Abfolge im Vorgehenskonzept, bei der die Systemmodellierung den jeweiligen Stand der Systemkonkretisierung und -analyse abbildet und hierüber anschließend die Ermittlung des Produktreifegrads ermöglicht.

4.1 Einordnung in den Produktentwicklungsprozess

Zur Abgrenzung gegenüber anderen Ansätzen und zur Festlegung verfügbarer Informationen ist es erforderlich, das Vorgehenskonzept in die Phasen des Produktentwicklungsprozesses einzuordnen. Als Referenz dient der Systementwurf des V-Modells [VDI 2206, S. 29ff] bezüglich der Abgrenzung der frühen Phasen der Produktentwicklung. Hinsichtlich der Tätigkeiten stellt der Ansatz von PONN und LINDEMANN einen wichtigen Input bezüglich durchzuführender Schritte dar [Ponn/Lindemann 2011]. Demnach bilden Anforderungen den Startpunkt der Produktentwicklung. Diese sind zu spezifizieren und bilden die Grundlage für die im V-Modell vorgesehene Eigenschaftsabsicherung in der Phase der Systemintegration. Für das Vorgehenskonzept wird somit vorausgesetzt, dass Anforderungen vollständig erfasst und spezifiziert sind. Im Sinne des Systemmodells ist es zudem sinnvoll, dass alle Anforderungen als Elemente

modelliert und Abhängigkeiten zwischen diesen über Relationen beschrieben sind. Diese unter den Begriffen Anforderungsmanagement, Requirements Engineering und Requirements Management zusammenfassbaren Tätigkeiten wie Stakeholderanalyse, Anforderungserfassung, -strukturierung, -priorisierung, -spezifikation, -bewertung und -verwaltung bilden ein eigenes Themen- und Forschungsfeld mit spezifischen Modellen und Vorgehensweisen (vgl. u.a. [Danner/Lindemann 1996, Schienmann 2002, Mayer-Bachmann 2007]). Die Tätigkeiten sind daher kein Bestandteil des Vorgehenskonzepts. Dementsprechend wird als Input der Systementwurfsphase eine vollständige und widerspruchsfreie Anforderungsliste angenommen.

Den Anwendungsschwerpunkt des Vorgehenskonzepts bildet die Systementwurfsphase, wobei der Ansatz prinzipiell geeignet ist, um ebenfalls in der nachfolgenden Phase des domänenspezifischen Entwurfs angewandt zu werden. Hintergrund ist, dass die sukzessive Systemkonkretisierung über Lösungsebenen beliebig granular durchgeführt werden kann, sodass ein Systemmodell im Anschluss an die Entwurfsphase, d.h. in der Phase des domänenspezifischen Entwurfs, im Sinne der Produktkonkretisierung weiter beschrieben, analysiert und modelliert werden kann. Auch für die nachfolgende Systemintegration liefert das Vorgehenskonzept einen wichtigen Input. Der Abgleich mit den Anforderungen bzw. der Nachweis der Anforderungserfüllung in der Systementwurfsphase ist zwar kein Bestandteil des Vorgehenskonzepts, die Systemintegration wird jedoch durch die zuvor erfolgte Identifizierung und Analyse von Schnittstellen vorbereitet. Die entwicklungsbegleitende Bewertung des Produktreifegrads stellt zudem sicher, dass eventuelle Fehler nicht erst im Rahmen von Qualifizierungsprüfungen in der Phase der Systemintegration, sondern frühzeitig – bereits während des Systementwurfs – erkannt und behoben werden können.

In der Systementwurfsphase existieren Unterphasen, welche die schrittweise Konkretisierung des Systementwurfs beschreiben. Dies umfasst insbesondere Tätigkeiten zur Festlegung der funktionalen und gestalterischen Lösungsmöglichkeiten über Funktionen und Komponenten. Hierbei werden durch das Vorgehenskonzept Anforderungen umgesetzt, welche sich aus den Vorgaben zum generischen Prozessbezug der Referenzmodelle ergeben und somit die Erreichung eines prozessbezogenen Reifegrads sichergestellt.

Das vorgestellte Vorgehenskonzept endet mit der Übergabe der Informationen in Form des Systemmodells, der identifizierten und analysierten Schnittstellen sowie der Bewertung des produktbezogenen Reifegrads in die Phase des domänenspezifischen Entwurfs. Durch das Systemmodell werden alle Informationen über das System im Sinne der Komplexitätsbeherrschung in übersichtlicher Form bereitgestellt. Die identifizierten Schnittstellen ermöglichen den an der Produktentwicklung beteiligten Domänen, diese als Input aufzunehmen, auf Wechselbeziehungen zu prüfen und im Bedarfsfall zielgerichtete Analysen und Simulationen einzuleiten, um die Bewertung der Auswirkung auf ihr Subsystem zu ermöglichen. Der Reifegrad liefert zudem Aussagen über die prinzipielle Erfüllbarkeit der Anforderungen mit konkretem Produktbezug. Abb. 30 veranschaulicht die Einordnung in den PEP, den Input und Output des Vorgehenskonzepts sowie die Schnittstellen zwischen Vorgehenskonzept und Systementwurf.

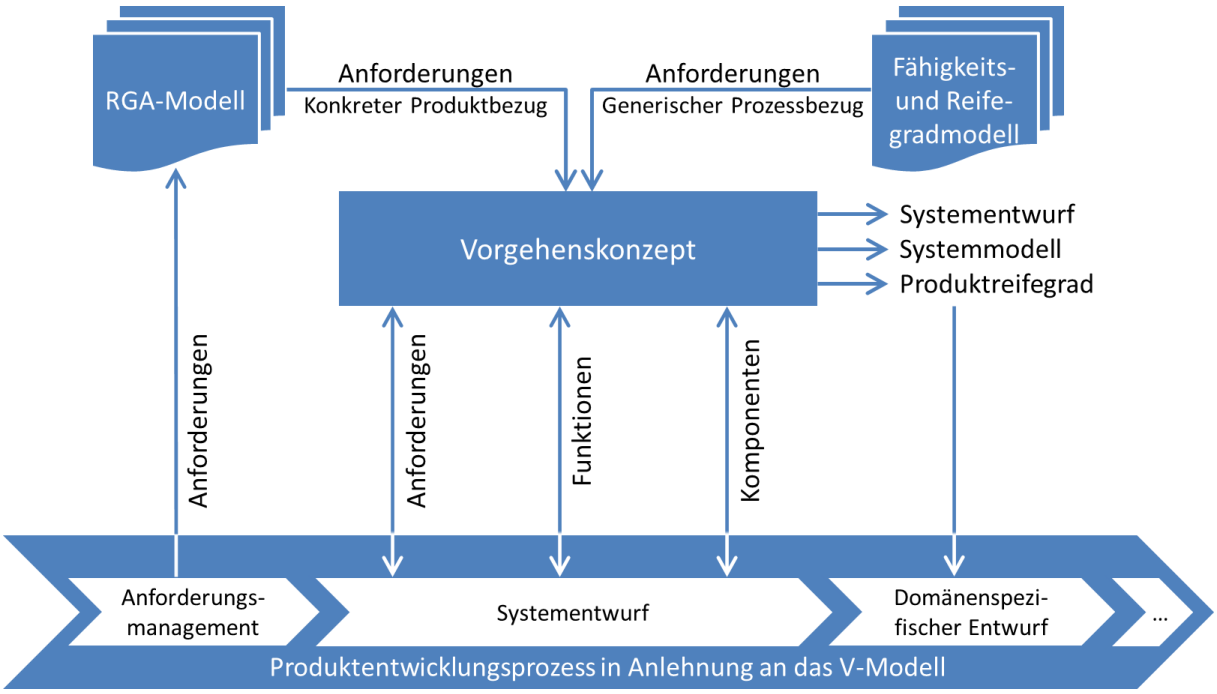


Abb. 30: Einordnung des Konzepts in die PEP-Phasen

Wie dargestellt, ist das Anforderungsmanagement im Produktentwicklungsprozess der Systementwurfsphase vorgeschaltet und liefert Anforderungen an das Produkt. Diese werden als Eingangsgröße für den Systementwurf genutzt und fließen parallel in das RGA-Modell ein. Hieraus werden anschließend Anforderungen mit konkretem Produktbezug an das Vorgehenskonzept abgeleitet. Sie dienen der entwicklungsbegleitenden Messung des Reifegrads an den zu definierenden Meilensteinen. Neben den Anforderungen mit konkretem Produktbezug fließen Anforderungen mit generischem Prozessbezug aus dem Fähigkeits- und Reifegradmodell in das Vorgehenskonzept ein, dargestellt auf der rechten Seite der Abbildung. Die Umsetzung dieser Anforderungen ist unabhängig von dem zu entwickelnden Produkt und bezieht sich auf das Vorgehenskonzept selbst. Auf diese Weise wird sichergestellt, dass durch das Vorgehenskonzept fähige Prozesse umgesetzt werden. Da diese Anforderungen keinen Bezug zu den PEP-Phasen besitzen, sind sie nicht, wie die Anforderungen der RGA, im zugrundeliegenden Produktentwicklungsprozess verankert, sondern werden unabhängig von diesem vorgegeben.

Das Vorgehenskonzept nimmt die beschriebenen Anforderungen auf und setzt diese in der Phase des Systementwurfs um. Über das Vorgehenskonzept werden dabei zunächst die funktionellen Lösungsmöglichkeiten (Funktionen) beschrieben, bevor über die gestalterischen Lösungsmöglichkeiten (Komponenten) der Lösungsraum zugunsten der Produktkonkretisierung eingeschränkt wird. Diese Beziehungen sind über die Doppelpfeile zwischen Vorgehenskonzept und der Phase Systementwurf dargestellt.

Durch das Vorgehensmodell werden die dargestellten Informationen aufgenommen und umgesetzt, sodass als Ergebnis der Systemkonkretisierung der Systementwurf (i), das Systemmodell mit den identifizierten und analysierten strukturellen Relationen und technisch-physikalischen Schnittstellen (ii) sowie der Produktreifegrad (iii) bereitgestellt werden können. Diese Informationen werden in die nachfolgende Phase des domänenspezifischen Entwurfs übergeben.

4.2 Aufbau des Vorgehenskonzepts

Das im vorherigen Kapitel in den PEP eingeordnete Vorgehenskonzept basiert auf drei zusammenhängenden Bausteinen: der Schrittfolge zur Systemkonkretisierung und -analyse, dem Systemmodell und einer meilensteinbezogenen Darstellung des Produktreifegrads, vgl. Abb. 31. Die links dargestellten Anforderungen des Fähigkeits- und Reifegradmodells CMMI-DEV bilden mit ihrem generischen Prozessbezug die Basis des Vorgehenskonzepts. Hieraus werden Anforderungen an die Systemkonkretisierung und -analyse (i), das Systemmodell (ii) und die Bewertung des Produktreifegrads (iii) abgeleitet, dargestellt über drei verknüpfte Bausteine. Die Schrittfolge zur Systemkonkretisierung und -analyse gibt die Tätigkeiten im Entwicklungsprozess vor. Gleichzeitig sind nach jedem Durchlauf der Schrittfolge durch Rauten symbolisierte Meilensteine für die Reifegradbewertung definiert, die in dem Produktentwicklungsprozess verankert sind. Um den Entwicklungsfortschritt darzustellen, wird ein Systemmodell genutzt, welches gleichzeitig die Basis für die Bewertung der Produktreife an den definierten Meilensteinen darstellt. Durch diese Verknüpfung bilden die Schrittfolge zur Konkretisierung des Lösungsraums bei der funktionsorientierten Produktentwicklung, die Systemmodellierung und die Schrittfolge zur Ermittlung des Produktreifegrads den Kernbestandteil des Vorgehenskonzepts. Dieses Vorgehenskonzept kombiniert und synchronisiert somit Fähigkeits- und Reifegradmodelle mit dem Produktreifegrad und schließt durch die Vorgabe einer Schrittfolge deren methodische Lücke.

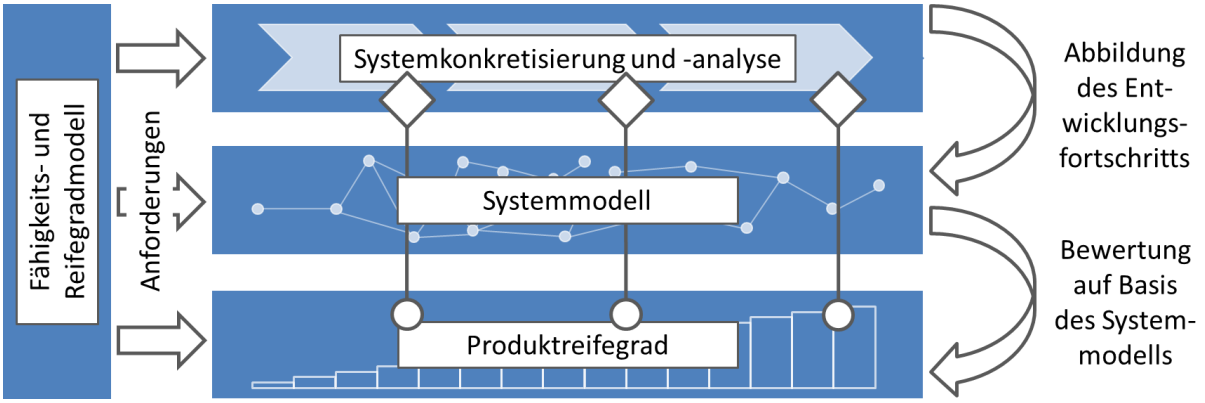


Abb. 31: Vorgehenskonzept

Während die aus dem Reifegrad- und Fähigkeitsmodell CMMI-DEV abgeleiteten Anforderungen mit generischem Produktbezug statisch sind, d.h. nur einmalig definiert und anschließend durch den Entwicklungsprozess umgesetzt werden müssen, ist der produktbezogene Teil des Vorgehenskonzepts dynamisch. Entsprechend müssen die aus dem RGA-Modell abgeleiteten Anforderungen mit konkretem Produktbezug für das jeweilige Produkt umgesetzt werden. Somit determiniert die grundlegende Schrittfolge, ob die Anforderungen mit generischen Prozessbezug erfüllt sind oder nicht, während die Erfüllung von Anforderungen mit konkreten Produktbezug für jedes Entwicklungsprojekt separat nachgewiesen werden muss. Die Umsetzung der Anforderungen mit generischem Prozessbezug aus dem CMMI-DEV ist integraler Bestandteil des Vorgehenskonzepts, durch das die Forderungen zur Umsetzung fähiger Prozesse sichergestellt werden.

Zur Vorstellung des Vorgehenskonzepts werden die drei Bausteine getrennt voneinander betrachtet. Da für die Schrittfolge zur Systemkonkretisierung und -analyse eine direkte Kopplung mit der Systemmodellierung vorgesehen ist, wird zum besseren Verständnis des Ansatzes

zunächst der Aufbau des Systemmodells erläutert (Kap. 4.2.1). Hierbei wird die Adaption des DeCoDe-Systemverständnisses betrachtet, bevor die Systemmodellierung unter Berücksichtigung des konkreten Systemverständnisses aufgezeigt wird. Dafür werden das Systemmodell und die zugehörigen Sichten detailliert erläutert und abgegrenzt. Zur Sicherstellung reproduzierbarer Modellierungsergebnisse werden die Systemelemente und die zwischen ihnen existierenden Relationen und Schnittstellen standardisiert. Der Fokus liegt hierbei auf der Funktionssicht, da Funktionen ein zentrales Element des Konzepts darstellen. Gleichzeitig wird die Lösungsneutralität und die Bildung von Lösungsebenen thematisiert, da die Schrittfolge je Lösungsebene durchlaufen wird.

Ziel der Schrittfolge zur Systemkonkretisierung und -analyse (Kap. 4.2.2) ist es, einerseits den in der Produktentwicklung erforderlichen Freiraum zur Sicherstellung innovativer Lösungen zu erhalten, andererseits zielgerichtet die Produktkonkretisierung bei gleichzeitiger Berücksichtigung der beschriebenen Herausforderungen der Produktentwicklung zu forcieren. Hierfür müssen insbesondere Relationen und Schnittstellen zwischen Systemelementen identifiziert und analysiert werden, um Wechselbeziehungen zu berücksichtigen. Zur Unterstützung der Relationsbestimmung und Schnittstellenidentifizierung und -analyse, die mit fortschreitender Produktkonkretisierung mehr Informationen darstellen muss, liegt dem Vorgehenskonzept ein Ansatz zur Nutzung von Lösungsebenen zugrunde. Diese ermöglichen ein schrittweises Vorgehen bei der Systemkonkretisierung und -analyse. Durch die parallele Abbildung des Entwicklungsfortschritts sowie der Relationen und Schnittstellen im Systemmodell wird die erforderliche Transparenz sichergestellt, was auch zur Komplexitätsbeherrschung beiträgt. Gleichzeitig wird über die Schrittfolge die Durchführung von Tätigkeiten abgesichert, welche für einen fähigen Prozess gemäß der Anforderungen mit generischem Prozessbezug erforderlich sind.

In Kap. 4.2.3 erfolgt die Erläuterung der Schrittfolge zur Bestimmung des Produktreifegrads. Dabei dient das Systemmodell an definierten Entwicklungsmeilensteinen, welche in der Schrittfolge vorgegeben sind, als Bewertungsgrundlage für die Reifegradbewertung. Während die funktionsorientierte Produktentwicklung und das verknüpfte Systemmodell einen konkreten Produktbezug besitzen und darüber Forderungen des RGA-Modells mit konkretem Produktbezug umsetzen, werden durch das gesamte Vorgehenskonzept Forderungen des CMMI-DEV hinsichtlich des generischen Prozessbezugs umgesetzt. Entsprechend stellt die Implementierung der Anforderungen mit generischem Prozessbezug den Brückenschlag zur Synchronisierung der Fähigkeits- und Reifegradmodelle mit Methoden der Reifegradabsicherung dar. Das Vorgehenskonzept schließt so die methodische Lücke zwischen generischem Prozess- und konkretem Produktbezug, indem ein Ansatz bereitgestellt wird, der nicht nur das *Was* berücksichtigt, sondern auch das *Wie* beantwortet.

4.2.1 Systemmodellierung

Das erweiterte DeCoDe-Grundschema mit den Sichten Anforderungen, Funktionen, Komponenten und Prozesse bildet eine geeignete Grundlage zur Modellierung technischer Systeme. Wie bereits dargestellt, sind für das Vorgehenskonzept Prozesse nicht von Bedeutung und werden nicht betrachtet. Das DeCoDe-Grundschema wird daher entsprechend angepasst und die resultierenden Systemmatrizen werden in Kap. 4.2.1.1 erläutert. Anschließend werden mögliche Beziehungen zwischen Systemelementen in Kap. 4.2.1.2 definiert, die sich in strukturelle Relationen (Kap. 4.2.1.2.1) und technisch-physikalische Schnittstellen (Kap. 4.2.1.2.2)

unterteilen. Kap. 4.2.1.3 zeigt für diese Beziehungen die Einordnung in das Systemmodell auf. Die getroffenen Festlegungen ermöglichen die eigentliche Systemmodellierung. Hierfür werden in Kap. 4.2.1.4 die unterschiedlichen Darstellungsarten des Systemmodells (Symbol-, Graphen- und Matrizendarstellung) aufgezeigt. Das Kapitel zur Systemmodellierung schließt mit der Erläuterung der für das Vorgehen wichtigen Festlegungen zur Lösungsneutralität und Lösungsebenen (Kap. 4.2.1.5) sowie zur Vererbung von Größen (Kap. 4.2.1.6) ab.

4.2.1.1 Systemmatrizen

Grundlage der Systemmodellierung der vorliegenden Arbeit bildet das erweiterte DeCoDe-Grundschema mit seinen 16 Systemmatrizen für Anforderungen, Funktionen, Komponenten und Prozesse, siehe Abb. 29. Dieses Schema wird prinzipiell übernommen, für die frühen Phasen der Produktentwicklung werden jedoch, wie in Abb. 30 dargestellt, nur die Systemsichten Anforderungen, Funktionen und Komponenten betrachtet. Hieraus ergeben sich neun Systemmatrizen, von denen drei Matrizen DSM (S_A, S_F, S_K) und sechs Matrizen DMM beschreiben ($S_{A,F}$, $S_{A,K}$, $S_{F,A}$, $S_{F,K}$, $S_{K,A}$ $S_{K,F}$), vgl. Abb. 32.

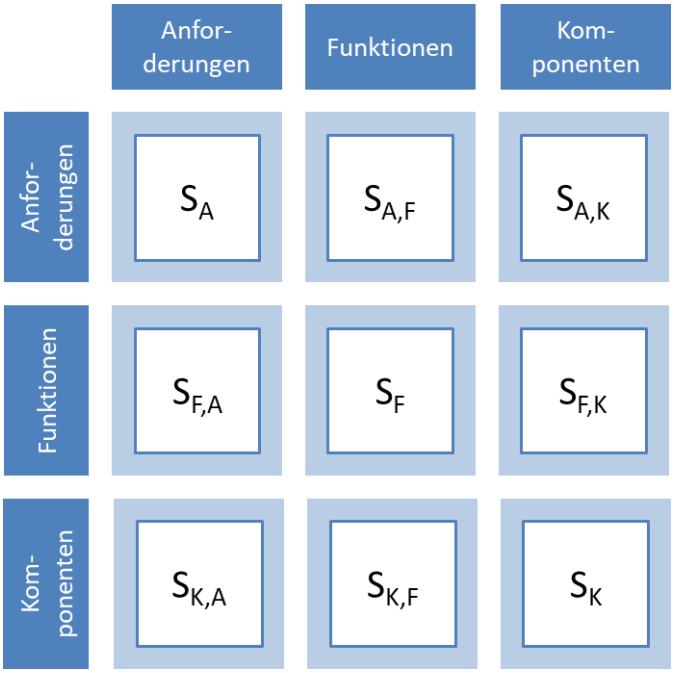


Abb. 32: Angepasstes DeCoDe-Grundschema für frühe Produktentwicklungsphasen

4.2.1.2 Beziehungen zwischen Systemelementen

Zwischen den Systemelementen bestehen Beziehungen, welche in den neun Systemmatrizen des angepassten DeCoDe-Grundschemas beschrieben werden. Dabei wird zwischen Beziehungen von Systemelementen unterschieden, die zur Beherrschung der Komplexität die Systemstruktur abbilden, und Beziehungen, die Wechselwirkungen zwischen Systemelementen aus technisch-physikalischer Sicht beschreiben. Diese unterschiedlichen Beziehungsarten werden als strukturelle Relationen bzw. technisch-physikalische Schnittstellen bezeichnet und nachfolgend erläutert. Um die Eindeutigkeit und Reproduzierbarkeit der Systemmodellierung zu gewährleisten, werden Relationen und Schnittstellen nachfolgend kategorisiert und standardisiert. Der Konvention der vorliegenden Arbeit folgend stellen sie die einzig zulässigen Beziehungen zwischen Systemelementen dar.

4.2.1.2.1 Strukturelle Relationen

Der im Systemmodell beschriebene, strukturelle Zusammenhang zwischen Systemelementen, sowohl innerhalb einer Systemsicht als auch zwischen Systemelementen unterschiedlicher Systemsichten, ist der Kern der Systemmodellierung und von zentraler Bedeutung für das Systemverständnis. Über die Beschreibung dieser sogenannten strukturellen Relationen zwischen Systemelementen werden Aussagen über das System ermöglicht, die einen wesentlichen Informationsgewinn für die Produktentwicklung darstellen und zur Beherrschung von Komplexität beitragen können, vgl. Kap. 2.2 zur Informationsverfügbarkeit. Aus diesem Grund ist es erforderlich, Relationen zwischen Systemelementen zu definieren, mittels derer Zusammenhänge für alle Systemsichten in identischer Weise beschreibbar sind. Strukturelle Relationen beschreiben logische, theoretische oder qualitative Beziehungen zwischen Systemelementen, welche der Abbildung der Systemstruktur dienen.

4.2.1.2.1.1 Hierarchie

Eine in der Konstruktionslehre oft verwendete Beziehung für Elemente gleicher Systemsicht ist die hierarchische Relation. Über sie erfolgt eine Zuordnung von Systemen und Subsystemen im Sinne einer Baumstruktur, beispielsweise bei einem Element, welches aus zwei Subelementen besteht und somit hierarchisch mit diesen verbunden ist. Hierarchische Relationen sind für DSM relevant, da sie die hierarchische Systemstruktur eindeutig definieren und darüber eine Abgrenzung des Systems zu anderen Systemen bzw. Subsystemen ermöglichen.

In der Graphendarstellung wird eine hierarchische Relation zwischen zwei oder mehreren Elementen über Pfeile dargestellt, die vom Element untergeordneter Hierarchieebene zum Element übergeordneter Hierarchieebene gehen. In der Matrizendarstellung wird eine hierarchische Relation zwischen Elementen in einer DSM über ein Kreuz in der Matrix dargestellt, wobei das Element untergeordneter Hierarchieebene in der Zeile steht und das Element übergeordneter Hierarchieebene in der Spalte. Dabei ist die Relation immer vom Element untergeordneter Hierarchieebene zum Element übergeordneter Hierarchieebene im Sinne einer „ist Bestandteil von“-Relation gerichtet und folgt der Logik „Zeile beeinflusst Spalte“.

4.2.1.2.1.2 Abfolge

Eine ebenfalls weit verbreitete Relationsart, insbesondere in der Prozesstechnik, ist die Abfolge von Elementen. Diese Relation definiert die Reihenfolge, in der Elemente zeitlich oder logisch vor anderen Elementen ablaufen, durchgeführt werden oder sich befinden. Eine Abfolgerelation existiert nur zwischen Elementen gleicher Systemsicht und gleicher Systemebene, da in untergeordneten Ebenen alle Elemente der übergeordneten Ebene übernommen werden müssen, sofern diese nicht durch eine Systemkonkretisierung granularer beschrieben werden. Während Abfolgerelationen für Funktionen sowie die im Rahmen dieser Arbeit nicht betrachtete Systemsicht der Prozesse von Bedeutung sind, sind sie für Komponenten nicht relevant. Hintergrund ist, dass über Abfolgerelationen logische, funktions- oder prozesstechnische Zusammenhänge zwischen den Elementen beschrieben werden. Bei Komponenten wäre dies nur möglich, solange eine Komponente nur eine Funktion realisiert. Werden hingegen mehrere, insbesondere nicht in direkter Abfolge zueinanderstehende Funktionen durch eine Komponente realisiert, ist eine Abfolgerelation für Komponenten nicht beschreibbar. Zwischen Komponenten können anstelle von Abfolgerelationen allerdings Energie-, Stoff- oder Signalflüsse über technisch-physikalische Schnittstellen beschrieben werden.

Zur Darstellung der Abfolge zeigt im Graphen vom ersten Element ein Pfeil auf die nachfolgenden Elemente. In der Matrizendarstellung wird das erste Element in der Zeile mit dem oder den folgenden Elementen in der Spalte verknüpft. Die Abfolgerelation ist immer vom vorhergehenden auf das nachfolgende Element gerichtet und bedeutet „folgt vor“.

4.2.1.2.1.3 Realisierung

Gemäß dem Systemverständnis existieren zwischen Anforderungen, Funktionen und Komponenten Relationen, welche die Realisierung von Anforderung durch Funktionen (oder unter bestimmten Bedingungen auch direkt durch Komponenten) und die Realisierung von Funktionen durch Komponenten beschreiben. Die Realisierung kann als Relation für Elemente unterschiedlicher Systemsichten genutzt werden. Elemente, welche über diese Relation mit den realisierten Elementen verbunden sind, besitzen somit automatisch einen Zweck.

Eine Realisierungsrelation kann nur zwischen DMM existieren, wobei Funktionen Anforderungen und Komponenten Funktionen realisieren. Einen Sonderfall stellt die Realisierung von Anforderungen durch Komponenten dar. Dieser Fall liegt vor, wenn Anforderungen explizit die Nutzung einer spezifischen Komponente fordern und so durch die Komponente selbst die Anforderung realisiert wird.

Eine ebenenübergreifende Verknüpfung mittels Realisierung ist nicht sinnvoll, da jede Ebene alle Elemente enthält und somit die Realisierungsrelation ohne Informationsverlust vollständig innerhalb einer Ebene beschreibbar ist.

In der Graphendarstellung ist der Pfeil auf das Element gerichtet, welches durch das verknüpfte Element realisiert wird, während in der Matrizendarstellung innerhalb der DMM das realisierende Element in der Zeile steht und mit dem realisierten Element in der Spalte verknüpft wird. Vom realisierenden Element geht damit eine gerichtete Relation auf das realisierte Element, welches die „realisiert“-Relation verdeutlicht.

4.2.1.2.1.4 Erfordernis

Eine Erfordernisrelation stellt einen Sonderfall dar, da sie nur in zwei Fälle genutzt wird. Im Gegensatz zu den zuvor aufgeführten Relationen ist sie nicht zwingend zu bilden, sondern nur, wenn entsprechende Konditionen vorliegen.

Systemelemente können aus zwei unterschiedlichen Gründen erforderlich sein. Einerseits ist es möglich, dass Anforderungen explizite Vorgaben hinsichtlich bestimmter Funktionen oder Komponenten machen. In diesem Fall werden diese explizit geforderten Systemelemente mit der Anforderung über eine Erfordernisrelation verknüpft. Dies stellt einen Sonderfall dar, da Anforderungen lösungsneutral formuliert sein sollten und regulär nicht über eine Erfordernisrelation mit den anforderungsrealisierenden Elementen zu verknüpfen sind. Andererseits kann bei der Realisierung von Funktionen durch Komponenten der Fall auftreten, dass neben den gewünschten Sollgrößen zusätzliche Störgrößen verursacht werden. Deren Verursachung erfordert die Ableitung zusätzlicher Anforderungen, sogenannter Design Constraints, welche den Umgang mit der Störgröße beschreiben. In diesem Fall werden die Elemente, welche das Design Constraint begründen, über eine Erfordernisrelation mit der Anforderung verbunden, welche den Umgang mit der Störgröße beschreibt. Eine Störgröße kann nur durch eine funktionsrealisierende Komponente verursacht werden. Die Notwendigkeit eines Design Constraints kann sich somit nur über eine Komponente ergeben, welche eine Störgröße verursacht. In

diesem Fall ist eine Erfordernisrelation zwischen Komponente und Anforderung zu modellieren. Entfällt die Störgröße, z.B. durch die Auswahl einer anderen technischen Lösung, entfällt ebenso die Notwendigkeit der Anforderung sowie ihrer anforderungsrealisierenden Elemente.

Hieraus resultiert, dass eine Erfordernisrelation nur zwischen Elementen unterschiedlicher Sicht und nur für die genannten Sonderfälle modelliert werden darf. Eine ebenenübergreifende Modellierung erfolgt nicht, da Störgrößen innerhalb jeder Ebene identifiziert und Optimierungsmaßnahmen implementiert werden, siehe hierzu Kap. 4.2.2.

Zur Darstellung dieses Zusammenhangs werden die Systemelemente über eine vom verursachenden Element zum notwendig gewordenen Element gerichtete „erfordert“-Relation verknüpft. In der Graphendarstellung zeigt der Pfeil daher auf das erforderliche Element und in der Matrizendarstellung wird das Kreuz zwischen das auslösende Element in der Zeile und das erforderliche Element in der Spalte gesetzt.

4.2.1.2.2 Technisch-physikalische Schnittstellen

Im Unterschied zu strukturellen Relationen, welche ausschließlich aufgrund logischer, theoretischer oder qualitativer Überlegungen zur Systemstruktur gebildet werden, beschreiben technisch-physikalische Schnittstellen messbare, quantitative Größen zwischen Systemelementen. Bei diesen Größen handelt es sich um Schnittstellen, wenn sie zwischen Systemelementen oder, in Sonderfällen, zwischen einem Systemelement und der Systemumwelt existieren. Wesentliche Merkmale stellen dabei die Richtung der Größe (Eingangs- oder Ausgangsgröße) sowie die Art dar, d.h. ihre grundlegende Unterteilung in Energie-, Stoff- und Signalflüsse.

4.2.1.2.2.1 Eingangs- und Ausgangsgrößen

Die Modellierung technischer Systeme dient der Beschreibung technisch-mathematischer bzw. technisch-physikalischer Zusammenhänge. Einen zentralen Betrachtungsgegenstand bilden Eingangsgrößen und Ausgangsgrößen, über die sich Systemelemente miteinander verbinden lassen. Die Betrachtung der Eingangs- und Ausgangsgrößen besitzt daher eine generelle Bedeutung für das zugrundeliegende Systemverständnis dieser Arbeit.

Im Bereich der statistischen Versuchsplanung (DoE – Design of Experiments) werden Eingangsgrößen unterteilt in gezielt änderbare Größen und nicht gezielt änderbare Größen, welche das Ergebnis, d.h. die Ausgangs- bzw. Zielgröße, beeinflussen können [Siebertz et al. 2010, S. 3ff], vgl. Abb. 33. Nicht gezielt veränderbare oder unbekannte Eingangsgrößen werden auch als Störgrößen bezeichnet, die als Pfeil dargestellt werden, welcher von unten auf ein betrachtetes System wirkt.

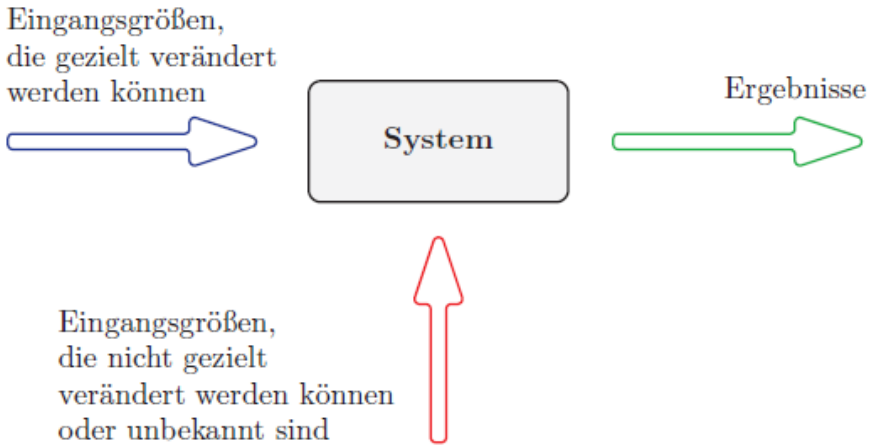


Abb. 33: Größen eines Systems [Siebertz et al. 2010, S. 3]

In Anlehnung an die beim DoE genutzte Differenzierung zwischen Eingangs-, Stör- und Zielgrößen wird in der vorliegenden Arbeit zwischen Eingangs- und Ausgangsgrößen unterschieden. In Abhängigkeit ihres Zielbezugs oder einem möglichen störenden Einfluss wird zudem zwischen Soll- und Störgrößen differenziert. Sollgrößen sind die zweckmäßigen und erforderlichen Größen eines Elements. Alle weiteren Größen, die auf ein Systemelement wirken oder aus diesen hervorgehen, sind somit entweder nicht zweckmäßig oder nicht erforderlich und werden demnach als Störgrößen bezeichnet.

Solleingangsgrößen beschreiben zweckmäßige und erforderliche Größen, die als Input in ein System oder ein Systemelement eingehen. Dies bedingt, dass sie gezielt veränder- und steuerbar sind. Im Gegensatz hierzu sind Störeingangsgrößen für den gewollten Zweck nicht erforderlich oder zweckmäßig und können potentiell negative Auswirkungen auf Systemelemente haben. Bei den Ausgangsgrößen erfolgt ebenfalls die Differenzierung zwischen Sollausgangs- und Störausgangsgröße. Sollausgangsgrößen stellen den zweckmäßigen Output eines Systemelements dar. Die Sollausgangsgröße ist entsprechend die Zielgröße, die den Zweck eines Systemelements begründet. Resultieren neben der Sollausgangsgröße weitere Größen, die sich nicht unmittelbar aus dem Zweck ergeben, werden diese als Störausgangsgröße bezeichnet. Störausgangsgrößen beschreiben einen Systemoutput, welcher parallel zur Sollausgangsgröße entstehen kann. Die Betrachtung der Störausgangsgrößen ist für die Schnittstellenanalyse von großer Bedeutung, da Störausgangsgrößen einen negativen Einfluss auf andere Systemelemente besitzen können, wo sie als Störeingangsgröße wirken. Allerdings können auch Sollausgangsgrößen eines Systembestandteils negative Auswirkungen auf andere Systembestandteile besitzen und würden für diese Systembestandteile daher als Störeingangsgröße betrachtet. Dementsprechend müssen Eingangs- und Ausgangsgrößen bezüglich ihres Einflusses auf andere Systemelemente analysiert werden.

Abb. 34 veranschaulicht diesen Ansatz. Hier sind die auf ein System wirkenden Eingangsgrößen und die resultierenden Ausgangsgrößen dargestellt. Grüne Kreise symbolisieren dabei gewünschte Solleingangs- und Sollausgangsgrößen, während rote Kreise unerwünschte Störeingangs- und Störausgangsgrößen symbolisieren.

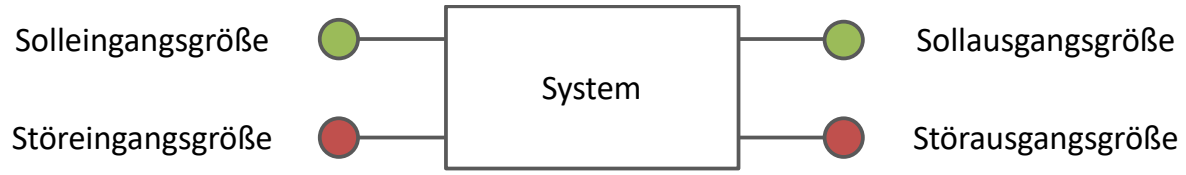


Abb. 34: Eingangs- und Ausgangsgrößen eines Systems

Die Beschreibung von Eingangs- und Ausgangsgrößen zwischen Elementen ist im angepassten DeCoDe-Grundschema nur für Funktionen und Komponenten möglich, da sie im Fall von Funktionen die theoretische Systembeschreibung und im Fall von Komponenten die technische Umsetzung hiervon darstellen. Für Anforderungen existieren demnach keine Eingangs- oder Ausgangsgrößen. Da Funktionen den gewollten Zustand beschreiben, existieren für Funktionen nur Solleingangs- und Sollausgangsgrößen, während für Komponenten zusätzlich Störeingangs- und Störausgangsgrößen existieren können.

4.2.1.2.2.2 Energie-, Stoff- und Signalflüsse

Die dargestellten Eingangs- und Ausgangsgrößen dienen der Beschreibung des technisch-physikalischen Zusammenhangs der Systemelemente. Während die Differenzierung in Soll- und Störgröße eine Aussage darüber liefert, ob Größen einen zweckmäßigen oder einen störenden Einfluss besitzen, ist hierüber noch keine Unterscheidung über die Art des technisch-physikalischen Zusammenhangs möglich. Aus diesem Grund wird eine Differenzierung genutzt, welche technisch-physikalische Schnittstellen als Energiefluss, Stofffluss oder Signalfluss definieren. Ein Fluss besteht, wenn die Ausgangsgröße eines Elements einem weiteren Element als Eingangsgröße dient. Dabei ist es unerheblich, ob sich diese Größe positiv oder negativ auf ein System oder Systemelement auswirkt.

Die Kategorisierung von Flüssen ermöglicht eine standardisierte Beschreibung der technisch-physikalischen Schnittstellen zwischen Systemelementen und stellt die Eindeutigkeit beschriebener Zusammenhänge im Systemmodell sicher. In Kombination mit der Differenzierung in Soll- und Störgrößen wird somit eine präzise Beschreibung im Systemmodell ermöglicht.

4.2.1.3 Strukturelle Relationen und technisch-physikalische Schnittstellen im Systemmodell

Für die Beschreibung der Relationen in den Systemmatrizen stehen die beschriebenen strukturellen Relationen (Hierarchie, Abfolge, Realisierung, Erfordernis) sowie technisch-physikalische Schnittstellen (Energiefluss, Stofffluss, Signalfluss) zur Verfügung. Dabei ist nicht jede Relation oder Schnittstelle für alle Systemmatrizen bzw. -sichten relevant.

Wie bereits erläutert, existieren hierarchische Relationen nur innerhalb von DSM, d.h. innerhalb einer Systemsicht für die Matrizen S_A, S_F und S_K. Für Anforderungen werden hierarchische Relationen nur bei Bedarf genutzt, um die Hierarchie einer Anforderung aufzuzeigen, welche sich aus der Notwendigkeit zum Umgang mit Störgrößen herleitet. In diesem Fall ist die abgeleitete Anforderung niedriger priorisiert als die ursprüngliche Anforderung. Die generelle Darstellung der Anforderungspriorisierung ist dem Anforderungsmanagement zugeordnet und somit nicht Bestandteil der Arbeit. Bei der Nutzung hierarchischer Relationen ist zu berücksichtigen, dass Hierarchien von Funktionen und Komponenten im Sinne einer „Bestandteil von“-Relation nur ebenenübergreifend genutzt werden dürfen, während Hierarchien von Anforderungen im Sinne der Priorisierung üblicherweise – aber nicht ausschließlich – innerhalb einer

Lösungsebene auftreten. Eine ebenenübergreifende Hierarchierelation ist generell nur zwischen Elementen direkt aufeinander aufbauender Lösungsebenen zulässig, also beispielsweise zwischen der Lösungsebene 1 und 2.

Abfolgerelationen bestehen ausschließlich innerhalb von DSM. Da Anforderungen nie und Komponenten nur indirekt über Energie-, Stoff- oder Signalflüsse in einem ablauforientierten Zusammenhang stehen können, existieren Abfolgerelationen nur für Funktionen. Sie können in der Matrix S_F modelliert werden.

Die Realisierungsrelation kann nur für DMM beschrieben werden, da Elemente nicht durch sich selbst realisierbar sind. Entsprechend kann eine Anforderung nicht durch eine andere Anforderung realisiert werden, sondern nur durch Funktionen oder Komponenten. Funktionen werden wiederum ausschließlich durch Komponenten realisiert, weshalb die Realisierungsrelation nur für die Matrizen $S_{F,A}$, $S_{K,A}$ und $S_{K,F}$ besteht. Hierbei stellt die Realisierung einer Anforderung über Funktionen (Matrix $S_{F,A}$) und die Realisierung einer Funktion über Komponenten (Matrix $S_{K,F}$) den regulären Lösungsweg der funktionsorientierten Produktentwicklung dar. Eine direkte Anforderungsrealisierung über eine Komponente (Matrix $S_{K,A}$) ist möglich, betrifft jedoch vorrangig nicht-funktionale Anforderungen.

Die Erfordernisrelation könnte fälschlicherweise als Gegenpart zur Realisierung interpretiert werden. Zur Vermeidung redundanter Informationen werden jedoch nur Relationen beschrieben, welche zur Sicherstellung eines eindeutigen Systemmodells erforderlich sind. Die Erfordernisrelation stellt jedoch nicht das inversdirektionale Pendant zur Realisierungsrelation dar, sondern wird als spezielle Relation im Falle einer expliziten Forderung gesetzt. So kann eine Anforderung explizit eine Funktion oder eine Komponente fordern, beispielsweise aufgrund einer direkten Kundenanforderung oder zur Erreichung bestimmter technischer Lösungen. In diesem Fall ist es zulässig, dass die Erfordernisrelation für DMM in den Matrizen $S_{A,F}$ oder $S_{A,K}$ gesetzt wird. Vorrangig wird die Erfordernisrelation jedoch verwendet, wenn Störgrößen durch Komponenten verursacht werden und dies die Definition abgeleiteter Anforderungen, sog. Design Constraints, erfordert. Dies erfolgt über die Matrix $S_{K,A}$. Entfällt das Systemelement, welches die Störgröße verursacht oder kann nachgewiesen werden, dass die Störgröße keinen negativen Einfluss auf das System besitzt und ignoriert werden kann, entfällt auch die Relation zur abgeleiteten Anforderung und damit die Anforderung selbst.

Tab. 17 fasst zusammen, in welchen Matrizenarten (DSM oder DMM) Relationen modelliert werden dürfen und ob eine ebenenübergreifende Modellierung zulässig ist.

Tab. 17: Anwendungsbereich von strukturellen Relationen

Relation	Matrizenart	Zulässigkeit ebenenübergreifender Modellierung
Hierarchie	DSM	ja
Abfolge	DSM	nein
Realisierung	DMM	nein
Erfordernis	DMM	nein

Schnittstellen beschreiben im Gegensatz zu strukturellen Relationen technisch-physikalische Beziehungen (Energie-, Stoff- oder Signalflüsse) zwischen Systemelementen und sind auf

Systemelemente gleicher Sicht beschränkt (DSM). Flüsse können zwar innerhalb der Funktionssicht oder der Komponentensicht existieren, nicht jedoch zwischen den beiden Systemelementen Funktionen und Komponenten, da ein Fluss zwischen einem logisch-theoretischen Systemelement (Funktion) und einem physischen Systemelement (Komponente) nicht möglich ist. Analog zu Abfolgerelationen existieren Schnittstellen auch nicht innerhalb der Anforderungssicht, sondern nur innerhalb der Funktions- und Komponentensicht in den Matrizen S_F und S_K.

Abb. 35 zeigt alle modellierbaren Beziehungen zwischen den Systemelementen im angepassten DeCoDe-Grundschema auf. Eine Besonderheit stellt die Matrix $S_{F,K}$ dar. Für diese existieren weder strukturelle Relationen noch technisch-physikalische Schnittstellen, sodass hierfür keine gerichteten Beziehungen von Funktionen auf Komponenten modelliert werden.

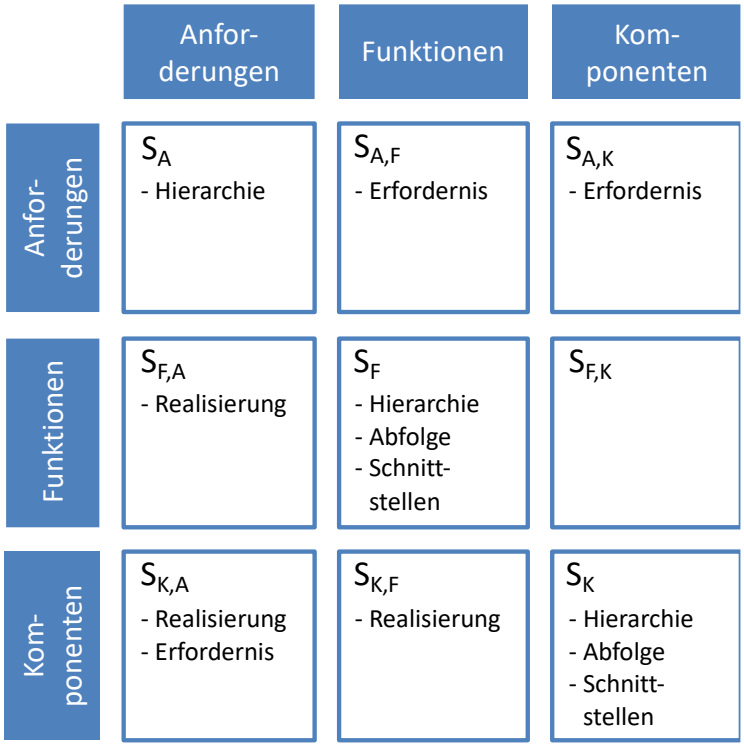


Abb. 35: Strukturelle Relationen und technisch-physikalische Schnittstellen im Systemmodell

Bei der Systemmodellierung wird zur Sicherstellung der Eindeutigkeit generell nur eine Relationsart je Matrix modelliert. Sofern mehrere Relationen modelliert werden müssen, beispielsweise für S_K oder $S_{F,A}$, resultieren Mehrfachkanten (vgl. Kap. 3.3.1 zur Graphentheorie), die jeweils in einer neuen Matrix modelliert werden müssen. Aus der Kombination der Systemmatrizen mit den vier strukturellen Relationsarten (Hierarchie, Abfolge, Realisierung, Erfordernis) und den drei technisch-physikalischen Schnittstellen (Energie-, Stoff-, Signalfluss) ergibt sich somit unter Ausnutzung aller Mehrfachkanten eine Maximalanzahl von 16 Matrizen, d.h. zehn Matrizen für strukturelle Relationen sowie zwei Matrizen mit jeweils drei technisch-physikalischen Schnittstellen. Dies stellt die theoretische Maximalanzahl möglicher Systemmatrizen dar und muss nicht für jedes Produktentwicklungsvorhaben zur Anwendung kommen. Dennoch erfordert die große Anzahl möglicher Matrizen eine Beschreibung der jeweils modellierten Relations- und Schnittstellenart. Zur eindeutigen Identifizierung wird die beschriebene Beziehung einer Matrix in ihrer Benennung über ein Kürzel indiziert angegeben, indem an das S_X der Systemmatrixbezeichnung nach einem Unterstrich der Anfangsbuchstabe der Relationsart angehängt wird. Hierbei kommen für strukturelle Relationen die Kürzel *H* (Hierarchie), *A* (Abfolge),

R (Realisierung) und *E* (Erfordernis) zur Anwendung und für technisch-physikalische Schnittstellen die Abkürzungen *StF* (Stofffluss), *EF* (Energiefluss) und *SF* (Signalfluss). So erfolgt beispielsweise die Modellierung einer Realisierungsrelation zwischen Funktionen und Anforderungen in der Matrix S_{F,A_R}, während der Energiefluss zwischen Komponenten in der Matrix S_{K_EF} beschrieben wird.

Um die Gesamtanzahl möglicher Matrizen nicht zusätzlich zu erhöhen, können innerhalb einer Systemmatrix Systemelemente unterschiedlicher Lösungsebenen modelliert werden. Der Vorteil dieser Festlegung ist, dass so beispielsweise hierarchische Beziehungen von Komponenten in nur einer Matrix beschreibbar sind. Dies verletzt nicht den Grundsatz, dass für bestimmte Elemente Beziehungen nur innerhalb der gleichen Lösungsebene modelliert werden dürfen, da lediglich eine Beschreibung der Beziehung in einer Matrix erfolgt, ohne jedoch die enthaltenen Elemente unterschiedlicher Lösungsebenen zu verknüpfen. Gleichzeitig erfordert diese Festlegung eine eindeutige Bezeichnung der Systemelemente, aus der ihre Zuordnung zur Systemebene ersichtlich ist. In Anlehnung an die Bezeichnung der Systemmatrizen werden Systemelemente nicht nur mit einem beschreibenden Namen benannt, sondern auch mit einem Kürzel des Anfangsbuchstabens für die Art des Systemelements (Anforderung *A*, Funktion *F*, Komponente *K*) und jeweils einem Index für die Ebene und die laufende Nummer. Letztere ist notwendig, wenn auf einer Ebene zwei gleich benannte Elemente existieren, was insbesondere für Komponenten häufig vorkommen kann, z.B. bei der Verwendung mehrerer baugleicher Teile. Als Beispiel erhalten zwei Zahnräder, die auf der dritten Ebene als Komponente modelliert werden, die Bezeichnungen „$K_{3.1}$ Zahnrad“ bzw. „$K_{3.2}$ Zahnrad“. Hierbei steht „K“ für das Systemelement Komponente, die erste Ziffer indiziert die Lösungsebene und die zweite Ziffer die laufende Nummer. Auf diese Weise wird eine Indizierung erreicht, welche eindeutig und nachverfolgbar ist. Ein weiterer Vorteil der Nutzung von Indexzahlen für Systemelemente innerhalb einer Matrix anstatt der Nutzung mehrerer Matrizen für jede Ebene ist, dass Relationen oder Schnittstellen zwischen Elementen unterschiedlicher Ebene direkt in einer Matrix beschrieben werden können, ohne dabei die Ebenenzuordnung zu verlieren. Dies reduziert die Gesamtanzahl notwendiger Matrizen deutlich.

4.2.1.4 Symbol-, Graphen- und Matrizendarstellung

Um alle Vorteile der Systemmodellierung nutzen zu können, erfolgt die Systemmodellierung neben der Graphen- und der Matrizendarstellung zusätzlich in der Symboldarstellung. Die Symboldarstellung ist eine Abart der Graphendarstellung, bei der die Systemelemente nicht über Knoten abgebildet, sondern mittels unterschiedlicher Symbole miteinander vernetzt werden. Anforderungen werden hierbei mittels einer Ellipse dargestellt, Funktionen als Rechteck mit dem jeweiligen Symbol der Funktionskategorie und Komponenten über ein Rechteck mit gerundeten Kanten. Abb. 36 zeigt die Symboldarstellung für die drei Systemsichten, wobei das symbolisierte Textfeld eine formulierte Anforderung darstellen soll und für die Funktion beispielhaft die Funktionskategorie „Wandeln“ dargestellt ist. Zur Bildung von Funktions- und Komponentennetzen sind neben dem Funktions- bzw. dem Komponentensymbol links und rechts Kreise abgebildet, welche die zugehörigen Eingangs- und Ausgangsgrößen symbolisieren. Je Eingangs- bzw. Ausgangsgröße wird ein Kreis dargestellt.

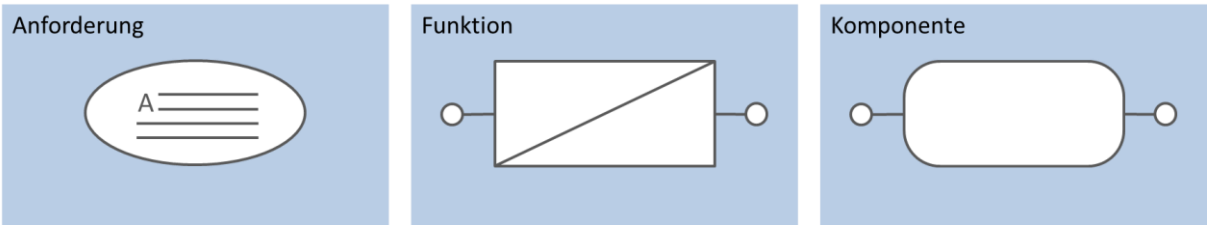


Abb. 36: Symboldarstellung von Anforderung, Funktion und Komponente

Die Symboldarstellung ergänzt die klassische Graphen- und Matrizendarstellung und unterstützt die Produktentwicklung durch eine übersichtliche und transparente Darstellung der Systemelemente und ihrer Relationen und Schnittstellen. Die Modellierung in der Graphen- und der Matrizendarstellung erfolgt über die Software LOOMEO der Fa. Teseon[27]. Für die Benennung der Matrizen und Elemente wird in LOOMEO die zuvor vorgestellte Logik genutzt. Bedingt durch die eingeschränkten Textformatierungsmöglichkeiten in LOOMEO entfällt jedoch für Systemmatrizen und Komponenten die Tiefstellung. Beispielsweise würde daher die Systemmatrix S_{F_H} zur Beschreibung der Hierarchie von Funktionen in LOOMEO als S.F_H und eine Komponente „Elektromotor“ der ersten Ebene und der dritten laufenden Nummer als „K1.3_Elektromotor“ bezeichnet werden. Die folgende Abb. 37 zeigt ein Beispiel für einen Energiefluss zwischen Komponenten der Systemmatrix S_{K_EF} in Graphen- und Matrizendarstellung in LOOMEO. Hierbei sind drei Komponenten der ersten Lösungsebene „K1.1_Energiespeicher“, „K1.2_Verkabelung“ und „K1.3_Elektromotor“ über einen Energiefluss miteinander verknüpft. In der links gezeigten Graphendarstellung sind die Komponenten über einen Pfeil miteinander verbunden. In der Matrizendarstellung auf der rechten Seite ist der Energiefluss über Kreuze in der Matrix dargestellt. Da alle Elemente unabhängig von ihrer Systemebene in den Systemsichten und zugehörigen Matrizen vorhanden sind, enthält die Matrizendarstellung auch die Komponente nullter Ebene „K0.1_System“, welche jedoch nicht über einen Energiefluss mit den Komponenten erster Ebene verbunden ist.

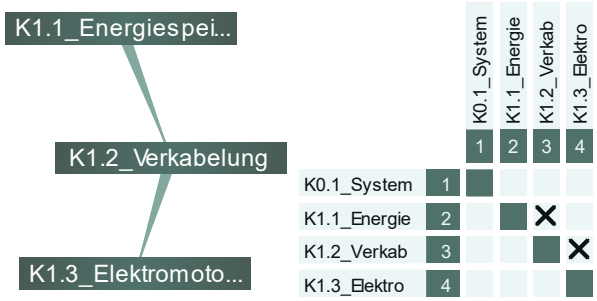


Abb. 37: Beispiel für die Graphen- und Matrizendarstellung der Matrix S_{K_EF}

4.2.1.5 Lösungsneutralität und Lösungsebenen

Für die funktionsorientierte Produktentwicklung ist die Lösungsneutralität von zentraler Bedeutung. Eine Beschreibung des Zusammenhangs zwischen einer oder mehrerer Eingangsgrößen und einer oder mehreren Ausgangsgrößen ist dann lösungsneutral, wenn sie Lösungsvarianten ihrer Umsetzung nicht vorwegnimmt und den Lösungsraum nicht einschränkt. Da eine sukzessive Einschränkung des Lösungsraums zugunsten der Systemkonkretisierung jedoch erforderlich ist, muss sich die Lösungsneutralität auf sogenannte Lösungsebenen beziehen. Der

[27] www.loomeo.com, www.teseon.com, zuletzt aufgerufen am 21.04.2022

größtmögliche Lösungsraum wird als Lösungsraum nullter Ebene bezeichnet und enthält keinerlei Einschränkungen durch Systemkonkretisierungen bis auf externe Vorgaben, beispielsweise den Zweck des Systems oder Bauraumvorgaben des Auftraggebers. Abb. 38 symbolisiert diesen Lösungsraum als dreidimensionales Objekt, welches weiter konkretisierte und damit eingeschränktere Lösungsräume erster und zweiter Ebene enthält. Der mittig dargestellte Lösungsraum erster Ebene ist hierbei verglichen mit dem übergeordneten Lösungsraum nullter Ebene lösungsgebunden und gegenüber dem untergeordneten Lösungsraum zweiter Ebene lösungsneutral, wobei die Lösungsräume untergeordneter Ebene, hier also der Lösungsraum erster und zweiter Ebene, immer eine Teilmenge des übergeordneten Lösungsraums bilden. Entsprechend ist es nicht zulässig, dass sich ein Lösungsraum untergeordneter Ebene in Teilen außerhalb des Lösungsraums einer übergeordneten Ebene befindet. In diesem Fall wäre eine unzulässige Konkretisierung gewählt worden. Entsprechend wäre zu prüfen, wie die gewählte Produktkonkretisierung so angepasst werden kann, dass sie den zulässigen Lösungsraum einhält oder der vorgegebene Lösungsraum muss hinterfragt und bei Bedarf erweitert werden.

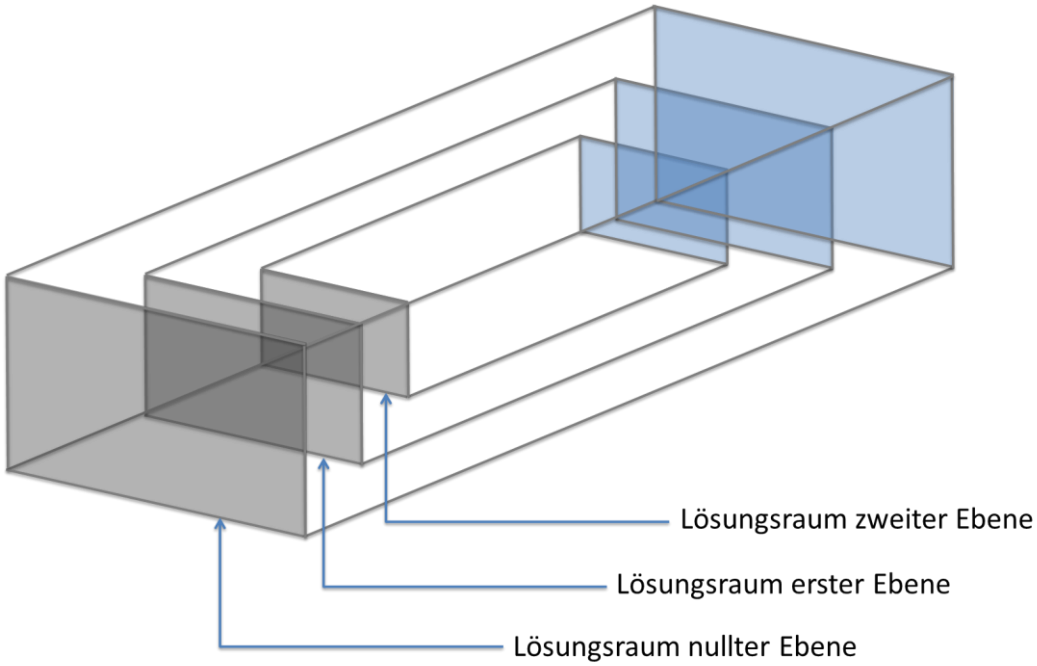


Abb. 38: Veranschaulichung des Lösungsraums

Im Laufe der Produktentwicklung muss der Systementwurf konkretisiert, d.h. sein Lösungsraum bewusst zugunsten einer bestimmten Lösung eingeschränkt werden. Mit jeder Einschränkung des Lösungsraums wird so eine neue Ebene *n* definiert, die bezogen auf die übergeordnete Ebene *n-1* lösungsgebunden ist und bezogen auf eine untergeordnete, nicht konkretisierte Ebene *n+1* lösungsneutral sein muss. Werden im Produktentwicklungsprozess Festlegungen zugunsten bestimmter Lösungen getroffen, steigt der Grad der Produktkonkretisierung an und die Lösungsneutralität sinkt analog hierzu.

4.2.1.6 Vererbung von Größen

Werden durch die nachfolgend beschriebenen Schrittfolgen Größen von Systemelementen identifiziert, müssen diese innerhalb der gleichen Ebene des Lösungsraums als auch auf untergeordnete Ebenen des Lösungsraums vererbt werden. Je Ebene des Lösungsraums wird

mindestens eine Funktion definiert, welche für die Ebene lösungsneutral ist und somit einer Beschreibung der funktionellen Lösungsmöglichkeit mit maximaler Abstraktion entspricht. Dies wird über die technische Beschreibung der Funktion mittels standardisierter Funktionskategorien ermöglicht, vgl. Kap. 3.3.2.2. Parallel werden die Eingangs- und Ausgangsgrößen der Funktion ermittelt und über festgelegte Kategorien beschrieben.

Sobald Funktionen mitsamt ihrer Eingangs- und Ausgangsgrößen beschrieben sind, müssen Komponenten als funktionsrealisierende Elemente definiert werden. Dabei sind die Eingangs- und Ausgangsgrößen der Funktion auf die Komponente zu vererben, vgl. Abb. 39. Die dargestellte Funktion als lösungsneutrale Umsetzung einer Anforderung besitzt jeweils eine Eingangs- und eine Ausgangsgröße. Von der Funktionssicht werden diese Größen auf die Komponentensicht vererbt, da diese eine Konkretisierung der Funktionssicht darstellt und somit mindestens über die gleichen Eingangs- und Ausgangsgrößen verfügen muss.

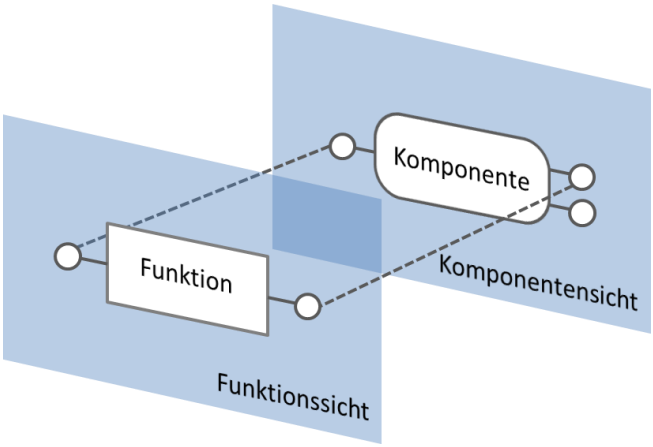


Abb. 39: Vererbung von Größen über Funktions- und Komponentensicht

Durch die Festlegung der Komponenten können auch neue Eingangs- und Ausgangsgrößen der Komponenten resultieren, da Komponenten lösungsgebunden sind und bedingt durch technisch-physikalische Eigenschaften Einschränkungen oder Ergänzungen der Eingangs- und Ausgangsgrößen erforderlich machen können. Es ist jedoch nicht möglich, dass bereits festgelegte Eingangs- oder Ausgangsgrößen von Funktionen durch die Auswahl einer Komponente geändert werden bzw. entfallen, solange nicht die eigentliche Funktion angepasst wird.

Im Beispiel ergibt sich durch die Auswahl der Komponente eine zusätzliche Ausgangsgröße, welche durch die technisch-physikalischen Eigenschaften der Komponente vorgegeben ist. Somit resultiert aus der Realisierung der Funktion durch die Komponente eine neue Ausgangsgröße. Da diese zwar aus technisch-physikalischen Gründen unvermeidbar, jedoch nicht für den Zweck der Funktion erforderlich ist, handelt es sich hierbei um eine Störausgangsgröße. Störausgangsgrößen, wie beispielsweise Abwärme, müssen nicht zwingend negative Auswirkungen auf das System besitzen, sondern können sogar, wie bei der Kraft-Wärme-Kopplung, synergetisch genutzt werden. Sie müssen jedoch identifiziert und ihre Auswirkung auf das System bewertet werden, um negative Einflüsse in Form von Störeingangsgrößen auf andere Systembestandteile auszuschließen oder dort entsprechend zu berücksichtigen.

Die Vererbung von Größen erfolgt nicht nur innerhalb der Ebene eines Lösungsraums, sondern auch ebenenübergreifend. Für Funktionen gilt zudem, dass sich definierte Funktionen auf jeder untergeordneten Ebene wiederfinden lassen müssen. Existiert beispielsweise auf der Ebene *n*

eine Wandlungsfunktion, muss diese auch auf der Ebene *n+1* sowie allen weiteren untergeordneten Ebene existieren. Dies ist in der sukzessiven Systemkonkretisierung über Lösungsebenen begründet, welche eine lösungsgebundene Konkretisierung der jeweils übergeordneten Ebene darstellt. Entsprechend darf eine Funktionsstruktur nicht bezüglich bereits existierender Elemente verändert werden, sondern darf diese nur erweitern bzw. konkretisieren. Auch die Eingangs- und Ausgangsgrößen müssen auf die jeweils untergeordnete Ebene vererbt werden. Hierbei ist eine Ergänzung durch neue Eingangs- und Ausgangsgrößen zulässig, wenn diese durch die funktionsrealisierenden Elemente vorgegeben werden.

Bei der Vererbung von Größen von Funktionen auf Komponenten innerhalb einer Lösungsebene oder bei der ebenenübergreifenden Vererbung von Größen innerhalb der Funktions- oder Komponentensicht ist zu beachten, dass die Zuordnung der Eingangs- bzw. Ausgangsgröße korrekt ist. Grundsätzlich existieren zwei Fälle: ein Element mit zu vererbenden Größen wird auf der untergeordneten Ebene nicht konkretisiert und damit direkt vererbt oder das Element wird auf untergeordneter Ebene über mindestens zwei Elemente konkretisiert. Im ersten Fall können Eingangs- und Ausgangsgrößen direkt übernommen werden. Beim zweiten Fall wird die Eingangsgröße nur auf das erste konkretisierte Element vererbt und die Ausgangsgröße auf das letzte konkretisierte Element.

Abb. 40 veranschaulicht beide Fälle. Auf der ersten Lösungsebene existieren drei Komponenten K_1.1, K_1.2 und K_1.3, die über nicht näher bestimmte Schnittstellen (Größen) miteinander verbunden sind. Auf der untergeordneten zweiten Ebene wird nur die Komponente K_1.2 durch Konkretisierung aufgeteilt und über K_2.2 und K_2.3 beschrieben. K_1.1 und K_1.3 werden mit ihren zugehörigen Größen direkt vererbt als K_2.1 bzw. K_2.4. Dieser Kontext lässt sich über eine hierarchische Relation zwischen den Komponenten erster und zweiter Ebene beschreiben. Während für K_1.1 und K_1.3 die Eingangs- und Ausgangsgrößen direkt vererbt werden (Fall 1), teilen sich diese für die konkretisierten Komponenten von K_1.2 (K_2.2 und K_2.3) auf (Fall 2). Zudem wird durch K_2.2 eine neue Ausgangsgröße erzeugt, welche den nachfolgenden Komponenten als zusätzliche Eingangsgröße dient.

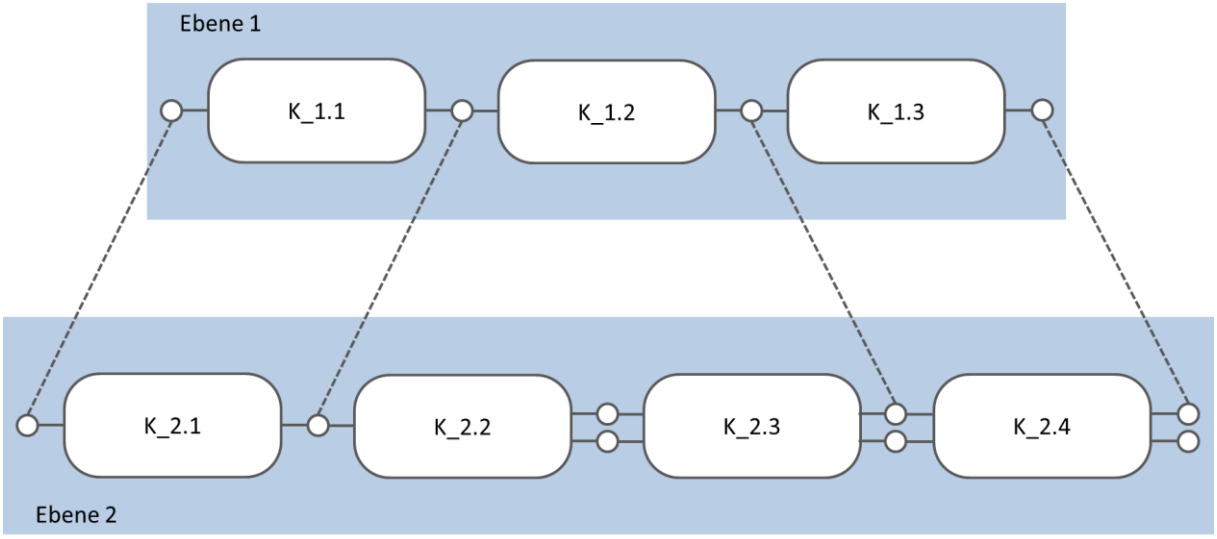


Abb. 40: Vererbung mit und ohne Konkretisierung auf untergeordneter Ebene

Die Eingangsgröße von K_1.2 stellt die vererbte Eingangsgröße von K_2.2 dar, da diese der ersten konkretisierten Komponente von K_1.2 auf der Ebene 2 entspricht. Die Ausgangsgröße

von K_1.2 findet sich jedoch erst als Ausgangsgröße der zweiten konkretisierenden Komponente K_2.3 wieder. Die dazwischen liegenden Größen, welche K_2.2 und K_2.3 als Schnittstellen verbinden, können identisch sein mit der Ausgangsgröße von K_1.2, müssen dies jedoch nicht. Bei der Vererbung von Größen muss daher beachtet werden, ob Elemente konkretisiert oder direkt übernommen wurden, um Größen auf die korrekten Elemente zu vererben.

4.2.2 Schrittfolge zur Systemkonkretisierung und -analyse

Die Schrittfolge zur Systemkonkretisierung und -analyse unterteilt sich in eine Schrittfolge zur Konkretisierung des Lösungsraums, welche den sequentiellen Aufbau des Systems über mehrere Lösungsebenen beschreibt, sowie eine Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse, welche Beziehungen zwischen Systemelementen erfasst und analysiert und parallel zu den Schritten der Systemkonkretisierung abläuft. Wie in Abb. 41 dargestellt, sind die Schrittfolgen eng verzahnt und laufen z.T. iterativ ab. Entsprechend muss die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse (dargestellt in den blauen Rechtecken) prinzipiell für jeden Schritt der Schrittfolge zur Konkretisierung des Lösungsraums (dargestellt als weiße Prozesssymbole) durchgeführt werden. Dies stellt sicher, dass bei jeder Festlegung zugunsten der Produktkonkretisierung alle potentiellen Relationen und Schnittstellen erfasst werden. In Abhängigkeit der betrachteten Lösungsebene und des Schrittes bei der Systemkonkretisierung sind nicht alle Schritte der Relationsbestimmung und Schnittstellenidentifizierung und -analyse relevant. Nicht relevante Schritte sind in der Darstellung daher ausgegraut.

Bei der Konkretisierung des Lösungsraums wird zwischen der Schrittfolge zur Konkretisierung des Lösungsraums nullter Ebene und der Schrittfolge aller weiteren *n*-ten Ebenen differenziert. So ist die Ableitung einer nullten Anforderung ausschließlich für die nullte Ebene relevant, während Definition abgeleiteter Anforderungen sowie die Implementierung von Optimierungsmaßnahmen nur für *n*-te Ebenen erfolgt. Die Prozessschritte sind entsprechend über graue Ecken und Rahmen für die Schrittfolge nullter Ebene bzw. gelbe Ecken und Rahmen für die *n*-ten Ebenen gekennzeichnet. Die Schritte „Funktion definieren“ und „Komponente festlegen“ existieren in beiden Schrittfolgen und sind demzufolge über graue und gelbe Ecken und einen grau-gelben Rahmen gekennzeichnet.

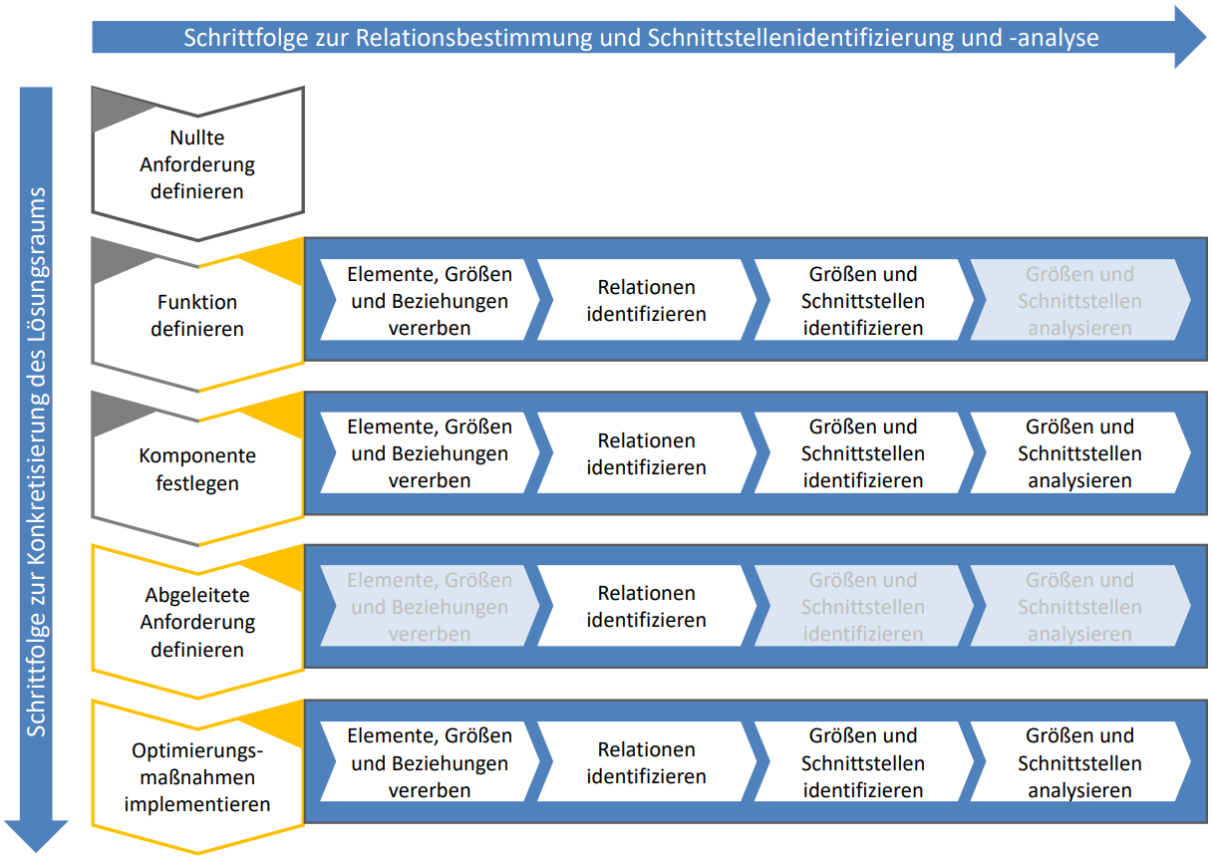


Abb. 41: Verzahnung zwischen den Schrittfolgen

Die Abbildung des Entwicklungsfortschritts erfolgt parallel zu den dargestellten Schritten über die Systemmodellierung. Die Schrittfolge und das Systemmodell interagieren auf diese Weise. Trotz der engen Verzahnung zwischen den Schrittfolgen werden diese nachfolgend separat beschrieben, um das Verständnis für die jeweiligen Schritte und ihre Abhängigkeit innerhalb der Schrittfolge sicherzustellen. In der Anwendung des Vorgehenskonzepts wird die Schrittfolge zur Konkretisierung des Lösungsraums für jede Ebene schrittweise durchlaufen. Für jeden Schritt erfolgt dann der Durchlauf der jeweils relevanten Schritte der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse.

Wie in der Abb. 41 dargestellt, laufen die Schritte der Schrittfolge zur Konkretisierung des Lösungsraums sequentiell ab, um die Reihenfolge der Systemkonkretisierung von Anforderungen über Funktionen zu Komponenten abzubilden. Mögliche identifizierbare Störgrößen, welche die Ableitung von Anforderungen und die Implementierung von Optimierungsmaßnahmen erfordern, können durch funktionsrealisierende Komponenten verursacht werden. Aus diesem Grund erfolgt nach der Festlegung der Komponenten bei der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse erstmalig der Schritt zur Analyse der Größen und Schnittstellen. Können Störgrößen identifiziert werden, muss eine Anforderung abgeleitet werden, welche den Umgang mit der Störgröße beschreibt. Sobald die abgeleitete Anforderung definiert ist und ihre Relationen festgelegt wurden, werden im nächsten Schritt Optimierungsmaßnahmen in Form von Funktionen und funktionsrealisierenden Komponenten festgelegt. Dafür werden prinzipiell die beiden Schritte zur Definition einer Funktion und zur Festlegung einer Komponente inklusive der verbundenen Schritte der Relationsbestimmung und Schnittstellenidentifizierung und -analyse durchlaufen. Da aus funktionsrealisierenden

Komponenten, welche als Reaktion auf Design Constraints definiert wurden, theoretisch neue Störgrößen resultieren können, muss der Schritt zur Analyse der Größen und Schnittstellen ebenfalls für die Optimierungsmaßnahmen durchgeführt werden, wobei dies nur die Komponenten- und nicht die Funktionssicht betrifft.

4.2.2.1 Schrittfolge zur Konkretisierung des Lösungsraums

Bedingt durch die grundlegenden Besonderheiten der nullten Ebene erfolgt die Beschreibung der Schrittfolge zur Konkretisierung des Lösungsraums für die nullte Ebene und alle weiteren n-*ten* Ebenen separat. Diese Trennung erfolgt zugunsten eines besseren Verständnisses der vorliegenden Arbeit und besitzt in der Praxis keine Relevanz, sodass die Schrittfolge in sich geschlossen durchlaufen werden kann.

4.2.2.1.1 Schrittfolge zur Konkretisierung des Lösungsraums nullter Ebene

Obwohl beim Vorgehenskonzept vom Vorhandensein einer vollständigen und widerspruchsfreien Anforderungsliste ausgegangen wird, sieht die Schrittfolge für die nullte Ebene die Definition einer Anforderung nullter Ebene vor. Da für nachfolgende Ebenen diese Anforderung nicht erneut abgeleitet wird, beginnt die Schrittfolge mit einem sogenannten nullten Schritt, welcher nur für den Lösungsraum nullter Ebene existiert. Zudem stellt das System nullter Ebene per Definition ein ideales System ohne Störgrößen dar, da diese erst auf untergeordneten Ebenen ermittelt und analysiert werden. Entsprechend werden für die nullte Ebene keine abgeleiteten Anforderungen definiert oder Optimierungsmaßnahmen implementiert, sodass die Schrittfolge lediglich die Definition der Anforderung, die Definition der Funktion sowie die Festlegung der Komponente nullter Ebene umfasst, welche üblicherweise das Gesamtsystem ist.

0. Schritt: Anforderung nullter Ebene definieren

Basierend auf den bereitgestellten Anforderungen aus vorgelagerten Aktivitäten muss zunächst die sogenannte Anforderung nullter Ebene abgeleitet werden. Diese Anforderung beschreibt zusammenfassend den generischen Zweck des technischen Systems. In Einzelfällen kann es erforderlich sein, mehrere Anforderungen nullter Ebene parallel aufzustellen. In diesem Fall ist jedoch zu prüfen, ob sich die parallelen Anforderungen nullter Ebene abstrakter durch eine einzige Anforderung nullter Ebene beschreiben lassen. Da die Anforderung nullter Ebene das erste zu erfassende Systemelement darstellt und keinerlei Eingangs- und Ausgangsgrößen oder Relationen besitzt, existieren keine zugeordneten Schritte der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse.

Die Anforderung nullter Ebene bildet die Basis der weiteren Schrittfolge und muss so formuliert sein, dass eine Funktion nullter Ebene gebildet werden kann, welche lösungsneutral die Anforderungserfüllung beschreibt. Das Prinzip der Lösungsneutralität drückt aus, dass bei der Realisierung einer Anforderung durch eine Funktion keine weitere Einschränkung des Lösungsraums erfolgt als durch die Anforderung bereits vorgegeben ist. Enthält eine Anforderung nullter Ebene aufgrund unzureichender Abstraktion der Anforderungsbasis unzulässige bzw. unnötige Einschränkungen, wird der Lösungsraum bereits an dieser Stelle ungewollt eingeschränkt.

Die nullte Anforderung dient in der funktionsorientierten Produktentwicklung der Fokussierung auf einen lösungsneutralen Systementwurf. Sie ersetzt jedoch nicht die Anforderungsbasis, die merkmalsbezogene Umsetzung und die Eigenschaftsabsicherung in der Phase der Systemintegration im Sinne des V-Modells.

1. Schritt: Funktion nullter Ebene definieren

Basierend auf der Anforderung nullter Ebene muss im ersten Schritt eine Funktion nullter Ebene definiert werden, welche die funktionelle Anforderungsumsetzung beschreibt und dabei lösungsneutral ist. Die Erfüllung einer Anforderung nullter Ebene soll nach Möglichkeit durch die Beschreibung einer einzigen Funktion nullter Ebene erfolgen. In Einzelfällen kann es erforderlich sein, dass mehrere Funktionen nullter Ebene zur Beschreibung der Anforderung nullter Ebene erforderlich sind. In diesem Fall ist zu prüfen, ob die Anforderung nullter Ebene soweit abstrahiert werden kann, dass die Erfüllung über nur eine Funktion nullter Ebene beschrieben werden kann. Für die Funktionsdefinition sind die vorgestellten Funktionskategorien zu nutzen.

Bevor die Komponente nullter Ebene festgelegt wird, ist gemäß des vorgestellten Vorgehenskonzepts die verzahnte Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse durchzuführen. Diese Schrittfolge ist in Kap. 4.2.2.2 beschrieben. Da keine übergeordnete Ebene existiert, können keine Elemente, Größen oder Beziehungen vererbt werden. Bei der Identifizierung der Relationen muss jedoch die Realisierungsrelation zwischen der Funktion und der Anforderung nullter Ebene erfasst werden. Dies stellt die einzige Relation für die Funktion nullter Ebene dar, sie muss jedoch immer existieren. Bei der Identifizierung von Größen und Schnittstellen sind die Eingangs- und Ausgangsgrößen der Funktion zu benennen. Da Funktionen den idealen Zustand beschreiben und damit per Definition keine Störgrößen für sie existieren, ist der Schritt zur Analyse der Größen und Schnittstellen nicht relevant.

2. Schritt: Komponente nullter Ebene festlegen

Nach der Definition der Funktion nullter Ebene muss eine für die Funktionsrealisierung geeignete Komponente festgelegt werden. Die Komponente nullter Ebene beschreibt das technische Gesamtsystem, weshalb idealerweise nur eine Komponente zur Realisierung der Funktion festzulegen ist. Sofern mehrere Komponenten festgelegt werden, ist zu überprüfen, ob sich das System über eine abstraktere Komponente beschreiben lässt. Die Festlegung einer oder mehrerer Komponenten stellt den höchsten Konkretisierungsgrad der Lösungsebene dar.

Bezogen auf die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse müssen die Eingangs- und Ausgangsgrößen der Funktion nullter Ebene auf die Komponente vererbt werden. Es können zudem weitere Eingangs- und Ausgangsgrößen existieren, welche zu berücksichtigen und zu modellieren sind. Für die nullte Ebene sollten aufgrund ihrer Definition als idealisiertes System mit größtmöglichem Lösungsraum keine Störgrößen existieren. Werden dennoch Störgrößen bei der Schnittstellenidentifizierung und -analyse festgestellt, ist zu überprüfen, ob das technische System in ausreichender Abstraktion beschrieben wurde.

4.2.2.1.2 Schrittfolge zur Konkretisierung des Lösungsraums *n*-ter Ebene

Während die Schrittfolge für den Lösungsraum der nullten Ebene nur einmalig durchlaufen wird, stellt die Schrittfolge der *n*-ten Ebene einen iterativ angewandten Prozess dar, welcher auf der jeweils übergeordneten Ebene aufbaut und sukzessiv die Produktkonkretisierung erhöht. Die Schrittfolge kann demnach beliebig oft durchgeführt werden, wobei mit zunehmender Konkretisierung ein Übergang vom Systementwurf in den domänenspezifischen Entwurf erfolgt und jeder Durchlauf in einer weiteren, untergeordneten Lösungsebene resultiert. Wie bereits erläutert, kann das Vorgehenskonzept zwar auch für den domänenspezifischen Entwurf angewendet werden, der Fokus liegt jedoch auf der Systementwurfsphase.

1. Schritt: Funktion n-ter Ebene definieren

Basierend auf der Funktion der übergeordneten Ebene muss diese für die *n*-te Ebene präzisiert bzw. konkretisiert werden. Bei zunächst gleichen Eingangs- und Ausgangsgrößen und Schnittstellen, die von der übergeordneten auf die betrachtete Ebene vererbt werden, müssen zur Konkretisierung zusätzliche Funktionen definiert werden, welche die funktionale Lösungsmöglichkeit exakter beschreiben. Dies umfasst insbesondere eine Aufteilung der Funktion der übergeordneten Ebene in Teilfunktionen für die betrachtete Ebene. Bezogen auf die Lösungsneutralität wird dabei der Lösungsraum eingeschränkt. Für die betrachtete Ebene stellt diese Konkretisierung eine notwendige Einschränkung dar, um zu einem definierten Systementwurf zu gelangen. Für die betrachtete Ebene sind die Funktionen wiederrum lösungsneutral.

Bei der Aufteilung der übergeordneten Funktion in Teilfunktionen müssen zunächst vererbte Elemente, Eingangs- und Ausgangsgrößen und Beziehungen (Relationen und Schnittstellen) beschrieben werden. Hinzu kommen die neu identifizierten Relationen, Eingangs- und Ausgangsgrößen und Schnittstellen der Teilfunktionen. Eine zentrale Bedingung bei der ebenenübergreifenden Funktionskonkretisierung ist, dass die aus übergeordneten Ebenen übernommenen Funktionskategorien unverändert bleiben. Entsprechend muss sich eine Funktionskategorie, welche auf übergeordneter Ebene definiert wurde, auf jeder untergeordneten Ebene wiederfinden. Eine Ergänzung um weitere Funktionskategorien ist zulässig, da sie die erforderliche Systemkonkretisierung darstellt.

Abb. 42 veranschaulicht diesen Zusammenhang. Auf der Ebene *n* wurde eine Funktion „Wandeln“ definiert, welche eine Eingangs- und eine Ausgangsgröße besitzt, die über Kreise neben dem Funktionssymbol dargestellt sind. Da sich auf jeder untergeordneten Ebene die Funktionskategorie der übergeordneten Ebene wiederfinden muss, wird die Funktion „Wandeln“ auf die Ebene *n+1* vererbt. Gleiches gilt für die Eingangs- und Ausgangsgröße, die ebenfalls auf die Ebene *n+1* vererbt werden, dargestellt über gestrichelte Linien. Zusätzlich wird eine neue Funktion „Vergrößern“ zur Konkretisierung des Lösungsraums definiert. Auf der Ebene *n+1* existieren daher für die Funktion „Wandeln“ die gleichen Größen wie auf der Ebene *n*. Dabei dient die Ausgangsgröße der Funktion „Wandeln“ der Funktion „Vergrößern“ als Eingangsgröße.

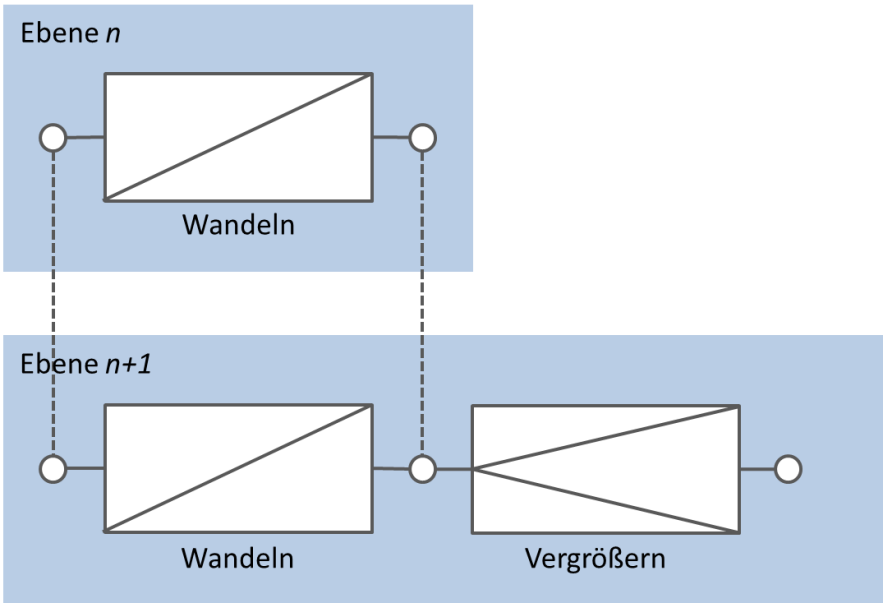


Abb. 42: Funktionskonkretisierung über zwei Ebenen

Das in Abb. 42 dargestellte Schema lässt sich prinzipiell auf alle Ebenen und Funktionskategorien anwenden. Hierbei werden Funktionskategorien, deren Größen und Schnittstellen auf die betrachtete Ebene vererbt und konkretisiert. Sofern sich durch die sukzessive Produktkonkretisierung neue (Teil-)Funktionen ergeben, werden deren Relation sowie die Größen und Schnittstellen identifiziert und letztere ebenfalls analysiert. Auf diese Weise entstehen Funktionsnetze, bei denen die funktionale Lösung der nullten Ebene durch eine schrittweise Einschränkung des Lösungsraums in über Eingangs- und Ausgangsgrößen vernetzte Teilfunktionen aufgeteilt wird. Hierbei wird jedoch nicht nur die Funktion höherer Ebene zerlegt, sondern es werden auch neue Funktionen eingebracht, die beispielsweise als Reaktion auf Störgrößen definiert wurden. Dies kann erforderlich sein, wenn bei der Funktionsrealisierung durch Komponenten Störgrößen entstehen, die einen Einfluss auf das System oder seine Bestandteile haben können, wie z.B. Abwärme. Für diese Größe wäre dann eine geeignete Funktion zu definieren, um ihren Einfluss zu reduzieren bzw. zu eliminieren. Konkret hieße das, dass eine Funktion „(Ab-)Leiten der Wärme in die Systemumgebung“ zu definieren wäre, wenn die Abwärme als Störgröße mit potentiellem Einfluss auf das System oder seine Bestandteile identifiziert wurde.

2. Schritt: Komponente n-ter Ebene festlegen

Analog zur Festlegung der Komponente nullter Ebene müssen für alle weiteren Ebenen geeignete, funktionsrealisierende Komponenten für die Funktionen der *n*-ten Ebene festgelegt werden. Wie auch bei den Funktionen können Komponenten sowohl Komponenten übergeordneter Ebene konkretisieren als auch auf der betrachteten Ebene neu hinzugekommene Funktionen realisieren. Im Gegensatz zu Funktionen, deren Funktionskategorie immer auf die betrachtete Ebene vererbt werden muss, kann eine Komponente übergeordneter Ebene entweder durch Subkomponenten auf der betrachteten Ebene durch deren logische Zerlegung ersetzt oder aber vererbt werden, falls auf der betrachteten Ebene keine Konkretisierung erfolgt.

In allen Fällen gilt, dass grundsätzlich alle Relationen, Eingangs- und Ausgangsgrößen sowie Schnittstellen von der übergeordneten Ebene *n* auf eine untergeordnete Ebene *n+1* vererbt werden, wobei Komponenten als funktionsrealisierende Elemente mindestens die gleichen Eingangs- und Ausgangsgrößen wie die Funktionen der betrachteten Ebene besitzen müssen. Die Vererbung von Größen erfolgt ausschließlich von übergeordneter zu untergeordneter Ebene innerhalb einer Systemsicht und von Funktionen auf Komponenten innerhalb einer Ebene, nicht jedoch von einer untergeordneten zu einer übergeordneten Ebene oder von Komponenten auf Funktionen innerhalb einer Ebene. Somit müssen bei der Komponentensicht je Ebene alle Eingangs- und Ausgangsgrößen aufgeführt werden, welche zuvor für Funktionen identifiziert wurden. Die Gesamtheit der Eingangs- und Ausgangsgrößen in der Komponentensicht entspricht dabei denen der jeweiligen Lösungsebene. Verglichen mit den Eingangs- und Ausgangsgrößen der Funktion können die zusätzlichen Größen der Komponente Störgrößen darstellen. Diese Bewertung und Klassifizierung erfolgt durch die Schrittfolge zur Schnittstellenidentifizierung und -analyse, welche für die Schritte „Funktion definieren“, „Komponente festlegen“ und „Optimierungsmaßnahmen implementieren“ der Systemkonkretisierung durchlaufen wird.

Bei der sukzessiven Produktkonkretisierung über Komponenten verschiedener Lösungsebenen können auch die Komponenten miteinander verknüpft werden. Entsprechend der Funktionsnetze werden Komponenten über ihre Eingangs- und Ausgangsgrößen, d.h. die technisch-physikalischen Schnittstellen, miteinander vernetzt. Um die Anzahl der Ebenen übersichtlich zu gestalten, sollten mit zunehmender Produktkonkretisierung und Ebene mehrere Komponenten

gleichzeitig festgelegt werden, solange hierdurch nicht der Lösungsraum mehr als nötig eingeschränkt wird. Die Einschränkung des Lösungsraums durch mehrere Komponenten ist dann nicht größer als nötig, wenn die Komponenten eine logische Konkretisierung einer Komponente übergeordneter Ebene darstellen, beispielsweise durch die Zerlegung eines Systems in seine Bestandteile, oder wenn definierte Funktionen durch Komponenten realisiert werden sollen. Während bei der Zerlegung eines Systems in seine Bestandteile sinnvollerweise mehrere Komponenten auf gleicher Ebene parallel festgelegt werden, sollte für die erstmalige Beschreibung der Realisierung einer Funktion nur eine Komponente festgelegt werden. Hierbei kann es sinnvoll sein, dass die Komponente zunächst nur ein (Sub-)System beschreibt, welches erst auf der nachfolgenden Ebene weiter konkretisiert wird. Auf diese Weise wird eine Produktkonkretisierung sichergestellt, die den Lösungsraum bewusst nur an den Stellen einschränkt, für die bereits entwicklungsrelevante Entscheidungen auf übergeordneten Ebenen getroffen wurden.

3. Schritt: Abgeleitete Anforderung n-ter Ebene definieren

Bei der Festlegung von funktionsrealisierenden Komponenten kann es mitunter unumgänglich sein, Systemelemente auszuwählen, die sich in Teilen negativ auf andere Systembestandteile auswirken. Die Beurteilung, ob Systemelemente einen negativen Einfluss haben können, ist Bestandteil der Schnittstellenanalyse, die für die Schrittfolge zur Konkretisierung des Lösungsraums durchlaufen und im Kap. 4.2.2.2 behandelt wird. Bestätigt diese Analyse negative Einflüsse, auf die eine systemseitige Reaktion erfolgen muss, wird in diesem Schritt eine Anforderung abgeleitet, welche die Zielrichtung im Umgang mit dem störenden Einfluss definiert. Diese Anforderung ergibt sich somit ausschließlich aus dem Vorhandensein eines störenden Einflusses. Liegt kein störender Einfluss vor, wird keine abgeleitete Anforderung definiert.

Da zwischen Anforderungen und anderen Systemelementen ausschließlich Relationen und keine technisch-physikalischen Schnittstellen bestehen können, besitzt für die verzahnte Schrittfolge nur die Identifizierung von Relationen Relevanz. Ebenso existieren keine Elemente oder Beziehungen, welche auf die abgeleitete Anforderung zu vererben wären, da diese nur innerhalb der betrachteten Ebene als Reaktion auf identifizierte Störgrößen Relevanz besitzt.

4. Optimierungsmaßnahmen implementieren

Wurden durch die Schnittstellenanalyse Störgrößen bestätigt und auf dieser Basis Anforderungen abgleitet, müssen geeignete Maßnahmen getroffen werden, um die Störgröße entweder zu eliminieren oder ihren Einfluss auf das System oder dessen Elemente zu reduzieren. Hierfür sind zunächst Funktionen zu definieren, welche die abgeleiteten Anforderungen umsetzen. Anschließend sind Komponenten festzulegen, welche die Funktionen realisieren. Ziel bei der Implementierung der Optimierungsmaßnahmen ist es, den Einfluss von Störgrößen zu eliminieren oder ihn bestmöglich zu reduzieren. Insbesondere Störgrößen, die durch Komponenten verursacht werden, die aufgrund der Umsetzung eines zugrundeliegenden Wirkzusammenhangs bzw. einer Prinziplösung resultieren, können jedoch oftmals nicht ohne eine Neukonzeptionierung des Systems im Sinne der geplanten Funktionen oder der gewählten Prinziplösung eliminiert werden. Zudem können auch durch geänderte Prinziplösungen oder Komponenten neue Störgrößen entstehen, sodass diese Maßnahme nachrangig umzusetzen ist. Treten Störgrößen auf, ist es zielführend den Einsatz von Alternativkomponenten als primäre Maßnahme zu prüfen. Generell müssen bei allen implementierten Optimierungsmaßnahmen Relationen, Größen und Schnittstellen identifiziert und analysiert werden. Dies dient der Sicherstellung, dass die

implementierten Maßnahmen ihrerseits keine negativen Wechselwirkungen mit anderen Systemelementen erzeugen.

Sofern der Einfluss von Störgrößen nicht eliminiert werden kann, muss ihre Wirkung reduziert werden. Hierfür wird methodisch nicht an dem verursachenden Element angesetzt. Stattdessen besteht das Ziel in der Reduzierung der Auswirkung der Störgröße. Dies ist exemplarisch in Abb. 43 dargestellt. Hier verursacht eine Komponente K_1.1 auf der Ebene 1 eine Störausgangsgröße, dargestellt als roter Kreis. Über die Schnittstellenanalyse konnte ein Einfluss auf die Komponente K_1.2 identifiziert werden, welcher zu verhindern ist. Gemäß der Schrittfolge zur Konkretisierung des Lösungsraums muss eine abgeleitete Anforderung A_1.1 (Design Constraint) definiert werden, welche eine Zielvorgabe zum Umgang mit der Störgröße liefert. Dies ist im unteren Bereich nach Implementierung der Optimierungsmaßnahme dargestellt. Zunächst wird eine neue Anforderung zum Umgang mit der Störgröße definiert. Diese Anforderung stellt ein Design Constraint dar und hat nur Bestand, solange eine Störgröße mit negativem Einfluss auf das System existiert. Sie wird mit der Komponente K_1.1, welche die Störgröße verursacht, über eine Erfordernisrelation verknüpft, dargestellt als gestrichelter Pfeil. Da es ausreichend ist, eine Erfordernisrelation einmalig in einer Kausalkette aus Störgröße, Design Constraint, Funktion und Komponente einzubinden, werden die Funktion zur Realisierung der Anforderung und die Komponente zur Realisierung der Funktion regulär über eine Realisierungsrelation beschrieben, ebenfalls über einen gestrichelten Pfeil dargestellt. Aus der abgeleiteten Anforderung A_1.1 wird eine Funktion F_1.1 zum Umgang mit der Störgröße definiert und für diese Funktion im Anschluss ein zusätzliches funktionsrealisierendes Element (Komponente K_1.3) festgelegt. Durch die Einführung dieser Komponente wird der Einfluss der Störgröße auf die Komponente K1.2 eliminiert. Zwischen den Komponenten ändern sich dementsprechend die Schnittstellen. Während vor Implementierung der Optimierungsmaßnahmen zwei direkte Schnittstellen zwischen Komponente K_1.1 und K_1.2 existierten, wird nach der Optimierung die Störgröße durch die zusätzliche Komponente K_1.3 aufgenommen, wodurch die Störgröße nicht mehr Komponente K_1.2 beeinflusst. Die Schnittstellen und Relationen sind in der Systemmodellierung nach dem Durchlauf einer Optimierungsschleife daher entsprechend anzupassen.

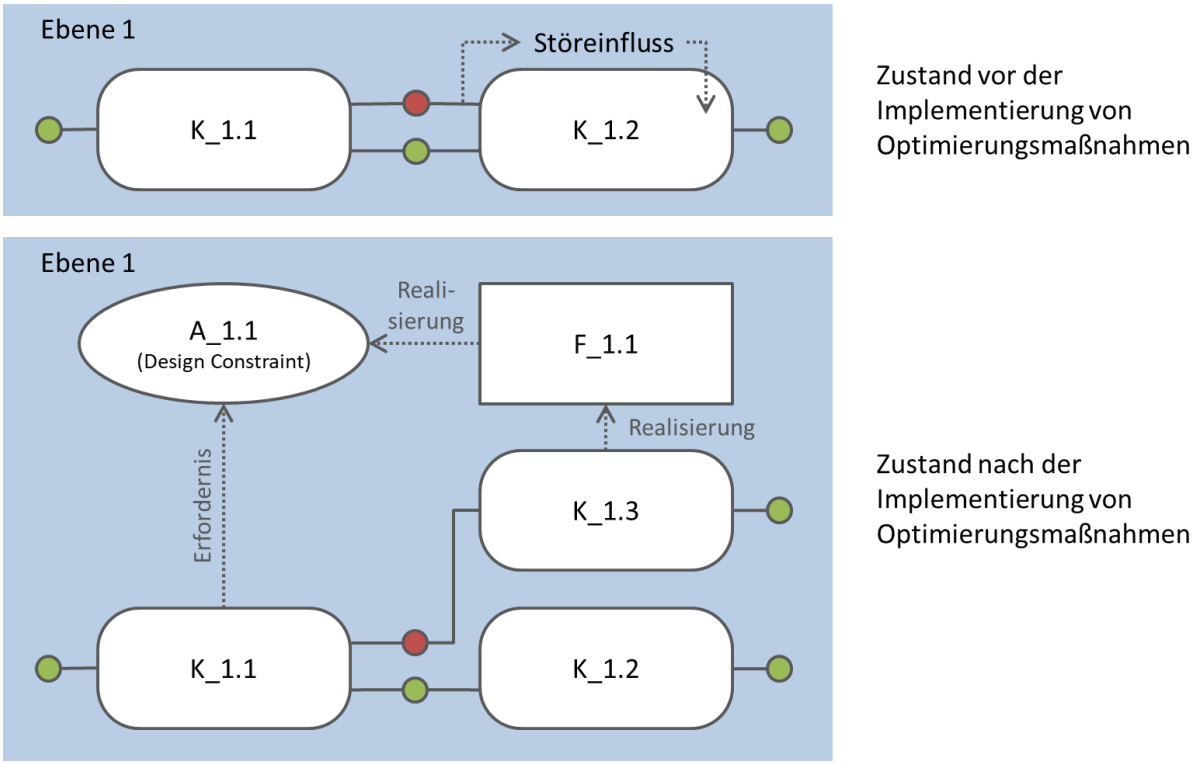


Abb. 43: Implementierung von Optimierungsmaßnahmen zum Umgang mit einer Störgröße

Wie im unteren Bereich der Abbildung dargestellt, sind sowohl die störgrößenverursachende Komponente K_1.1 als auch die Funktion F_1.1 zur Realisierung des Design Constraints über gerichtete Relationen mit der Anforderung A_1.1 verbunden (Erfordernis und Realisierung). Diese zunächst widersprüchlich erscheinende Tatsache, dass von Anforderungen über Funktionen zu Komponenten keine gleichgerichtete Relation existiert, ist in der Besonderheit von Design Constraints begründet, ausschließlich den Umgang mit Störgrößen zu beschreiben. Entfiele die Störgröße, existierte keine Erfordernisrelation, sodass ebenfalls das Design Constraint entfiele. Die Funktion, welche das Design Contraint realisiert, würde über ihre Relation somit nun „ins Leere“ führen. Dies veranschaulicht, dass nun auch nicht mehr die Funktion sowie die sie realisierende Komponente erforderlich wären. Diese Kausalität kann insbesondere bei späteren Änderungen hilfreich sein, beispielsweise im Rahmen des Engineering Change Managements, da nicht mehr erforderliche Relikte von Systemelementen durch Relationen, die „ins Leere laufen“, zielgerichtet und einfach identifiziert und entfernt werden können.

Um den Einfluss von Störgrößen zu reduzieren, bieten sich insbesondere die Funktionskategorien Verkleinern, Trennen, Speichern/Isolieren oder Leiten an, z.B. beim Ableiten von Wärme. Diese Funktionen müssen anschließend durch Komponenten, z.B. zusätzliche Kühlrippen, umgesetzt werden. Sollte zu einem späteren Zeitpunkt ein Systemelement entfallen oder geändert werden, sodass auch die Störgröße entfällt, hat die abgeleitete Anforderung keine Gültigkeit mehr und entfällt ebenfalls. Entsprechend müssen auch die sie realisierende Funktion und deren realisierende Komponente entfallen. Aus diesem Grund dürfen Funktionen und Komponenten zur Beschreibung des Umgangs mit Störgrößen nicht über hierarchische Relationen mit Funktionen oder Komponenten übergeordneter Ebene verknüpft werden, da sie keine direkte Systemkonkretisierung bzw. Vererbung bestehender Systemelemente darstellen, sondern lediglich an das Design Constraint gebunden sind.

Mit Abschluss der Implementierung von Optimierungsmaßnahmen ist die zu durchlaufende Schrittfolge einer Ebene abgeschlossen. Insbesondere bei einer großen Anzahl von Störgrößen oder schwerwiegenden Auswirkungen kann es jedoch sinnvoll sein, iterativ vorzugehen und bereits festgelegte Konkretisierungen auf der betrachteten oder der übergeordneten Ebene erneut zu prüfen und bei Bedarf Änderungen vorzusehen. Dies bedeutet, dass als Zielrichtung zunächst das Auftreten der Störgröße zu verhindern ist, sofern dies technisch möglich und aus Sicht des Projektmanagements verhältnismäßig ist. Sollten entsprechende Maßnahmen im Sinne einer Vermeidung nicht umsetzbar sein, müssen korrektive Maßnahmen ergriffen werden, um negative Einflüsse auf Systemelemente oder das Gesamtsystem zu verhindern oder zu mindern. Gleichzeitig kann auch ein Rücksprung zur Definition der zugrundeliegenden Funktion oder zur Festlegung der Komponente und eine geeignete Änderung auf der betrachteten oder ggf. auf übergeordneter Ebene sinnvoll sein, sofern dies nicht über den durch die Anforderung nullter Ebene vorgegebenen Lösungsraum ausgeschlossen ist.

4.2.2.2 Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse

Als ein zentraler Bestandteil des Vorgehenskonzepts müssen Relationen bestimmt und Größen bzw. Schnittstellen zwischen Systembestandteilen gezielt identifiziert und anschließend hinsichtlich ihres Einflusses auf das System bzw. seine Bestandteile analysiert werden. Hierfür wird eine Schrittfolge definiert, welche prinzipiell für jeden Schritt bei der Konkretisierung des Lösungsraums – mit Ausnahme des Schritts zur Definition der Anforderung nullter Ebene – durchlaufen wird und Vorgaben macht, welche Relationen grundsätzlich existieren können und wie Schnittstellen zu identifizieren und zu analysieren sind. Einschränkungen bestehen hinsichtlich der durchzuführenden Schritte, welche bedarfsgerecht auf die Einzelschritte der Schrittfolge zur Systemkonkretisierung angepasst sind. Zudem wird aufgezeigt, welche Relationen und Schnittstellen für Anforderungen, Funktionen und Komponenten charakteristisch sind und welche einer standardisierten Überprüfung bedürfen.

Den ersten Schritt bildet die Vererbung von Elementen, Größen und Beziehungen (Relationen und Schnittstellen). Hierüber werden die in der übergeordneten Ebene vorhandenen Informationen auf die betrachtete Ebene vererbt. Dies umfasst zunächst die Elemente. Wie zuvor erläutert, müssen für eine untergeordnete Ebene die Funktionskategorien vererbt werden. Sofern Komponenten nicht auf der untergeordneten Ebene konkretisiert werden, müssen diese ebenfalls vererbt werden. Erfolgt eine Konkretisierung auf der untergeordneten Ebene, so müssen alle zuvor existierenden Relationen und Schnittstellen auf die konkretisierten Komponenten der untergeordneten Ebene vererbt werden.

Bei der Relationsidentifizierung wird im zweiten Schritt der Fokus auf die strukturellen Relationen zwischen Elementen gelegt. Eine strukturelle Relation beschreibt die Beziehung zwischen Systemelementen aus systemtheoretischer Sicht, d.h. Beziehungen zwischen Elementen, die diesen zugewiesen sind, um die Systemstruktur abzubilden. Mögliche Relationsarten sind Hierarchie, Abfolge, Realisierung und Erfordernis. Ihnen liegt im Gegensatz zu Schnittstellen kein Fluss determinierbarer Größen zugrunde, welcher Elemente miteinander verbindet.

Während die Beschreibung struktureller Relationen primär der Komplexitätsbeherrschung dient, ergibt sich die Bedeutung der Schnittstellen, d.h. der Energie-, Stoff- oder Signalflüsse, aus den bereits aufgeführten Herausforderungen bei der Entwicklung technischer Systeme

hinsichtlich Informationsverfügbarkeit und Zuverlässigkeit. Bezogen auf die Informationsverfügbarkeit ist es für die an der Produktentwicklung beteiligten Domänen nicht unmittelbar ersichtlich, welche Größen durch Subsysteme anderer Domänen verursacht werden und ob diese einen Einfluss auf den eigenen domänenspezifischen Entwurf besitzen. Einhergehend mit der Unkenntnis relevanter Flüsse zwischen Systemelementen kann wiederum die Komplexität des Systems nicht beherrscht werden. In der Folge kann dies zu Problemen bezogen auf die Zuverlässigkeit des Gesamtsystems führen. Werden hingegen Schnittstellen bereits in den frühen Phasen der Produktentwicklung gezielt identifiziert und analysiert, können Erkenntnisse hieraus unmittelbar in den Systementwurf einfließen. Dies erhöht den Reifegrad des Produkts und reduziert die Anzahl erforderlicher Entwicklungsiterationen. Durch die Systematisierung und Implementierung des Vorgehenskonzepts zur Schnittstellenidentifizierung und -analyse wird parallel ein Beitrag zur Sicherstellung fähiger Produktentwicklungsprozesse geleistet. Dementsprechend stellen der dritte Schritt, die Identifizierung von Größen und Schnittstellen, und der vierte Schritt, deren Analyse, einen Beitrag zur Synchronisierung von Methoden der Reifegradabsicherung mit Fähigkeits- und Reifegradmodellen dar.

Im Gegensatz zur Schrittfolge bei der Konkretisierung des Lösungsraums unterscheidet sich die Schrittfolge nicht in ihrem prinzipiellen Ablauf für verschiedene Lösungsebenen. Die Relationsbestimmung und Schnittstellenidentifizierung und -analyse wird prinzipiell, aber nicht vollständig, für jeden Schritt der Schrittfolge zur Konkretisierung des Lösungsraums mit Ausnahme der Anforderungsdefinition nullter Ebene durchgeführt.

1. Schritt: Elemente, Größen und Beziehungen vererben

Zur Sicherstellung einer vollständigen und korrekten Konkretisierung des Lösungsraums müssen zunächst die Elemente (Funktionskategorien und ggf. Komponenten), Größen (Eingangs- und Ausgangsgrößen) und Beziehungen (Relationen und Schnittstellen) der übergeordneten Ebene übernommen werden. Entsprechend sind noch vor der eigentlichen Konkretisierung des jeweiligen Elements auf der betrachteten Ebene alle Elemente inklusive ihrer Größen zu übernehmen, welche nicht durch eine Konkretisierung ersetzt werden.

Für Funktionen bedeutet dies, dass die Funktionskategorien der übergeordneten Ebene vollständig zu übernehmen sind, da sich durch die Konkretisierung zwar neue und granulare Funktionen ergeben können, bereits auf übergeordneter Ebene existierende Funktionskategorien jedoch auch auf der betrachteten Ebene existieren und daher vererbt werden müssen. Bei Komponenten müssen nur solche Elemente vererbt werden, welche nicht durch die Systemkonkretisierung granularer beschrieben werden. Wird eine Komponente auf der untergeordneten Ebene über seine Bestandteile beschrieben und somit aufgeteilt, wird diese Komponente nicht vererbt. Für die Definition abgeleiteter Anforderungen sowie die Implementierung von Optimierungsmaßnahmen ist eine Vererbung von Beziehungen oder Elementen nicht erforderlich.

Unabhängig davon, ob Elemente direkt oder indirekt auf die betrachtete Ebene vererbt werden, ist die Vererbung von Eingangs- und Ausgangsgrößen erforderlich. Werden Elemente direkt vererbt, sind auch die zugehörigen Größen bzw. Schnittstellen (wenn bereits ein Fluss determinierbarer Größen zwischen mehreren Elementen vorliegt) zu übernehmen. Werden Elemente indirekt auf die betrachtete Ebene vererbt, d.h. durch Konkretisierung in ihre Bestandteile aufgeteilt, ist darauf zu achten, dass die zugrundeliegenden Eingangs- und Ausgangsgrößen bzw. Schnittstellen korrekt auf Elemente der betrachteten Ebene vererbt werden.

2. Relationen identifizieren

Sobald Funktionen oder Komponenten einer Lösungsebene definiert sind und Informationen der übergeordneten Ebenen übernommen wurden, werden die strukturellen Relationen zwischen den Systemelementen identifiziert. Dabei sind zunächst hierarchische Beziehungen zu betrachten, die sich zwischen den Lösungsebenen ergeben, beispielsweise bei der Zerlegung einer Komponente übergeordneter Ebene in seine Systembestandteile auf untergeordneter Ebene. Wie bereits in Kap. 4.2.1.3 dargelegt, können hierarchische Relationen nur innerhalb der DSM bestehen (S_A, S_F, S_K), wobei sie für Komponenten am bedeutsamsten sind und als einzige Relationsart ebenenübergreifend genutzt werden können.

Abb. 44 veranschaulicht hierarchische Relationen in der Matrix S_K. Die Komponente K_0.1 der nullten Ebene wird auf der untergeordneten Ebene 1 durch drei Komponenten K_1.1, K_1.2 und K_1.3 konkretisiert, ohne dabei selbst erneut aufgeführt zu werden. Dementsprechend besteht eine gerichtete, hierarchische Relation zwischen den Komponenten der Ebenen 0 und 1. Die Hierarchie wird als gerichtete Relation von den Komponenten untergeordneter Ebene auf die Komponente übergeordneter Ebene mit einem Pfeil dargestellt. Weitere, nicht-hierarchische Relationen zwischen den drei Komponenten der Ebene 1 oder deren Schnittstellen sind im Beispiel nicht betrachtet.

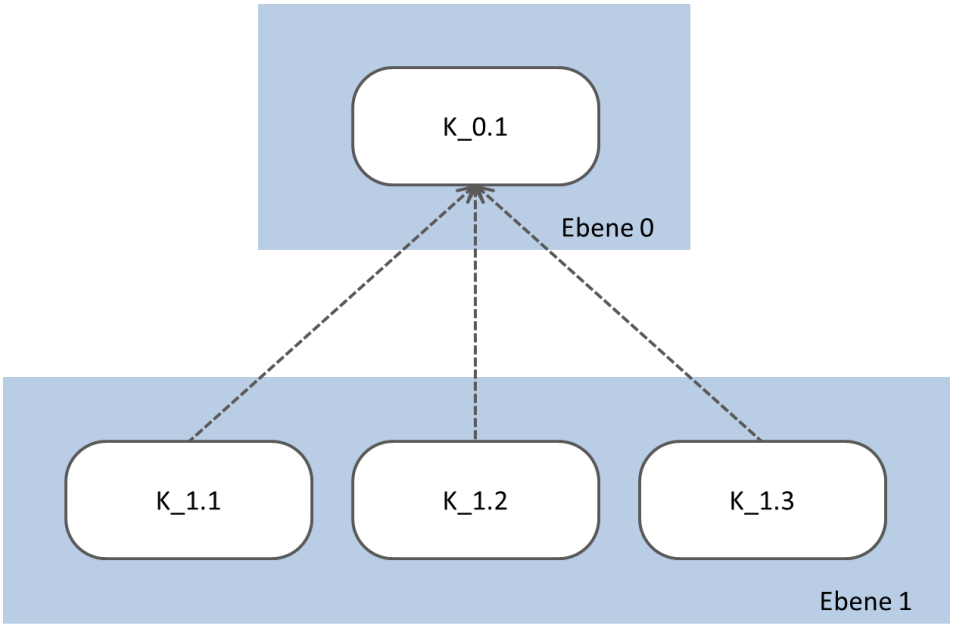


Abb. 44: Hierarchierelation zwischen Ebenen

Nach den hierarchischen Relationen werden Abfolgebeziehungen beschrieben. Diese existieren zwischen Elementen gleicher Ebene, jedoch nur die DSM S_F. Eine ebenenübergreifende Beschreibung von Abfolgerelationen ist nicht möglich, da alle Systemelemente einer übergeordneten Ebene auf untergeordneten Ebenen entweder direkt abgebildet oder auf der untergeordneten Ebene durch mehrere Elemente konkretisiert werden, welche das Elelement der übergeordneten Ebene ersetzen.

Abb. 45 stellt Abfolgerelationen zwischen drei nicht näher bestimmten Funktionen F_n.1, F_n.2 und F_n.3 der Ebene *n* dar. Die Richtung der Relation entspricht der logischen Abfolge und wird durch gestrichelte Pfeile gekennzeichnet.

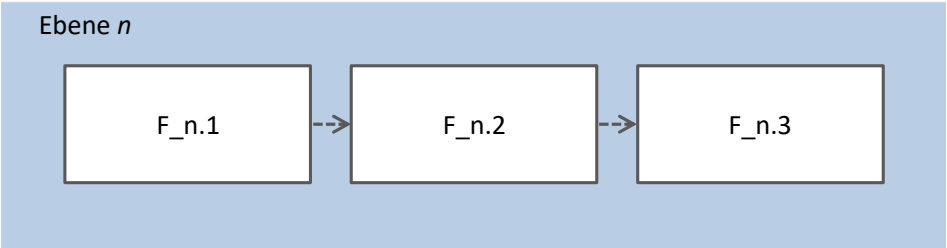


Abb. 45: Abfolgerelation zwischen Funktionen

Bei der sich anschließenden Identifizierung von Relationen zur Beschreibung einer Realisierung bestimmter Systemelemente durch andere Systemelemente können Relationen in den DMM S_{F,A_R}, S_{K,A_R} und S_{K,F_R} existieren. Eine Realisierung kann nur zwischen Elementen verschiedener Systemsichten existieren und liegt üblicherweise zwischen Funktionen und Anforderungen sowie Komponenten und Funktionen vor.

Die Realisierung von Anforderungen durch Komponenten bildet einen Sonderfall und kann vorliegen, wenn Komponenten nicht-funktionale Anforderungen realisieren. Dies sind Anforderungen, deren Realisierung nicht über Funktionen beschrieben werden kann, sondern direkt an eine Komponente geknüpft ist. Beispielsweise könnte für ein Steuergerät gefordert werden, dass die Bauform bestimmter SMD-Komponenten[28] eine konkrete Größe (z.B. 0603[29]) nicht überschreiten darf oder dass als mechanische Schnittstelle zwischen zwei Baugruppen Sechskantschrauben mit definierter Länge und definiertem Gewindedurchmesser zu nutzen ist. Diese Anforderungen lassen sich nicht über eine Funktion, sondern nur über die entsprechenden Komponenten realisieren, weshalb die Anforderungen als nicht-funktional bezeichnet sind. Die Realisierungsrelation ist in diesem Sonderfall zwischen Anforderung und Komponente zu setzen.

Bei der Identifizierung von Realisierungsrelationen ist zu prüfen, ob ggf. mehrere Elemente gleichzeitig eine Anforderung oder eine Funktion realisieren und entsprechend verknüpft werden müssen. Zudem ist auf eine vollständige Modellierung von Realisierungsrelationen zu achten, da anforderungs- oder funktionsrealisierende Elemente per se einen Zweck erfüllen. Bei nichtverknüpften Elementen wäre im Umkehrschluss zu prüfen, ob diese erforderlich sind. Auch für das Engineering Change Management stellt die Realisierungsrelation einen wichtigen Input dar, da bei allen Änderungen über diese Verknüpfung schnell und transparent überprüft werden kann, ob Änderungen geforderte Funktionen oder zugrundeliegende Anforderung betreffen. Dies ermöglicht eine gezielte Analyse, ob Änderungen einen Einfluss auf die Anforderungs- oder Funktionsrealisierung besitzen.

Abb. 46 zeigt eine Kausalkette, bei der die Realisierung einer Funktion durch eine Komponente und die Realisierung einer Anforderung durch diese Funktion erfolgt. Die Relationsrichtung ist durch gestrichelte Pfeile dargestellt. Am Beispiel ist erkennbar, dass in der Symboldarstellung matrizenübergreifende Relationen darstellbar sind, die in der Matrizendarstellung über zwei DSM (S_{K,F_R} und S_{F,A_R}) darzustellen wären. Die Richtung der Relation stellt die erwartete Realisierung dar und bildet den Umkehrschluss zum Vorgehen bei der Systemkonkretisierung, bei der Anforderungen definiert und durch Funktionen realisiert werden, welche anschließend durch Komponenten realisiert werden.

[28] SMD: Surface-mounted device, ein auf Leiterkarten gelötetes, elektronisches Bauteil
[29] Metrische Beschreibung der Größe elektronischer Bauteile, entsprechend 0603 = 0,6 mm x 0,3 mm

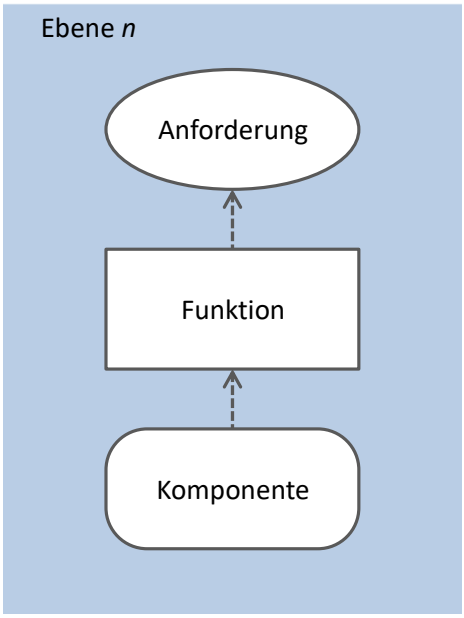


Abb. 46: Realisierungsrelation zwischen Elementen

Als letztes müssen Erfordernisrelationen identifiziert werden. Eine Erfordernisrelation liegt vor, wenn über Anforderungen konkrete Funktionen oder Komponenten explizit gefordert werden. Diese werden in den Matrizen $S_{A,F}$ oder $S_{A,K}$ modelliert. Eine andere Voraussetzung für eine Erfordernisrelation kann aufgrund von Störgrößen vorliegen. Diese können üblicherweise erst identifiziert werden, wenn die in den folgenden Schritten durchgeführte Schnittstellenidentifizierung und -analyse Störgrößen aufzeigt, auf die mit der Definition einer neuen, abgeleiteten Anforderung (Design Constraint) reagiert werden muss. Ist dies der Fall, muss die Erfordernisrelation auf die abgeleitete Anforderung von dem Element ausgehend gebildet werden, welches initial die Störgröße verursacht, d.h. die Komponente. Dies wird in der Matrix S_{K,A_E} beschrieben. Diese Relation wurde bereits in der Abb. 43 dargestellt.

Die Identifizierung und Beschreibung struktureller Relationen unterstützt auf diese Weise das Systemverständnis und die Komplexitätsbeherrschung. Gleichzeitig dient es als Wissensspeicher für das Engineering Change Management, wenn Änderungen gefordert sind und ihre strukturelle Auswirkung auf das System zu bewerten ist.

3. Größen und Schnittstellen identifizieren

Im Anschluss an die Bestimmung struktureller Relationen sind die Eingangs- und Ausgangsgrößen sowie die technisch-physikalischen Schnittstellen zu identifizieren. Neben den vererbten Größen und Schnittstellen können Funktionen und Komponente neue Größen besitzen, welche hier zu erfassen sind. Wird über Größen ein Fluss (Energie, Stoff oder Signal) zwischen Elementen gebildet, resultieren Schnittstellen zwischen den beteiligten Elementen.

Bei der Schnittstellenidentifizierung ist darauf zu achten, dass Schnittstellen nur zwischen Elementen gleicher Systemsicht existieren können. Obwohl Eingangs- und Ausgangsgrößen von Funktionen auf Komponenten vererbt werden, existiert zwischen Funktionen und Komponenten keine Schnittstelle im Sinne eines Energie-, Stoff- oder Signalflusses. Die Vererbung dient somit ausschließlich der Sicherstellung, dass alle für Funktionen identifizierten Größen auch für Komponenten existieren müssen. Weiterhin existieren Schnittstellen nur innerhalb einer Ebene. Wie bereits in Abb. 44 dargestellt, gibt es zwar Relationen, jedoch keine Schnittstellen

zwischen der Komponente 1 auf der Ebene *n* und den Komponenten 1.1, 1.2 und 1.3 der Ebene *n+1*. Hintergrund ist, dass alle Größen bereits von der Ebene *n* auf die Ebene *n+1* vererbt wurden. Dementsprechend stellt Ebene *n+1* eine granularere Beschreibung der Ebene *n* dar, ohne dass jedoch Energie-, Stoff- oder Signalflüsse zwischen den Ebenen existieren.

Schnittstellen werden gemäß der Schrittfolge zur Konkretisierung des Lösungsraums zunächst für Funktionen und anschließend für Komponenten identifiziert. Bei der Schnittstellenidentifizierung müssen alle Eingangs- und Ausgangsgrößen der Systemelemente identifiziert werden. Sie liegen vor, wenn zwischen Systemelementen Energie-, Stoff- oder Signalflüsse existieren.

Abb. 47 zeigt zwei Komponenten K_n.1 und K_n.2 der Ebene *n* in der Symboldarstellung vor und nach der durchgeführten Größen- und Schnittstellenidentifizierung. Vor der Identifizierung existieren keine erfassten Größen oder technisch-physikalischen Schnittstellen zwischen den Komponenten. Durch die Identifizierung werden existierende Schnittstellen erfasst und anschließend modelliert. Im Beispiel besitzt Komponente K_n.1 eine Eingangs- und eine Ausgangsgröße. Diese Ausgangsgröße stellt für Komponente K_n.2 die Eingangsgröße dar, entsprechend handelt es sich um einen Fluss zwischen den Komponenten. Komponente K_n.2 besitzt im Beispiel zwei Ausgangsgrößen.

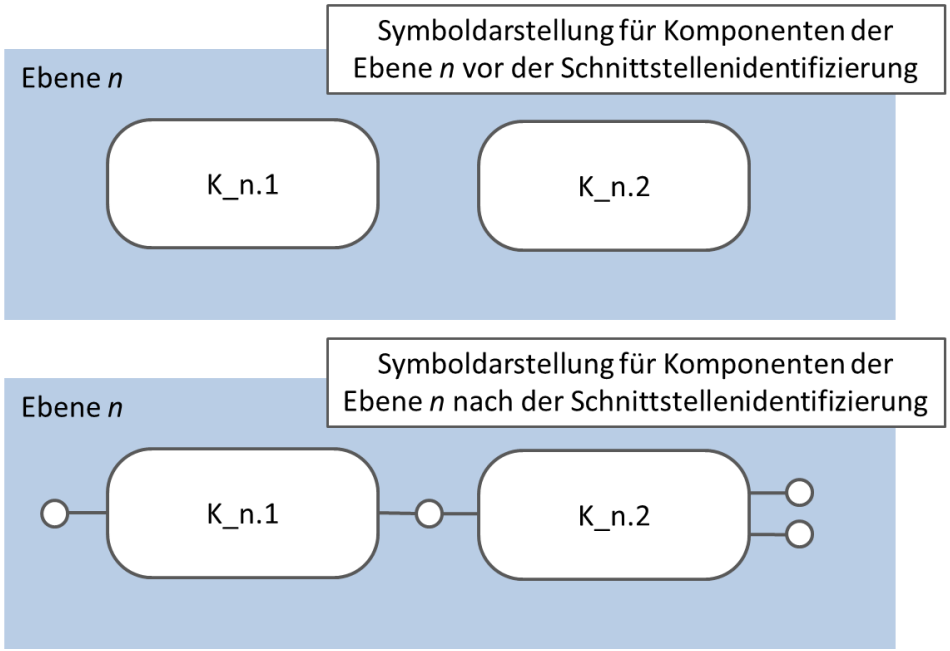


Abb. 47: Schnittstellenidentifizierung

Die Art der Größen (Stoff, Energie oder Signal) ist ebenfalls zu erfassen. Sofern möglich, sollten die identifizierten Größen dabei quantitativ beschrieben werden, d.h. mit Wert und Einheit der Größe. Für die parallel erfolgende Systemmodellierung determiniert die Art der Größen auch die Systemmatrizen, in denen die Beziehung zwischen den Komponenten beschrieben wird. Würde es sich bei den dargestellten Größen um Energie handeln, wären die Schnittstellen zwischen den Komponente K_n.1 und K_n.2 in der Matrix S_{K_EF} zu modellieren. Die Klassifizierung in Soll- oder Störgrößen erfolgt bei der Schnittstellenanalyse. Die Identifizierung der Größen und Schnittstellen ist somit wertungsfrei und stellt lediglich das Vorhandensein von Größen sowie ggf. deren Wert und Einheit fest.

4. Größen und Schnittstellen analysieren

Bei der Größen- und Schnittstellenanalyse werden die zuvor identifizierten Eingangs- und Ausgangsgrößen sowie die Schnittstellen hinsichtlich ihres Einflusses auf das System analysiert und bewertet. Dies kann sowohl qualitativ als auch quantitativ erfolgen. Sofern die Größen und Schnittstellen zwar identifiziert, aber noch nicht quantifiziert sind, was insbesondere in den frühen Phasen der Produktentwicklung wahrscheinlich ist, muss die Analyse qualitativ durchgeführt werden. Hierbei wird unabhängig von Wert und Einheit der physikalischen Größe nur die potentielle Auswirkung auf das Systemelement betrachtet, auf welches sie als Eingangsgröße wirkt. Besitzt die Ausgangsgröße eines Elements für ein anderes Element oder das Gesamtsystem einen potentiell negativen Einfluss, wird sie unabhängig von Wert und Einheit der Größe als Störgröße bezeichnet. Besitzt sie nur unter bestimmten Voraussetzungen einen negativen Einfluss auf ein Element oder das Gesamtsystem, der aber abhängig von Wert und Einheit der Größe ist, wird sie solange als Störgröße behandelt, bis sie entweder durch eine quantitative Bewertung als irrelevant oder als relevant, d.h. mit störendem Einfluss, bewertet und entsprechend bestätigt wurde. Die Schnittstellenanalyse ist grundsätzlich möglich, unabhängig davon, ob identifizierte Störgrößen qualitativ erfasst oder quantitativ spezifiziert wurden.

Sofern Größen quantifiziert sind, ist ihr Einfluss über quantitative Analysen zu bewerten. Hierfür bieten sich insbesondere Berechnungsmodelle, Simulationen oder Richtlinien an, welche den genauen Einfluss der Störgröße bewerten können, vgl. [Riekhof et al. 2013b]. Sofern eine potentielle Störgröße einen negativen Einfluss in Form von Leistungs- bzw. Wirkungsgradverlusten oder einer reduzierten Zuverlässigkeit verursacht, ist sie als Störgröße bestätigt, andernfalls ist sie nicht als Störgröße zu behandeln.

Abb. 48 veranschaulicht die Anwendung der Schnittstellenanalyse. Die Komponente K_n.1 besitzt eine Eingangsgröße und zwei Ausgangsgrößen, welche der Komponente K_n.2 als Eingangsgrößen dienen. Komponente K_n.2 erzeugt aus diesen Eingangsgrößen eine Ausgangsgröße. Während sich dieser Zusammenhang bereits aus der Schnittstellenidentifizierung ergibt, müssen für die Schnittstellenanalyse diese Größen analysiert und als Soll- oder Störgröße klassifiziert werden. Im aufgeführten Beispiel handelt es sich bei der zweiten Ausgangsgröße der Komponente K_n.1 um eine Störgröße. Dies wird in der Symboldarstellung über die farbigen Kreise veranschaulicht, welche die Größen darstellen. Grüne Kreise stehen demnach für Solleingangs- oder Sollausgangsgrößen, rote Kreise für Störeingangs- oder Störausgangsgrößen. In der Abbildung wird die Durchführung der Schnittstellenanalyse durch die Lupe symbolisiert.

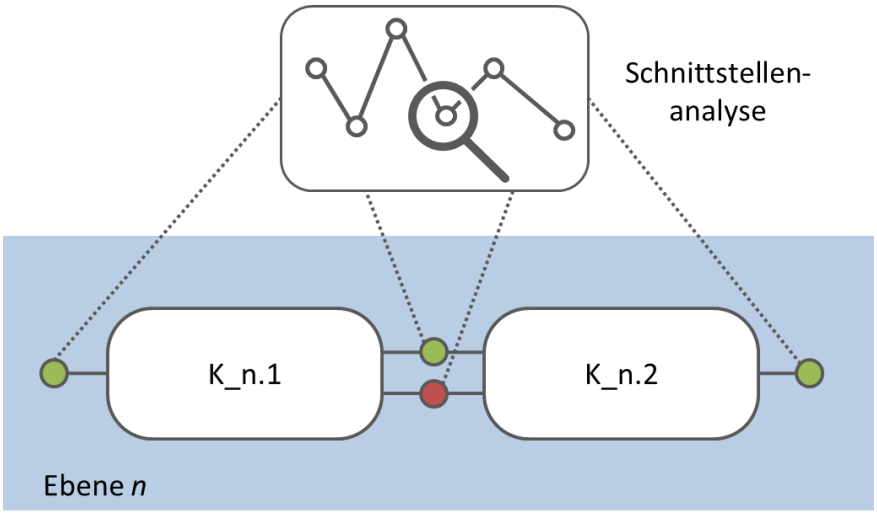


Abb. 48: Schnittstellenanalyse

Identifizierte Störgrößen können unterschiedlichen Einfluss auf Systemelemente oder das Gesamtsystem besitzen. Die Unterscheidung der Störeinflüsse dient der präziseren Identifizierung von Störgrößen und ist im Rahmen der Schnittstellenanalyse anzuwenden. In Summe werden drei Fälle unterschieden, die jeweils von der Störeingangsgröße ausgehen.

Wie in Abb. 49 dargestellt, kann von der Störgröße ein Störeinfluss auf die Sollausgangsgröße des Elements ausgehen, auf das sie einwirkt (Fall 1). Dies äußerst sich beispielsweise durch Leistungsverluste. Bezieht sich der Störeinfluss auf das Systemelement selbst (Fall 2), führt dies üblicherweise zu einer Verschlechterung von Zuverlässigkeitsparametern. Als Sonderform von Fall 2 ist es auch möglich, dass eine erzeugte Störausgangsgröße das erzeugende Element selbst beeinflusst. Fall 3 führt zu einer neuen bzw. der gleichen Störausgangsgröße des von der Störeingangsgröße betroffenen Elements. Grundsätzlich können die genannten Störeinflüsse auch parallel auftreten.

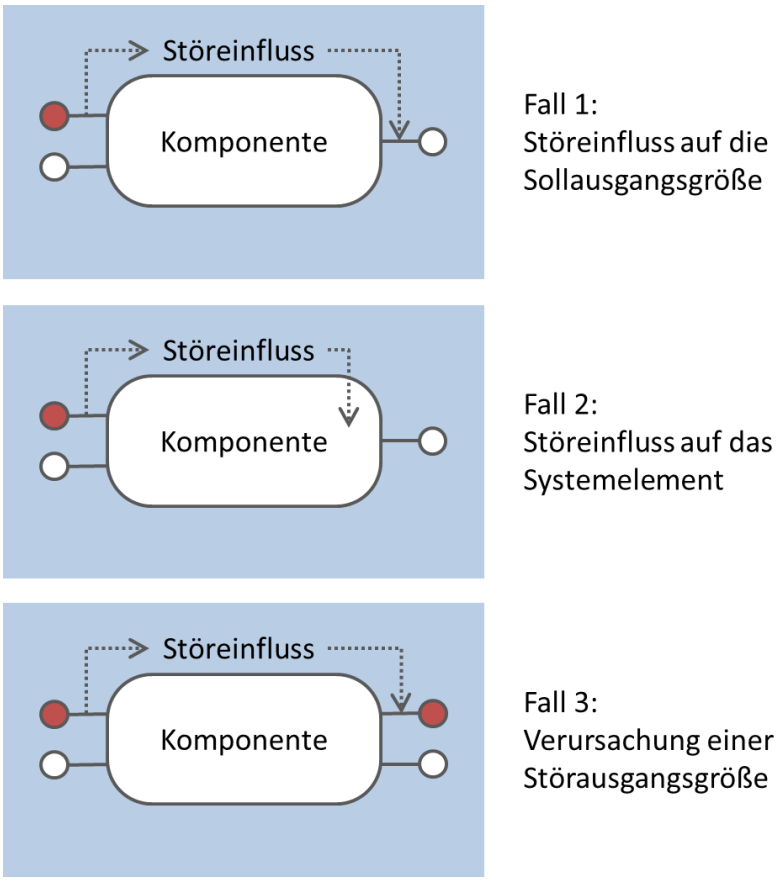


Abb. 49: Störeinflüsse einer Größe

Der Schritt zur Durchführung der Schnittstellenanalyse ist integraler Bestandteil der Schrittfolge des Vorgehenskonzepts. Für die inhaltliche Analyse wird jedoch auf bestehende Methoden und Vorgehensmodelle verwiesen. So können durch die identifizierten und quantifizierten Störgrößen Berechnungen ermöglicht werden, welche Aussagen über Wirkungsgradverluste oder eine reduzierte Zuverlässigkeit ermöglichen. Auch Tests können ein geeignetes Mittel sein, den Einfluss von Störgrößen zu evaluieren. Hierfür stellen quantifizierte Störgrößen den notwendigen Input dar, um bereits frühzeitig Bauteile zielgerichtet testen zu können.

Mit Abschluss der Relationsbestimmung und Schnittstellenidentifizierung und -analyse ist die gesamte Schrittfolge zur Systemkonkretisierung und -analyse für die jeweilige Lösungsebene abgeschlossen. Sie ist für jede weitere, untergeordnete Lösungsebene zu wiederholen, bis ein Entwicklungsstand erreicht ist, der in die Phase des domänenspezifischen Entwurfs übergeben werden kann. Die Entscheidungsbasis, ob alle Tätigkeiten der Schrittfolge die erwarteten Ergebnisse erzielt haben und die Produktkonkretisierung auf einer untergeordneten Lösungsebene fortgesetzt werden kann, bildet die Ermittlung des Produktreifegrads, welche im folgenden Kapitel vorgestellt wird.

4.2.3 Schrittfolge zur Ermittlung des Produktreifegrads

Die Ermittlung des Produktreifegrads stellt ein wichtiges Instrument in der Produktentwicklung dar, um frühzeitig Aussagen über die Anforderungserfüllung und den erwarteten Entwicklungsstand entlang des Projektterminplans zu generieren. Hierfür werden an definierten Meilensteinen die vorhandenen Produktmerkmale mit den geforderten Produktmerkmalen abgeglichen, worüber ein Reifegrad des Produkts ermittelt wird. Die Meilensteine dienen dabei als

sogenannte Quality Gates, bei denen in Abhängigkeit des ermittelten Reifegrads entschieden wird, ob die Produktentwicklung ohne Änderungen fortgesetzt werden kann, ob Korrekturen erforderlich sind oder ob der aktuelle Entwicklungsfortschritt komplett verworfen und zu einem vorherigen Quality Gate zurückgesprungen werden muss.

4.2.3.1 Meilensteine in den frühen Phasen der Produktentwicklung

Gemäß der Einordnung des Vorgehenskonzepts in den PEP (vgl. Kap. 4.1) werden ausschließlich Meilensteine betrachtet, welche ab dem Feststehen einer Anforderungsbasis in Form eines Lastenhefts und bis zur Übergabe des Systementwurfs in den domänenspezifischen Entwurf existieren. Die Meilensteine orientieren sich dabei am Vorgehen zur Konkretisierung des Lösungsraums und den resultierenden Lösungsebenen. Mit Ausnahme der nullten Ebene wird die Schrittfolge zur Ermittlung des Produktreifegrads für jede Lösungsebene durchlaufen, d.h. nach Durchführung der Schrittfolgen zur Konkretisierung des Lösungsraums, zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse sowie der begleitenden Systemmodellierung. Abb. 50 konkretisiert hierbei die allgemeine Verknüpfung der Meilensteine bei der funktionsorientierten Produktentwicklung (vgl. Abb. 31) für die Systementwurfsphase. Die aus der Systemkonkretisierung und -analyse resultierenden Lösungsebenen schließen ab der Ebene 1 jeweils mit einem Meilenstein ab, welche durch Rauten dargestellt sind. Für die nullte Lösungsebene ist aufgrund der geringen Produktkonkretisierung kein Meilenstein vorgesehen. Die Bewertung des Produktreifegrads setzt voraus, dass der Entwicklungsfortschritt je Lösungsebene im Systemmodell abgebildet ist und dieses als Grundlage der Reifegradbewertung zur Verfügung steht, weshalb die Schrittfolge für die Systemkonkretisierung und -analyse mit dem Systemmodell und der Schrittfolge die Ermittlung des Produktreifegrads über die Meilensteine gekoppelt ist. Auf Ebene *z*, für welche der letzte Meilenstein vorgesehen ist, liegt der Systementwurf in der gewünschten Konkretisierung vor und kann bei positiver Bewertung des Produktreifegrads anschließend in die Phase des domänenspezifischen Entwurfs übergeben werden.

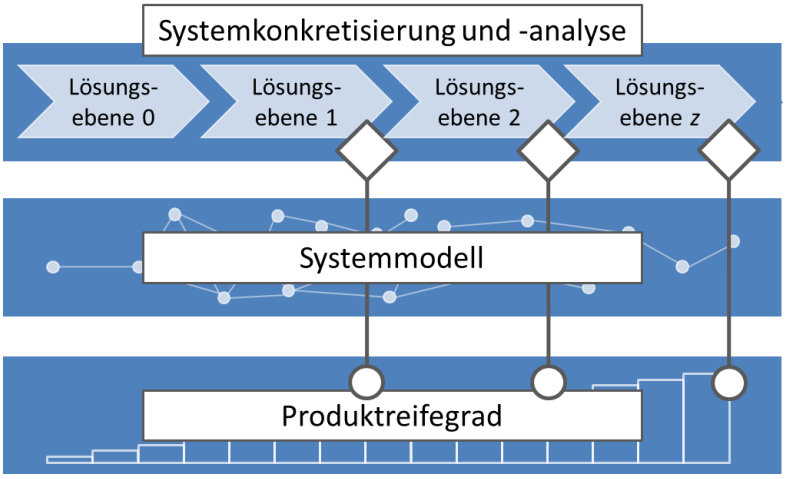


Abb. 50: Meilensteinzuordnung der Lösungsebenen

Da durch die sukzessive Einschränkung des Lösungsraums der Konkretisierungsgrad des Systementwurfs je Ebene wächst, nimmt auch der Umfang der Reifegradbewertung zu und wird damit automatisch dem Entwicklungsfortschritt angepasst. Gleichzeitig gibt es keine Vorgabe, wie viele Lösungsebenen bis zur Übergabe des Systementwurfs an den domänenspezifischen Entwurf durchlaufen werden müssen. Die bisherigen Erfahrungswerte deuten jedoch darauf

hin, dass die Anzahl sinnvoller Lösungsebenen im mittleren einstelligen Bereich liegt, bevor der Systementwurf in die Phase des domänenspezifischen Entwurfs übergeben werden kann. Die Bewertungskriterien werden nachfolgend vorgestellt.

Wie bereits erläutert, fokussiert die vorliegende Arbeit ausschließlich auf die Phase des Systementwurfs. Grundsätzlich ist das vorgestellte Vorgehenskonzept einschließlich der Lösungsebenen und der bei diesen Meilensteinen vorgesehenen Produktreifegradbewertung auch für die Phase des domänenspezifischen Entwurfs geeignet. Entsprechend kann der Ansatz (ein Meilenstein je Lösungsebene) ebenfalls in nachgelagerten Phasen angewendet werden. Da dieser Anwendungsfall kein Betrachtungsgegenstand der vorliegenden Arbeit ist, werden jedoch keine Bewertungskriterien für den domänenspezifischen Entwurf abgeleitet.

4.2.3.2 Bewertungskriterien des Produktreifegrads

Zur Ermittlung eines konkreten Produktreifegrads sind Bewertungskriterien erforderlich, welche die erwarteten Ergebnisse je Meilenstein definieren und ihren Einfluss auf die Reifegradbewertung festlegen. Die Erreichung eines hohen Reifegrads setzt voraus, dass die erwarteten Ergebnisse erfüllt sind. Durch die sukzessive Produktkonkretisierung und den damit verbundenen steigenden Umfang des Systementwurfs erhöht sich je Meilenstein auch der Umfang der erwarteten Ergebnisse. Somit müssen für die Erreichung eines hohen Reifegrads bei der schrittweisen Annäherung des Systementwurfs an den domänenspezifischen Entwurf umfangreichere Tätigkeiten in den einzelnen Schrittfolgen des Vorgehenskonzepts durchgeführt und nachgewiesen werden. Ist dies nicht der Fall, würde ein sinkender Reifegrad bei der ebenenbezogenen Bewertung resultieren. Der ermittelte Reifegrad dient als Entscheidungsgrundlage, ob der Produktreifegrad der Lösungsebene bzw. des Systementwurfs zur Übergabe auf die nächste Ebene bzw. in die nächste PEP-Phase ausreichend ist und bildet damit die Logik der Ampelkaskade des RGA-Modells ab. Entsprechend stellt ein zu geringer Reifegrad ein Kriterium dar, um den vorhandenen Stand vor der Übergabe zurückzuweisen und ihn bis zur Erreichung des geforderten Reifegrads zu optimieren.

Für jede Lösungsebene kann ein Reifegrad zwischen 0 % und 100 % erreicht werden. Ein Prozentsatz, bei dessen Unterschreitung ein Quality Gate nicht passiert werden darf, kann projektspezifisch festgelegt werden. Für die vorliegende Arbeit erfolgt die Ableitung der Bewertungsskala für das Vorgehenskonzept in Anlehnung an das ASPICE-Modell. Bei diesem wird die Anforderungserfüllung in den vier Abstufungen (Ratings) „not achieved“ (≥0 % … ≤15 %), „partially“ (>15 % … ≤50 %), largely (>50 % … ≤85 %) und „fully“ (>85 % … ≤100 %) bewertet [VDA 2023, S. 19]. Dabei sind nur die Ratings „largely“ und „fully“ ausreichend, um ein ASPICE Level 1 … 5 zu erreichen, wobei ein „largely“ nur für einzelne Prozessattribute als ausreichend für das Bestehen des Levels angesehen wird, während die meisten Prozessattribute ein „fully“ erreichen müssen [ebd., S. 22]. Entsprechend führen im Vorgehenskonzept alle Reifegrade unterhalb von 50 % zu einer Zurückweisung. Reifegrade zwischen 50 % und 85 % erfordern eine Korrektur des vorhandenen Systementwurfs und bei Reifegraden größer 85 % gilt ein Quality Gate als bestanden.

Die Bewertungskriterien sind analog zum RGA-Modell aufgebaut, so dass die zugehörigen Fragen binär mit „Ja“ oder „Nein“ beantwortet werden können. Sofern eine Frage mit „Ja“ beantwortet wird, gilt das entsprechende Bewertungskriterium als erfüllt. Muss eine Frage mit

„Nein“ beantwortet werden, ist das zugehörige Kriterium nicht erfüllt. Ist eine Frage unzutreffend, da beispielsweise auf einer Ebene keine abgeleiteten Anforderungen als Reaktion auf Störgrößen festgelegt wurden und demnach keine Optimierungsmaßnahmen implementiert werden müssen, ist das Bewertungskriterium als „n/a“ zu bewerten. In diesem Fall wird das Kriterium für die Bewertung des Produktreifegrads als erfülltes Bewertungskriterium angesehen. Da sich nicht anwendbare Bewertungskriterien oftmals auf Störgrößen, abgeleitete Anforderungen oder Implementierungsmaßnahmen beziehen, besteht jedoch ein Risiko, dass diese nicht oder nur unvollständig betrachtet und folglich als „n/a“ gekennzeichnet wurden. Der Produktreifegrad als Prozentangabe kann dieses Risiko nicht adäquat ausdrücken. Um Risiken einer unvollständigen Durchführung der Schrittfolge zur Systemkonkretisierung und -analyse transparent darzustellen, wird daher zusätzlich zur Produktreifegradbewertung ein Vertrauensintervall angegeben. Nur wenn maximal zwei Kriterien mit n/a bewertet werden, ist das Vertrauensintervall mit grün zu bewerten. Drei bis vier Angaben einer n/a-Bewertung führen zu einem gelben Vertrauensintervall und ab fünf n/a-Bewertungen zu einer roten Einstufung. Auf diese Weise können Risiken transparent dargestellt werden. Im Unterschied zur Bewertungsskala, anhand der entschieden wird, ob ein Quality Gate passiert werden kann und welche sich am ASPICE-Schema orientiert, ist die Bewertung eines Vertrauensintervalls nicht an entsprechende Kriterien angelehnt. Ziel ist an dieser Stelle die kritische Hinterfragung der Validität der vorgenommenen Bewertung. Die Anzahl der n/a-Angaben, die zu einer gelben bzw. roten Bewertung führen, sollte daher bedarfsgerecht und anwendungsspezifisch erfolgen.

Inhaltlich orientieren sich die Bewertungskriterien einerseits an den aus dem RGA-Modell abgeleiteten Anforderungen (vgl. Tab. 14), andererseits an den Erkenntnissen zum Vorgehensmodell und den zugehörigen Schrittfolgen, wobei die Bewertungskriterien basierend auf den Schrittfolgen zur Konkretisierung der Lösungsebenen, der Relationsbestimmung und Schnittstellenidentifizierung und -analyse sowie der Systemmodellierung klassifiziert werden, um eine einfache Bewertung zu ermöglichen.

Wie in Tab. 18 zusammengefasst, definieren Bewertungskriterien mit Bezug auf die Konkretisierung des Lösungsraums erwartete Ergebnisse, die über die Durchführung der zugehörigen Schrittfolge (vgl. 4.2.2.1.2) erzeugt werden können. Entsprechend erfolgt die Überprüfung, ob der Lösungsraum sukzessiv konkretisiert wurde. Das Ziel besteht in der Sicherstellung, dass alle für die Produktkonkretisierung erforderlichen Schritte durchgeführt wurden.

Im Unterschied hierzu fokussieren die Bewertungskriterien der Relationsbestimmung und Schnittstellenidentifizierung und -analyse auf die Ergebnisse bei der Erfassung und Analyse von Systemstrukturen. Während die Relationsbestimmung einen Beitrag zur Komplexitätsbeherrschung leistet, dient die Erfassung und Bewertung der technisch-physikalischen Schnittstellen der Sicherstellung, dass Schnittstellen erkannt und bewertet werden.

Für die Systemmodellierung sichern die zugehörigen Bewertungskriterien die Transparenz des Vorgehens ab. Während der eigentliche Systementwurf beispielsweise in Form eines 3D-Datenmodells vorliegt bzw. Simulationen oder Kalkulationen zur Auslegung durchgeführt wurden, ermöglichen erst die im Systemmodell verfügbaren Informationen eine direkte Nachweisführung der Umsetzung erwarteter Ergebnisse.

Tab. 18: Bewertungskriterien des Produktreifegrads

Schrittfolge	Erwartete Ergebnisse	Gewichtung
Konkretisierung des Lösungsraums	Auf der betrachteten Ebene wurde mindestens eine zusätzliche Funktion im Sinne der Systemkonkretisierung definiert	10 %
	Für jede Funktion der betrachteten Ebene wurde mindestens eine funktionsrealisierende Komponente festgelegt	10 %
	Beim Auftreten von Störgrößen auf der betrachteten Ebene wurden Anforderungen zum Umgang mit diesen abgeleitet	10 %
	Beim Vorliegen abgeleiteter Anforderungen wurden Optimierungsmaßnahmen auf der betrachteten Ebene implementiert	10 %
Relationsbestimmung und Schnittstellenidentifizierung und -analyse	Funktionen und Komponenten wurden von der übergeordneten Ebene auf die betrachtete Ebene vererbt oder auf der betrachteten Ebene konkretisiert	5 %
	Eingangs- und Ausgangsgrößen und Schnittstellen der übergeordneten Ebene wurden auf Elemente der betrachteten Ebene vererbt	5 %
	Hierarchische Relationen von Elementen der betrachteten Ebene zu Elementen der übergeordneten Ebene wurden vollständig bestimmt	5 %
	Strukturelle Relationen von Elementen der betrachteten Ebene wurden vollständig bestimmt	5 %
	Größen und technisch-physikalische Schnittstellen von Elementen der betrachteten Ebene wurden vollständig identifiziert	5 %
	Größen und technisch-physikalische Schnittstellen von Elementen der betrachteten Ebene wurden vollständig analysiert	5 %
	Auftretende Störgrößen der betrachteten Ebene wurden hinsichtlich ihres Einflusses auf das System bewertet	5 %
Systemmodellierung	Funktionen der betrachteten Ebene wurden vollständig modelliert	5 %
	Komponenten der betrachteten Ebene wurden vollständig modelliert	5 %
	Abgeleitete Anforderungen der betrachteten Ebene wurden vollständig modelliert	5 %
	Strukturelle Relationen der betrachteten Ebene und hierarchische Relationen zur übergeordneten Ebene wurden vollständig modelliert	5 %
	Größen und technisch-physikalische Schnittstellen der betrachteten Ebene wurden vollständig modelliert	5 %

Die aufgeführten Bewertungskriterien setzen sich zusammen aus ihrem Bezug zur jeweiligen Schrittfolge, den erwarteten Ergebnissen sowie einer Gewichtung. Die Gewichtung wurde so gewählt, dass die vollständige Durchführung der Schrittfolgen annähernd gleich bedeutsam ist, während die Systemmodellierung als Hilfsmittel eine geringere Gewichtung erhält. Die erwarteten Ergebnisse werden dabei entweder mit 5 % oder 10 % gewichtet. Auf die Bewertungskriterien der Schrittfolge zur Konkretisierung des Lösungsraums entfallen somit 40 %, auf die

Bewertungskriterien der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse 35 % und auf die Bewertungskriterien zur Systemmodellierung 25 %.

Die Bewertungskriterien mit Bezug zur Konkretisierung des Lösungsraums stellen mit 40 % die am höchsten gewichtete Klasse dar, was in der generellen Bedeutung der erwarteten Ergebnisse begründet ist. Die sukzessive Systemkonkretisierung bildet den zentralen Bestandteil der funktionsorientierten Produktentwicklung. Eine Binnendifferenzierung der enthaltenen Schritte ist jedoch nicht zielführend, da die Umsetzung jedes erwarteten Ergebnisses die Qualität des Systementwurfs im gleichen Maße beeinflusst.

Die Bewertungskriterien mit Bezug zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse stellen mit einer Bedeutung von 35 % die zweitwichtigste Anforderungsklasse dar. Dies ist darin begründet, dass die definierten erwarteten Ergebnisse über die zielgerichtete Bestimmung und Analyse von Relationen und Schnittstellen ein wesentliches Merkmal der funktionsorientierten Produktentwicklung sind und den Kern des Vorgehenskonzepts bilden.

Die Bewertungskriterien mit Bezug zur Systemmodellierung stellen mit einem Anteil von 25 % am erreichbaren Produktreifegrad den am niedrigsten bewerteten Part dar. Die Systemmodellierung ist ein Hilfsmittel in der Produktentwicklung, um den Entwicklungsfortschritt abzubilden und hierüber Transparenz und Eindeutigkeit zu erhöhen. Gleichzeitig kann über das Systemmodell die Nachweisführung erfolgen, dass die zuvor definierten erwarteten Ergebnisse generiert wurden. Allerdings ließen sich die erwarteten Ergebnisse der anderen Bewertungskriterien theoretisch ohne eine entwicklungsbegleitende Systemmodellierung erreichen, da das Systemmodell zwar Transparenz schafft, jedoch im Kern die Systemkonkretisierung nicht vorantreibt, sondern lediglich unterstützt. Die Analyse von Herausforderungen bei der Entwicklung technischer Systeme hat jedoch gezeigt, dass methodische Defizite in der Produktentwicklung bestehen, welche das Auftreten von Problemen begünstigen. Somit kann eine Systemmodellierung einen entscheidenden Vorteil bieten, um komplexe Produkte anforderungsgerecht zu entwickeln.

Da bei der Bewertung des Produktreifegrads die Erfüllung aller erwarteten Ergebnisse in den klassierten Bewertungskriterien binär beantwortet werden kann, existieren keine abgestuften Bewertungen. Demnach liegen die erwarteten Ergebnisse entweder vor oder nicht vor, was zu einer entsprechenden Bewertung des Produktreifegrads führt. Der erreichte Produktreifegrad je Ebene ergibt sich aus der Addition der Gewichtung der einzelnen Bewertungskriterien und liegt somit zwischen 0 % und 100 %.

Zeigt die Bewertung des Reifegrads, dass erwartete Ergebnisse für die Konkretisierung des Lösungsraums zwar nicht vorliegen, jedoch auch nicht benötigt werden, ist dies ein Indikator für die Übergabe des Systementwurfs in die Phase des domänenspezifischen Entwurfs oder einer unvollständigen Konkretisierung des Lösungsraums auf der übergeordneten Ebene. Ersteres trifft zu, wenn sich das System auf der betrachteten Ebene nicht sinnvoll über Funktionen detaillierter konkretisieren lässt und somit keine zusätzlichen Funktionen und analog hierzu keine weiteren funktionsrealisierenden Elemente definiert würden. Eine weitere Systemkonkretisierung über Funktionen ist in diesem Fall oftmals nicht möglich, ohne dass diese bereits einen domänenspezifischen Entwurf vorwegnimmt. Die zweite mögliche Ursache ist die unvollständige Festlegung funktionsrealisierender Elemente auf der übergeordneten Ebene. Hierbei

würden zunächst weitere Festlegungen von Komponenten auf der betrachteten Ebene erforderlich werden, bevor die Funktionsstruktur granularer beschrieben werden kann.

Im Unterschied zum konkreten Produktreifegrad kann der generische Prozessreifegrad nicht durch eine Bewertung des Systementwurfs erfasst werden, da er direkt durch das Vorgehenskonzept umgesetzt wird. Stattdessen stellt das Vorgehenskonzept mit den Schrittfolgen zur Konkretisierung der Lösungsebenen, zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse sowie der parallelen Systemmodellierung den zu bewertenden Prozess dar. Die Bewertung des Prozessreifegrads erfolgt somit generisch, d.h. unabhängig vom konkreten Entwicklungsvorhaben und ausschließlich auf Basis des im Vorgehensmodell vorgestellten Prozesses. Die Bewertung des Prozessreifegrads des Vorgehenskonzepts erfolgt demnach bei der Bewertung des Ansatzes im Anschluss an die Validierung, siehe Kap 5.5.2.

5 Validierung

Die Validierung des vorgestellten Vorgehenskonzepts erfolgt zum Zweck des vollständigen Durchlaufs der Schrittfolgen und zur Bewertung der Eignung des Ansatzes sowie zur Erfassung des generischen Prozessreifegrads. Hierfür werden anhand eines Beispielsystems die Schrittfolgen zur sukzessiven Konkretisierung des Lösungsraums und zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse angewendet, wobei eine parallele Systemmodellierung erfolgt, um Ergebnisse zu visualisieren. Zu den jeweiligen Meilensteinen wird die Schrittfolge zur Ermittlung des Produktreifegrads durchlaufen.

Die Validierung dient der Veranschaulichung und Bewertung des Vorgehenskonzepts. Die aufgeführten Elemente des Beispielsystems sowie deren Relationen und Schnittstellen werden nur soweit betrachtet und modelliert, wie sie für die Darlegung des Vorgehenskonzepts notwendig sind. Das exemplarisch modellierte System erhebt daher keinen Anspruch auf Vollständigkeit bezüglich der Funktions- oder der Komponentenstruktur. Die Visualisierung des Beispielsystems erfolgt in allen drei möglichen Formen, d.h. in der Matrizen-, Graphen- und Symboldarstellung, um die Vorteile und Grenzen der jeweiligen Modellierungsart aufzuzeigen.

Vor der eigentlichen Validierung wird zunächst das Beispielsystem, eine Linear-Asynchronmaschine, aus dem Projekt „Q-ELF“ vorgestellt. Gemäß dem Fokus des Vorgehenskonzepts wird von einer vollständigen und widerspruchsfreien Anforderungsliste ausgegangen, wobei für das Beispielsystem nur einzelne Anforderungen exemplarisch genannt werden.

5.1 Vorstellung des Validierungsbeispiels Q-ELF

Das Projekt „**Q**ualitätsorientierter Methodenworkflow für die Produktneu**e**ntwicklung eines **L**inearantriebs in der **F**ördertechnik“ (Q-ELF) wurde von der Deutschen Forschungsgesellschaft (DFG) von Oktober 2011 bis März 2014 gefördert und von Lehrstühlen „Produktsicherheit und Qualitätswesen“ der Bergischen Universität Wuppertal und „Elektrische Antriebe und Mechatronik“ der Technischen Universität Dortmund gemeinsam erarbeitet. Die Arbeitshypothese lautete, dass Herausforderungen in der Entwicklung komplexer technischer Produkte, hier eines Linearantriebs, prinzipiell durch die Anwendung eines Methodenworkflows beherrschbar sind und das Linearantriebe, deren Produktentwicklung methodisch und anforderungsorientiert unterstützt wird, geeignete Lösungen für spezielle intralogistische Anwendungsfälle darstellen können. Hieraus wurden vier Arbeitspakete mit neuen Forschungsfragen abgleitet, welche die Anforderungsanalyse von Linearantrieben in der Fördertechnik, die Anforderungspriorisierung und deren Wechselwirkungen, die methodische Unterstützung bei der Auslegung einer Linearantriebskonzepts sowie die Validierung des methodischen Ansatzes am zu entwickelnden Produkt umfassten. [Willing et al. 2014]

Die grundsätzliche Eignung bzw. die Vor- und Nachteile der verschiedenen Antriebskonzepte für die Intralogistik werden an dieser Stelle nicht betrachtet. Für die Validierung des Vorgehenskonzepts wird ausschließlich die im Projekt Q-ELF betrachtete Linearasynchronmaschine fokussiert. Diese wird nachfolgend zunächst bezüglich ihres Wirkprinzips und Aufbaus beschrieben, bevor die Schrittfolgen durchlaufen werden.

5.2 Linearasynchronmaschinen

Bei einer Linearasynchronmaschine (LASM) handelt es sich um eine Variante von Drehstromlinearantrieben (DLA). Im Gegensatz zu rotatorischen Antrieben wie Drehstromsynchron- oder Drehstromasynchronmaschinen wird bei DLA elektrische Energie zur Erzeugung einer translatorischen Bewegung genutzt. Das Funktionsprinzip von Linearmaschinen beruht auf einer abgewickelten Asynchronmaschine, d.h. Stator und Rotor sind nicht gewickelt, sondern befinden sich auf einer flachen Ebene und werden als Primärteil bzw. Sekundärteil bezeichnet. Durch Spannungsinduktion wird eine Wanderwelle erzeugt, wodurch statt einer rotatorischen Bewegung eine translatorische Bewegung entsteht. Linearantriebe können dabei im Langstator- oder Kurzstator-Prinzip ausgelegt werden. Beim Langstator-Prinzip ist der untere, immobile Teil (die „Fahrstrecke“) mit Drehstromwicklungen versehen, während beim Kurzstator-Prinzip der obere, mobile Bereich aktiv ist. [Marenbach et al. 2020, S. 119f; Wörner 2013, S. 83ff, Riekhof et al. 2012, S. 127ff]

Für das Projekt Q-ELF wurden Anforderungen an Antriebssysteme in intralogistischen Anlagen über eine Industriebefragung erhoben. Anschließend wurde die Anforderungserfüllung verschiedener Antriebskonzepte qualitativ verglichen. Dabei konnte sich die LASM in Langstatorauslegung gegenüber Linearsynchron- und Linearpermanentmagnetsynchronmaschinen durchsetzen. Aufgrund einer einfacheren Energieversorgung ohne Schleppkabel, wie bei der Kurzstatorauslegung, wurde die LASM in Langstatorauslegung für das Projekt ausgewählt und deren Entwicklung methodisch betrachtet. [Wörner et al. 2014]

Abb. 51: Vergleich einer LASM in Langstator- und Kurzstatorauslegung [Wörner et al. 2014]

Abb. 51 zeigt eine LASM im Langstator-Prinzip (links) und im Kurzstator-Prinzip (rechts). Während sich beim Langstator-Prinzip der Primärteil (engl. primary) unterhalb befindet und somit der Sekundärteil (engl. secondary), auf dem sich das Fördergut befindet, frei über die Förderstrecke bewegen kann, ist beim Kurzstator-Prinzip der Primärteil gleichzeitig der sich bewegende Part. Da diese Lösung zur Energieversorgung ein Schleppkabel erfordern würde, was für viele intralogistische Anwendungen nicht umsetzbar wäre, wurde das Langstator-Prinzip gewählt. Zudem zeigt die Abbildung die erforderlichen Nuten (engl. slots) für die Wicklungen im Primärteil.

5.3 Validierung des Vorgehenskonzept am Beispiel der LASM

Gemäß der Einordnung des Vorgehenskonzepts in den PEP wird für die Validierung vom Vorhandensein einer vollständigen Anforderungsbasis ausgegangen. Im Projekt Q-ELF wurden die Anforderungen vereinfacht bzw. idealisiert für ein intralogistisches System betrachtet. Hierbei sollte die Länge des Förderguts 400 mm und dessen Breite 300 mm bei einer maximalen Last von 50 kg betragen. Die Förderstrecke besitzt eine maximale Steigung von 30° und die Nenngeschwindigkeit des Förderguts auf dem Sekundärteil beträgt 2 m/s [Wörner 2013]. Bei den über die durchgeführte Industriebefragung priorisierten Anforderungen dominiert die Zuverlässigkeit des intralogistischen Systems noch vor der Bedeutung der konkreten

Einsatzbedingungen. Das Steuerkonzept, Wartung und Instandhaltung sowie Fertigung und Montage sind diesen untergeordnet, liegen in ihrer Bedeutung für Antriebshersteller sowie Hersteller und Nutzer intralogistischer Anlagen dennoch vor Gesetzen und Normen und der Leistung bzw. der Funktion. Weitere Anforderungen ordnen sich hierarchisch von den Herstellungskosten, dem Design, über Sicherheit und die Betriebskosten bis zum Recycling, welches in der Fördertechnik den geringsten Stellenwert einnimmt [Willing et al. 2014]. Diese Anforderungen bilden die technische Grundlage, auf denen das Vorgehenskonzept aufbaut.

5.4 Anwendung der Schrittfolgen

Für die Validierung wird zunächst die Schrittfolge zur Konkretisierung des Lösungsraums nullter Ebene durchlaufen und im Anschluss daran die Schrittfolge für die *n*-te Ebene. Die Schrittfolgen werden solange durchlaufen, bis der Systementwurf in die Phase des domänenspezifischen Entwurfs übergeben werden kann. Durch die Verzahnung mit der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse werden diese Schritte für jeden Schritt der Lösungsraumkonkretisierung durchlaufen, sofern sie hierfür vorgesehen sind. Parallel wird der Entwicklungsstand im Systemmodell gemäß den vorgestellten Konventionen zur Systemmodellierung in Graphen-, Matrizen- und Symboldarstellung, abhängig von der Eignung der Darstellungsform, abgebildet. Die Schrittfolge zur Ermittlung des Produktreifegrads wird zu jedem geforderten Meilenstein durchlaufen, d.h. jeweils nach dem Durchlauf der Schrittfolge zur Systemkonkretisierung und -analyse einer Lösungsebene. Das Ergebnis entscheidet darüber, ob die schrittweise Konkretisierung des Lösungsraums fortgesetzt werden kann oder ob aufgrund einer zu geringen Produktreife einzelne Schritte der betrachteten Ebene wiederholt werden müssen.

Generell sieht das Vorgehenskonzept eine entwicklungsbegleitende, detaillierte Identifizierung und Analyse von Systemelementen, Relationen, Größen und Schnittstellen vor. Inhaltlich ist hierfür die Expertise des Entwicklungsteams erforderlich. Das aufgezeigte System sowie dessen Relationen, Größen und Schnittstellen sind daher exemplarisch und zur Validierung des Vorgehenskonzepts zu verstehen und erheben keinen Anspruch auf Vollständigkeit.

5.4.1 Nullte Lösungsebene

Die nullte Lösungsebene stellt gegenüber den *n*-ten Ebenen eine Besonderheit dar, da ausschließlich hier die Ableitung einer nullten Anforderung erfolgt. Gleichzeitig endet die nullte Ebene bereits mit der Festlegung der Komponente. Zudem erfolgt keine Bewertung des Produktreifegrads nach Abschluss der Schrittfolge der nullten Ebene.

5.4.1.1 Anforderung nullter Ebene definieren

Basierend auf den dargestellten Anforderungen an das intralogistische System muss zunächst die Anforderung nullter Ebene definiert werden, welche den Zweck des Systems in abstrahierter Form beschreibt. Für die LASM besteht der Systemzweck in dem Transport von Fördergut, in diesem Fall Stückgut. Die Anforderung lautet daher „Stückgut befördern“. Sie stellt die maximale Lösungsneutralität sicher, da keinerlei Vorgaben über die Umsetzung enthalten sind. Demzufolge könnten auch intralogistische Systeme mit rotatorischen Antrieben oder anderen Linearantrieben als LASM die Anforderung erfüllen.

Parallel hierzu wird die Anforderung nullter Ebene modelliert. In der Symboldarstellung wird die Anforderung durch eine Ellipse dargestellt und enthält die Symboldarstellung die Angabe der Lösungsebene, vgl. Abb. 52. Entsprechend der in Kap. 4.2.1.3 vorgestellten Nomenklatur wird die nullte Anforderung als erstes Element der nullten Ebene als *A_0.1_Stückgut befördern* bezeichnet.

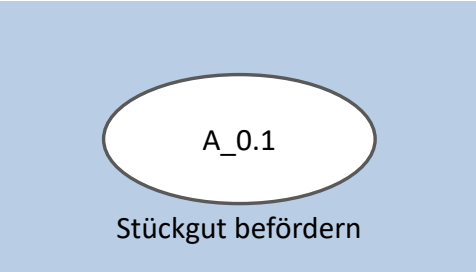


Abb. 52: Symboldarstellung der Anforderung nullter Ebene

Für die Graphen- und Matrizendarstellung in LOOMEO müssen zunächst sogenannte Domänen angelegt werden, welche den Systemsichten entsprechen. Gemäß dem verwendeten angepassten DeCoDe-Grundschema werden die drei Domänen bzw. Systemsichten Anforderung, Funktion und Komponente angelegt. Für jede Domäne muss mindestens ein Datensatz vorhanden sein. Datensätze beschreiben in LOOMEO Systemmatrizen und müssen vor der Modellierung von Elementen angelegt werden. Für das Vorgehenskonzept wird der erste Datensatz immer in der DSM angelegt, für Anforderungen daher in der Systemmatrix S_A. Da für jeden Datensatz eine Kanteninterpretation hinterlegt werden muss, wird für den ersten Datensatz der DSM, d.h. S_A, S_F oder S_K, immer die Kanteninterpretation Hierarchie hinterlegt, da nur diese Relation bei allen DSM existiert. Entsprechend wird der Datensatz für die Matrix S_{A_H} angelegt, siehe Abb. 53. Dieser enthält den Namen der Systemmatrix und die enthaltene Relation als Akronym gemäß der in Kapiteln 4.2.1.3 und 4.2.1.4 vorgestellten Konvention zur Benennung von Systemmatrizen und dargestellten Relationen. Im Kommentarfeld ist diese Information vollständig aufgeführt. Zudem enthält der Datensatz Informationen zur verknüpften Domäne und der modellierten Beziehung im Feld Kanteninterpretation, in diesem Fall „ist Bestandteil von“ für eine Hierarchierelation.

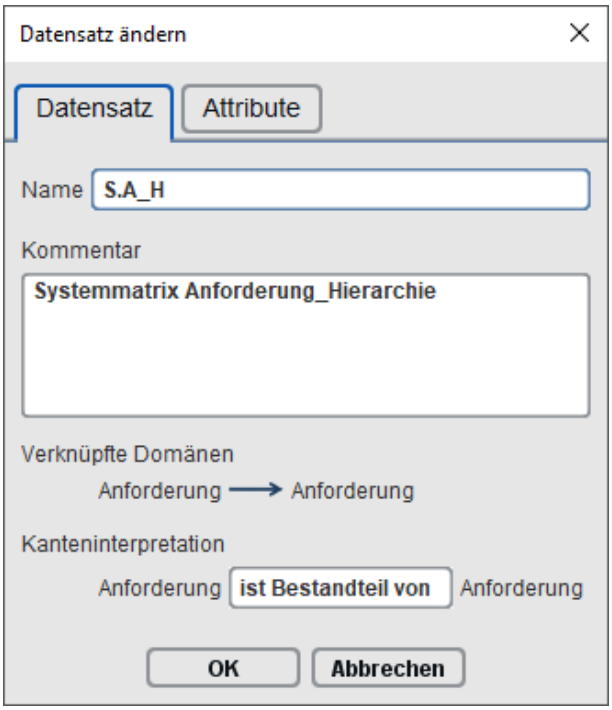


Abb. 53: Datensatz der Matrix S_{A_H}

Grundsätzlich können in LOOMEO für Domänen beliebig viele Datensätze angelegt werden. Im Rahmen der Arbeit werden maximal 16 Datensätze für die Systemmatrizen mit ihren jeweiligen strukturellen Relationen und Schnittstellen genutzt. Innerhalb eines Datensatzes können nun Systemelemente modelliert werden. Abb. 54 zeigt hierfür die Graphen- und die Matrizendarstellung der Anforderung nullter Ebene A_0.1 (Stückgut befördern). Da die Systemmatrix nur ein Element enthält, besteht die Graphendarstellung aus nur einem Knoten und in der Matrizendarstellung wird die Anforderung sich selbst gegenübergestellt. Da Elemente nicht mit sich selbst wechselwirken, kann keine Relation hinterlegt werden. Die Diagonale einer DSM ist somit immer leer.

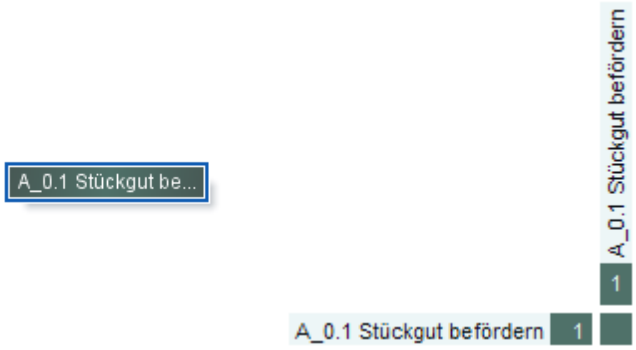


Abb. 54: Graphen- und Matrizendarstellung der Matrix S_{A_H} nullter Ebene

5.4.1.2 Funktion nullter Ebene definieren

Die Funktion nullter Ebene bildet durch die technische Beschreibung des Systemzwecks auf Basis der definierten Anforderungen das funktionelle System auf abstrakter Ebene ab. Obwohl bereits durch das Projekt eine Vorauswahl bezüglich des zu verwendenden Antriebs (LASM in Langstatorauslegung) des intralogistischen Systems getroffen wurde, kann die Funktion nullter Ebene lösungsneutral definiert werden. Entsprechend wird die Funktion definiert als „Wandeln

von elektrischer Leistung *P* in translatorische Kraft *F*“. Mit dieser Funktion ist lediglich festgelegt, dass das Fördersystem mittels elektrischer Energie betrieben und eine translatorische Bewegung durch eine zu erzeugende Kraft generiert werden soll. Die Funktion ist dennoch bezogen auf den durch das Projekt definierten Lösungsraum lösungsneutral, da zu ihrer Realisierung neben DLA auch rotatorische Antriebe genutzt werden könnten, welche Kraftumwandlungselemente nutzen, um die rotatorische in eine translatorische Bewegung zu wandeln.

Bei der Symboldarstellung als Teil der Systemmodellierung erfolgt die Modellierung über ein Rechteck, in dem das Symbol der Funktionskategorie Wandeln abgebildet ist, siehe Abb. 55. Neben der Funktionsbeschreibung enthält die Symboldarstellung die Bezeichnung F_0.1 für die erste Funktion nullter Ebene. Eingangs- und Ausgangsgrößen wurden noch nicht erfasst.

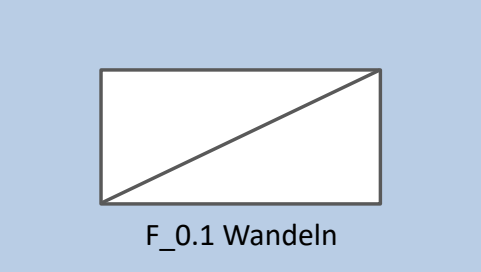


Abb. 55: Symboldarstellung der Funktion nullter Ebene

In LOOMEO wird zunächst ein Datensatz für die Domäne Funktion erstellt. Der in Abb. 56 beispielhaft dargestellte Datensatz gehört zur Matrix S_{F_H} und kennzeichnet somit für die Systemmatrix Funktion die Relationsart Hierarchie. Die Bezeichnung des zugehörigen Datensatzes der Matrix ist in LOOMEO hinterlegt. Neben der Bezeichnung sind hierbei die Art der DSM sowie die Kanteninterpretation der Relationsart hinterlegt.

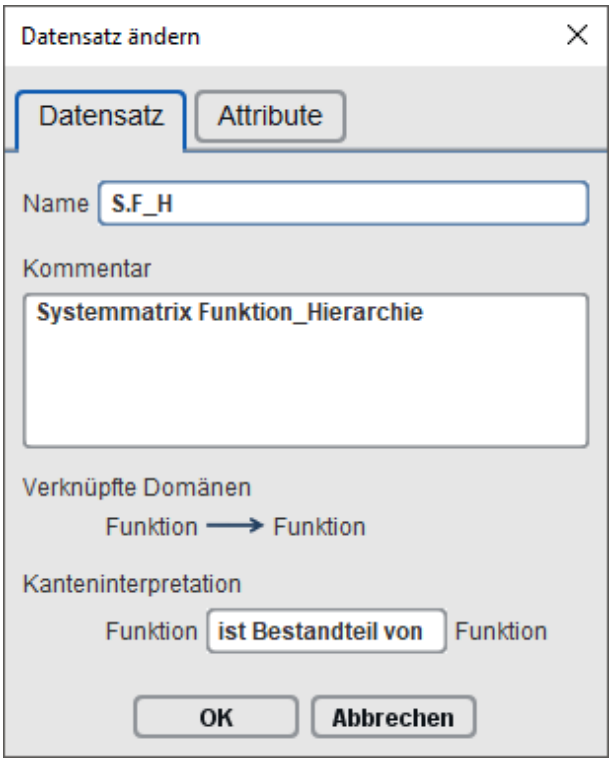


Abb. 56: Datensatz der Matrix S_{F_H}

In der Graphen- und Matrizendarstellung in Abb. 57 trägt der Knoten die Bezeichnung F_0.1, was die erste Funktion nullter Ebene kennzeichnet. Da bislang keine weiteren Elemente und somit keine weiteren Knoten existieren, entfallen in der Graphendarstellung etwaige Kanten für modellierbare Hierarchierelationen zwischen Funktionen. Diese werden erst dargestellt, sobald mindestens ein weiterer Knoten existiert, der über eine Relation mit der dargestellten Funktion verbunden wäre. Für das Systemmodell wird die Funktion über die DSM S_{F_H} für hierarchische Relationen von Funktionen dargestellt. Das Element wird dafür auf beiden Achsen der Matrix aufgetragen, wobei die Diagonale, d.h. der Einfluss eines Elements auf sich selbst, in DSM generell nicht betrachtet wird. Entsprechend kann in der Diagonalen kein Eintrag erfolgen.

Abb. 57: Graphen- und Matrizendarstellung der Matrix S_{F_H} nullter Ebene

Die Funktion ist nun vollständig modelliert, sodass die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse durchlaufen werden kann. Hierbei besitzen zwei Schritte für die Funktion nullter Ebene Relevanz: die Identifizierung von Relationen und die Identifizierung von Größen und Schnittstellen. Eine Vererbung von Elementen, Größen und Beziehungen übergeordneter Ebene ist für die nullte Ebene generell nicht möglich, da keine übergeordnete Ebene existiert. Zudem existieren für Funktionen keine Störgrößen, sodass eine Analyse von Größen und Schnittstellen nicht erforderlich ist.

5.4.1.2.1 Relationen identifizieren

Bei der Identifizierung von Relationen zeigt sich, dass die einzige Relation, welche für Funktionen der nullten Ebene besteht, die Realisierung ist. Da Funktionen Anforderungen realisieren, besteht zwischen der Funktion F_0.1 (Wandeln von elektrischer Leistung *P* in translatorische Kraft *F*) und der Anforderung A_0.1 (Stückgut befördern) eine Realisierungsrelation. Hierfür wird zunächst die Matrix S_{F,A_R} angelegt, in der alle Realisierungsrelationen zwischen Funktionen und Anforderungen beschrieben werden, siehe Abb. 58. Als Matrix zwischen zwei Sichten handelt es sich um eine DMM mit einer gerichteten Kanteninterpretation. Eine in LOOMEO

theoretisch beschreibbare entgegengesetzte Kanteninterpretation wird im Rahmen des Vorgehenskonzepts generell nicht genutzt.

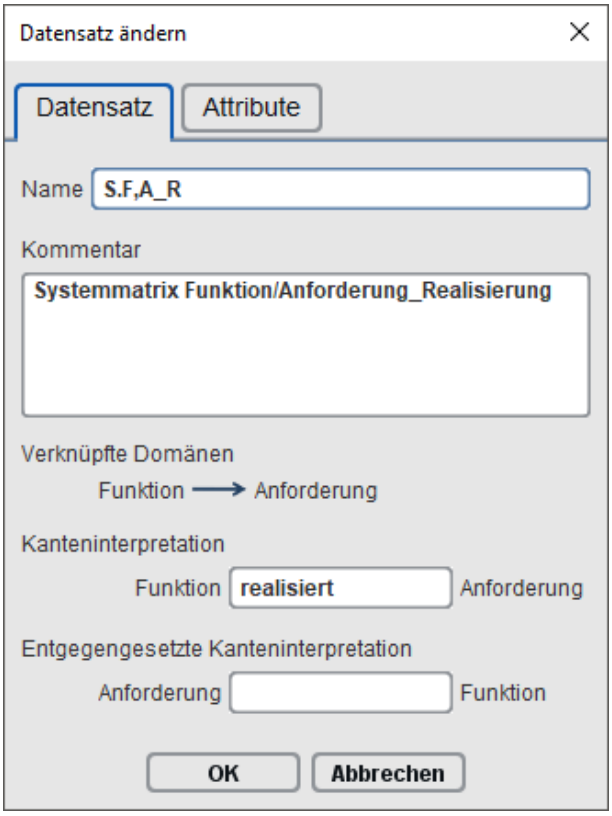


Abb. 58: Datensatz der Matrix S_{F,A_R}

Die Relation zwischen der Funktion F_0.1 und der Anforderung A_0.1 wird anschließend modelliert und kann sowohl in der Symboldarstellung als auch in LOOMEO in der Graphen- und Matrizendarstellung abgebildet werden, wobei in der Abb. 59 exemplarisch die Matrizendarstellung gewählt wurde. Hier wird die gerichtete Relation von der Funktion, welche in der Zeile aufgeführt ist, zur Anforderung, welche in der Spalte steht, in der Diagonalen der Matrix mittels eines gesetzten „X“ dargestellt.

Abb. 59: Matrizendarstellung der Matrix S_{F,A_R} nullter Ebene

5.4.1.2.2 Größen und Schnittstellen identifizieren

Die Identifizierung von Größen beschränkt sich für die nullte Ebene auf die Funktion F_0.1 „Wandeln von elektrischer Leistung *P* in translatorische Kraft *F*“, da Anforderungen keine technisch-physikalischen Größen und Schnittstellen besitzen. Wie aus der Funktionsbeschreibung hervorgeht, wird elektrische Leistung in translatorische Kraft gewandelt. Entsprechend

handelt es sich bei der elektrischen Leistung *P* um die Eingangsgröße und bei der translatorischen Kraft *F* um die Ausgangsgröße der Funktion. Diese sind bei der Identifizierung von Größen und Schnittstellen zu erfassen. Da beide Größen nicht spezifiziert sind, handelt es sich um qualitative Größen. Eine Verknüpfung der Größen in Form von Schnittstellen zwischen Systemelementen existiert für die Funktion F_0.1 nicht.

Für die Unterstützung bei der Identifizierung der Größen bietet sich die Symboldarstellung an, da hier die Erfassung von Größen separat für jedes Systemelement sowie die Erfassung von Schnittstellen über die Verknüpfung zwischen Systemelementen erfolgen kann. Dies wird in Abb. 60 dargestellt. Die Kreise links und rechts neben dem Funktionssymbol stehen für die Eingangs- bzw. die Ausgangsgröße und die identifizierten Größen *P* und *F* stehen über diesen.

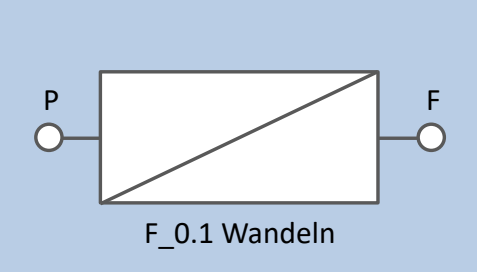


Abb. 60: Symboldarstellung der Funktion nullter Ebene mit Eingangs- und Ausgangsgröße

Mit dem Durchlaufen der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse sowie der parallelen Systemmodellierung ist der Schritt zur Definition der Funktion nullter Ebene abgeschlossen.

5.4.1.3 Komponente nullter Ebene definieren

Basierend auf der Funktion nullter Ebene muss eine geeignete Komponente zur Realisierung der Funktion festgelegt werden. Aufgrund der Festlegung im Projekt Q-ELF wird die Komponente nullter Ebene als LASM definiert. Damit schränkt die Komponente den Lösungsraum stärker ein, als es die Funktion vorgibt, da hier auch rotatorische Antriebe mit Kraftumwandlungselementen als mögliche Realisierung denkbar gewesen wären. Die Einschränkung erfolgt jedoch bewusst, um keine unnötigen Lösungsebenen zu generieren, über die das System von einer intralogistischen Anlage zu einer LASM in Langstatorauslegung konkretisiert werden müsste, da über das Forschungsprojekt bereits eine Festlegung auf diesen Antrieb erfolgte.

Die Komponente nullter Ebene lässt sich im Systemmodell über die Symboldarstellung abbilden, siehe. Abb. 61 . In der Symboldarstellung ist die LASM als Rechteck mit gerundeten Ecken dargestellt und wird als K_0.1 für die erste Komponente nullter Ebene bezeichnet.

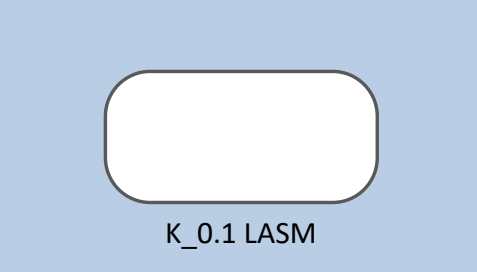


Abb. 61: Symboldarstellung der Komponente nullter Ebene

Parallel wird ein Datensatz für die Komponente angelegt. Wie bereits für die Funktion nullter Ebene aufgezeigt, wird als erster Datensatz immer eine DSM zur Abbildung einer hierarchischen Relation beschrieben, siehe Abb. 62. Der Datensatz für die Systemmatrix S_{K_H} enthält die vollständige Beschreibung der hinterlegten Relation sowie die Kanteninterpretation.

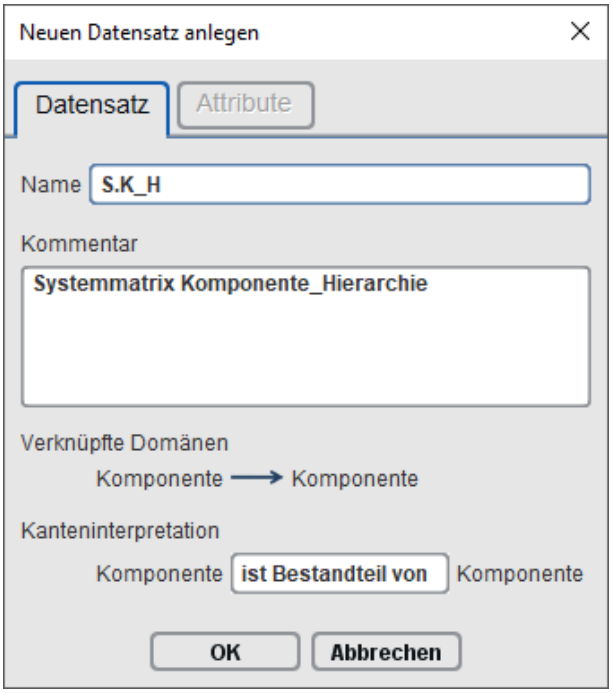


Abb. 62: Datensatz der Matrix S_{K_H}

Abb. 63 zeigt die zugehörige Graphen- und Matrizendarstellung der Komponente. Wie bereits erläutert und im Beispiel dargestellt, können Komponenten insbesondere auf höherer Lösungsebene als Systeme zusammengefasst werden. Diese werden dann auf niedrigerer Lösungsebene über hierarchische Relationen mit dem System verknüpft und granularer beschrieben bzw. konkretisiert. Entsprechend wird die Komponente nullter Ebene als LASM bezeichnet, da diese Beschreibung die größtmögliche Abstraktion des konkretisierten Systems ermöglicht.

Abb. 63: Graphen- und Matrizendarstellung der Matrix S_{K_H} nullter Ebene

Die anschließende Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse wird für die Komponente nullter Ebene erstmalig vollständig durchlaufen.

5.4.1.3.1 Elemente, Größen und Beziehungen vererben

Da keine übergeordnete Ebene existiert, werden keine Elemente, Größen oder Beziehungen auf die nullte Ebene vererbt. Per Definition müssen Komponenten als funktionsrealisierende Elemente jedoch mindestens über die gleichen Eingangs- und Ausgangsgrößen verfügen. Entsprechend sind die Größen der Funktion F_0.1 auf die Komponente K_0.1 zu vererben. Abb. 64

zeigt die LASM (K_0.1) mit den von der Funktion nullter Ebene (F_0.1) vererbten Größen, der elektrischen Leistung *P* und der translatorischen Kraft *F* in der Symboldarstellung.

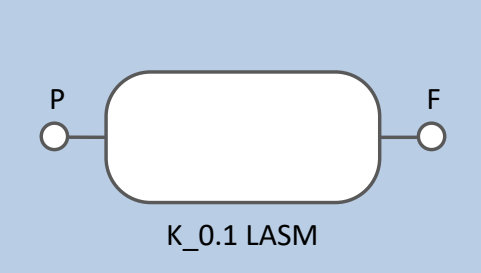


Abb. 64: Symboldarstellung der Komponente K_0.1 mit vererbten Eingangs- und Ausgangsgrößen

5.4.1.3.2 Relationen identifizieren

Mögliche Relationen zwischen Komponenten und anderen Systemelementen können gemäß Definition zwischen Komponenten und Anforderungen ($S_{K,A}$), Komponenten und Funktionen ($S_{K,F}$) sowie innerhalb der Komponenten-DSM bestehen (S_K), vgl. Abb. 35. Für die LASM existiert keine direkte Relation zur Anforderung A_0.1, welche über eine Realisierungs- oder Erfordernisrelation beschreibbar wäre. Zwar trägt die LASM zur Realisierung der Anforderung nullter Ebene bei, allerdings nur indirekt über die Realisierung der Funktion nullter Ebene F_0.1. Diese ist somit mit der Komponente K_0.1 über eine Realisierungsrelation zu verknüpfen. Hierfür wird zunächst die Systemmatrix $S_{K.F\_R}$ angelegt (vgl. Abb. 65), über die sich Realisierungsrelationen zwischen Komponenten und Funktionen beschreiben lassen.

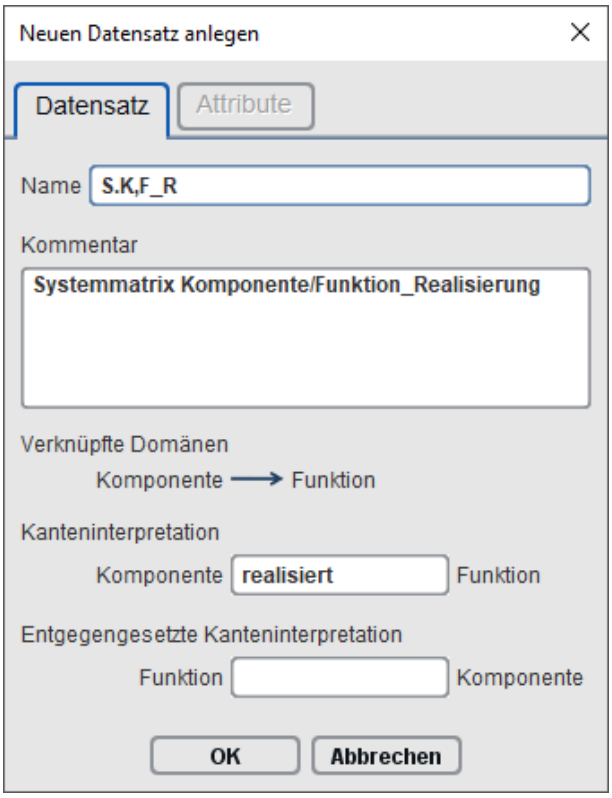


Abb. 65: Datensatz der Matrix S_{K,F_R}

Die Realisierungsrelation zwischen der Komponente K_0.1 und der Funktion F_0.1 wird in der Systemmatrix S_{K,F_R} modelliert. Abb. 66 zeigt diese Relation in der Matrizendarstellung, bei

der aus der Kanteninterpretation und der Auftragung der Komponente in der Zeile und der Funktion in der Spalte hervorgeht, dass die Komponente K_0.1 die Funktion F_0.1 realisiert.

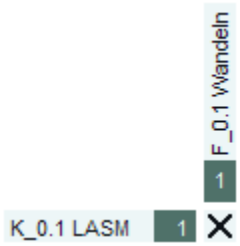


Abb. 66: Matrizendarstellung der Matrix S_{K,F_R} nullter Ebene

5.4.1.3.3 Größen und Schnittstellen identifizieren

Für die Komponente K_0.1 existieren bereits die Eingangsgröße *P* und die Ausgangsgröße *F*, welche durch die Funktion F_0.1 auf sie vererbt wurden. Auf der nullten Ebene lassen sich erwartungsgemäß keine zusätzlichen Größen identifizieren, welche für die Komponente K_0.1 zu berücksichtigen wären. Ebenso wenig existieren Schnittstellen, da diese ausschließlich zwischen Elementen gleicher Systemsicht gebildet werden dürfen und die Komponente K_0.1 das einzige Element dieser Sicht auf der nullten Ebene bildet.

5.4.1.3.4 Größen und Schnittstellen analysieren

Die elektrische Leistung *P* und die translatorische Kraft *F* bilden die einzigen Eingangs- und Ausgangsgrößen der Komponente K_0.1. Da nur eine Komponente nullter Ebene vorliegt, existieren keine Schnittstellen, sondern nur Größen. Schnittstellen können demzufolge nicht analysiert werden. Dies veranschaulicht den Grund für den reduzierten Umfang der Schrittfolge zur Konkretisierung des Lösungsraums: beim System nullter Ebene handelt es sich um ein idealisiertes System ohne Störgrößen, weshalb die Durchführung der Schritte „Abgeleitete Anforderung definieren“ und „Optimierungsmaßnahmen implementieren“ hierfür entfällt.

Eine Ermittlung des Produktreifegrads ist für die nullte Lösungsebene nicht vorgesehen. Dementsprechend ist die Schrittfolge abgeschlossen.

5.4.2 Erste Lösungsebene

Basierend auf der übergeordneten nullten Ebene ist die Schrittfolge für alle nun folgenden Lösungsebenen identisch. Die Schrittfolge zur Konkretisierung des Lösungsraums wird vollständig durchlaufen, beginnend mit der Definition der Funktion. Für jeden Schritt wird die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse durchlaufen, wobei für die Definition der Funktion, die Definition der abgeleiteten Anforderung und die Implementierung von Optimierungsmaßnahmen nicht alle Schritte relevant sind und somit entfallen. Die Schrittfolge zur Ermittlung des Produktreifegrads bildet den Abschluss der Lösungsebene.

5.4.2.1 Funktion definieren

Basierend auf der Funktion F_0.1 „Wandeln von elektrischer Leistung *P* in translatorische Kraft *F*“ muss die Funktionsstruktur konkretisiert werden. Dies erfordert einen tieferen Einstieg in die Wirkprinzipien von induktiv wirkenden LASM in Langstatorauslegung.

Um eine resultierende Kraft zu erzeugen, muss ein elektrisches Wanderfeld $\underline{B}_{Wander}$ erzeugt, geleitet und in eine Kraft *F* gewandelt werden. Die bisherige Funktion F_0.1 muss entsprechend

aufgeteilt bzw. erweitert werden, um die erweiterte Funktionsstruktur abzubilden. Auf diese Weise resultieren auf der ersten Lösungsebene die Funktionen F_1.1 „Wandeln von elektrischer Leistung *P* in Wanderfeld $\underline{B}_{Wander}$“, F_1.2 „Leiten des Wanderfelds $\underline{B}_{Wander}$“ und F_1.3 „Wandeln des Wanderfelds $\underline{B}_{Wander}$ in translatorische Kraft *F*“. Dabei wird bereits der Forderung Rechnung getragen, dass sich die Kategorie einer Funktion übergeordneter Ebene auf der untergeordneten Ebene bei der Konkretisierung dieser Funktion wiederfinden muss.

5.4.2.1.1 Elemente, Größen und Beziehungen vererben

Von der Funktion F_0.1 sind für die erste Ebene die Funktionskategorie (Wandeln), die Eingangsgröße *P* (elektrische Leistung) sowie die Ausgangsgröße *F* (translatorische Kraft) zu übernehmen. Darüber hinaus existieren keine Größen, Elemente oder Beziehungen, welche auf die Funktionen erster Ebene zu vererben sind. Die vererbten Größen werden daher auf die konkretisierten Funktionen der ersten Ebene übertragen, d.h. die Eingangsgröße auf die erste konkretisierte Funktion und die Ausgangsgröße auf die letzte konkretisierte Funktion der ersten Ebene, vgl. hierzu Kap. 4.2.1.6 zur Vererbung von Größen. Alle Schnittstellen zwischen den Funktionen sind im übernächsten Schritt zu bestimmen.

5.4.2.1.2 Relationen identifizieren

Gemäß der Reihenfolge bei der Identifizierung werden zunächst hierarchische Relationen identifiziert. Diese bestehen zwischen den Funktionen der ersten Ebene (F_1.1, F_1.2, F_1.3) und der übergeordneten Funktion nullter Ebene (F_0.1). Folglich sind die Funktionen erster Ebene Bestandteil der Funktion nullter Ebene. Abb. 67 zeigt die hierarchischen Relationen zwischen den Funktionen in Graphen- und Matrizendarstellung in der Systemmatrix S_{F_H}.

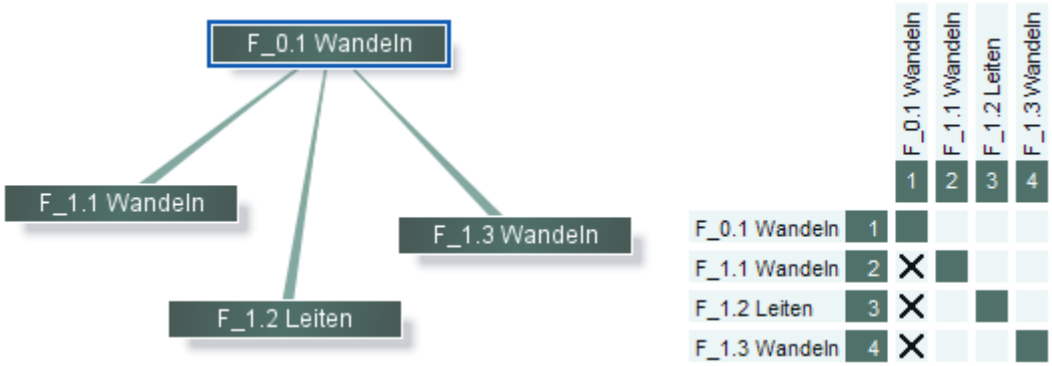


Abb. 67: Graphen- und Matrizendarstellung der Matrix S_{F_H} erster Ebene

Durch die Benennung der Systemelemente ist eine sichere Zuordnung zur Lösungsebene gewährleistet. Darüber hinaus kann die Beschreibung jedes Elements als Knotenkommentar in LOOMEO hinterlegt werden, in Abb. 68 dargestellt für die Funktion F_1.1.

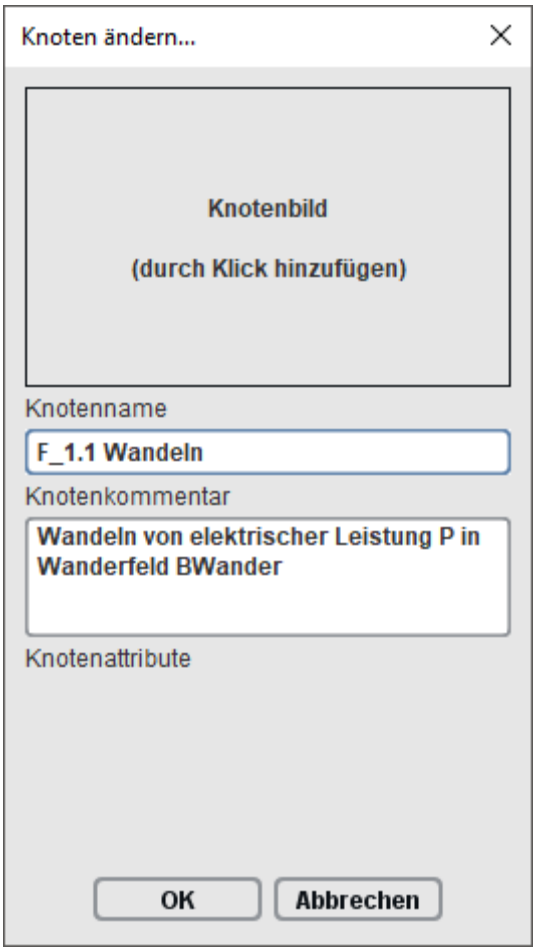


Abb. 68: Knotenkommentar zur Funktion F_1.1

Als weitere Relation kann eine Abfolge zwischen den Funktionen erster Ebene identifiziert werden. Hierbei muss zunächst elektrische Leistung *P* in ein Wanderfeld $\underline{B}_{Wander}$ gewandelt werden. Dies beschreibt die Funktion F_1.1. Anschließend muss das Wanderfeld geleitet (F_1.2) und in eine translatorische Kraft *F* gewandelt werden (F_1.3). Die Funktionen F_1.1, F_1.2 und F_1.3 stehen entsprechend in einer Kausalkette, bei der die Funktionen F_1.1, F_1.2 und F_1.3 sequenziell ablaufen. Diese Relation lässt sich über die Graphen- und Matrizendarstellung in der neu anzulegenden Systemmatrix S_{F_A} modellieren, vgl. Abb. 69. In der rechten Matrizendarstellung sind alle Elemente der Matrix S_{F_A} enthalten, d.h. neben den Funktionen der ersten Ebene auch die Funktion der nullten Ebene. In der linken Graphendarstellung sind ausschließlich die Funktionen der ersten Ebene vorhanden. Hintergrund ist, dass die Funktion F_0.1 zwar existiert, sie jedoch nicht über eine Abfolgerelation mit den Funktionen der ersten Ebene verknüpft ist und daher beim Graph nicht dargestellt wird.

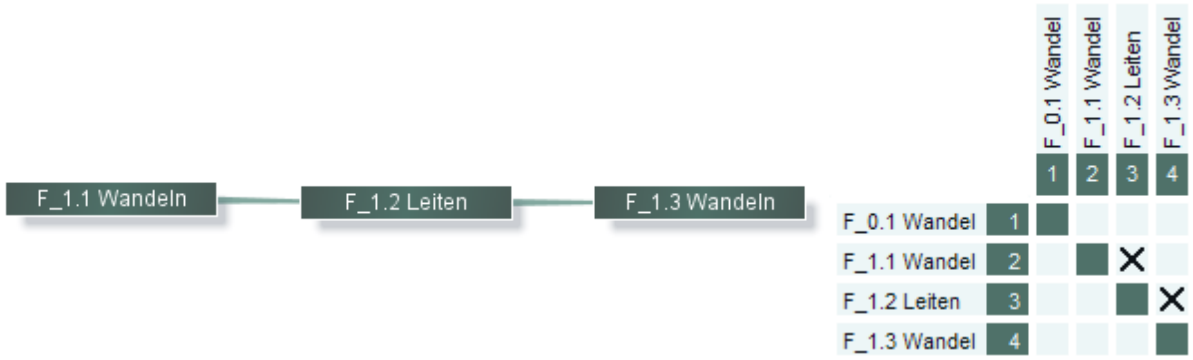


Abb. 69: Graphen- und Matrizendarstellung der Matrix S_{F_A} erster Ebene

Eine weitere mögliche Relationsart der Funktionssicht, d.h. eine Realisierung in der DMM S_{F,A_R}, existiert nicht, da sich eine Realisierungsrelation durch Funktionen nur auf Anforderungen beziehen kann und eine Realisierung nur innerhalb einer Lösungseben erfolgen darf. Da bislang keine Störgrößen von Komponenten identifiziert wurden, aus denen Anforderungen abzuleiten wären, gibt es folglich keine Realisierung abgeleiteter Anforderungen für die erste Lösungsebene.

5.4.2.1.3 Größen und Schnittstellen identifizieren

Die Eingangs- und Ausgangsgrößen der Funktionen F_1.1, F_1.2 und F_1.3, d.h. die elektrische Leistung *P*, das Wanderfeld $\underline{B}_{Wander}$ und die translatorische Kraft *F* müssen in diesem Schritt zugeordnet und modelliert werden. Über die Symboldarstellung kann dabei gleichzeitig die Identifizierung und Modellierung der Schnittstellen erfolgen. Die Verknüpfung der Funktionen über ihre Eingangs- und Ausgangsgrößen zu Schnittstellen entspricht einem Energiefluss, sodass Abb. 70 die Matrix S_{F_EF} darstellt. Die Schnittstellen der Funktionen sind bezüglich ihrer Struktur identisch mit der Abfolgerelation, die in Abb. 69 dargestellt ist. Dies ist eine typische, aber nicht zwingende Analogie zwischen der Abfolgerelation und dem Energiefluss zwischen Funktionen.

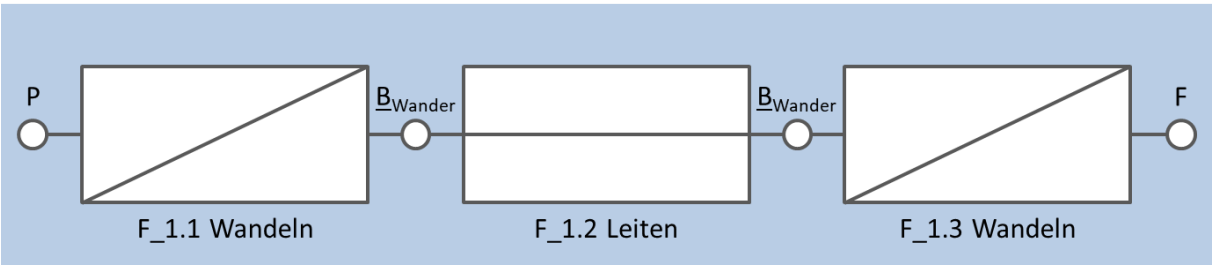


Abb. 70: Symboldarstellung der Größen und Schnittstellen der Funktionen erster Ebene

Da keine weiteren Größen oder Schnittstellen existieren und vorhandene Größen von Funktionen per Definition Sollgrößen darstellen, ist die Analyse von Größen und Schnittstellen nicht erforderlich und die Schrittfolge für die Definition der Funktionen abgeschlossen.

5.4.2.2 Komponente festlegen

Die Konkretisierung der Komponente nullter Ebene zeigt, dass eine LASM in Langstatorauslegung im Wesentlichen aus zwei physischen und einem logischen Systembestandteil besteht: einem Primärteil, einem Sekundärteil und dem Luftspalt zwischen diesen. Diese Bestandteile sind erforderlich, um die Funktionen der ersten Ebene zu realisieren. Entsprechend werden die Komponenten als Primärteil K_1.1, Luftspalt K_1.2 und Sekundärteil K_1.3 bezeichnet.

5.4.2.2.1 Elemente, Größen und Beziehungen vererben

Über die Festlegung der Komponente wurde im vorherigen Schritt die Komponente K_0.1 auf der ersten Ebene konkretisiert und muss daher nicht auf die erste Ebene vererbt werden. Es müssen jedoch ihre Eingangs- und Ausgangsgröße auf die Komponenten der ersten Ebene vererbt werden, d.h. die elektrische Leistung *P* und die translatorische Kraft *F*. Da die Aufteilung der Komponente K_0.1 in die drei Komponenten K_1.1, K_1.2 und K_1.3 eine Systemkonkretisierung darstellt, vererben sich die Größen der nullten Ebene nicht direkt auf jede Komponente der ersten Ebene. Sie müssen aber auf die „Black Box“ der konkretisierten Komponenten erster Ebene übernommen werden. Da auf der nullten Ebene keine technisch-physikalischen Schnittstellen der Komponente K_0.1 existieren, werden keine Beziehungen auf die Komponenten erster Ebene vererbt.

Neben den von der Komponentensicht nullter Ebene vererbten Informationen müssen Größen, welche sich aus der Funktionsstruktur erster Ebene ergeben haben, auf die jeweils funktionsrealisierenden Komponenten K_1.1, K_1.2 und K_1.3 vererben werden. Da über die Komponentensicht nullter Ebene bereits die elektrische Leistung *P* und die translatorische Kraft *F* vererbt wurden, wird über die Funktionen erster Ebene nur die Größe des Wanderfelds $\underline{B}_{Wander}$ vererbt.

5.4.2.2.2 Relationen identifizieren

Als erste Relation werden hierarchische Abhängigkeiten identifiziert. Analog zu den Funktionen erster Ebene sind die Komponenten erster Ebene Bestandteil der Komponente nullter Ebene. Dies wird in der Systemmatrix S_{K_H} erfasst und lässt sich über die Matrizen- und Graphendarstellung modellieren, vgl. Abb. 71.

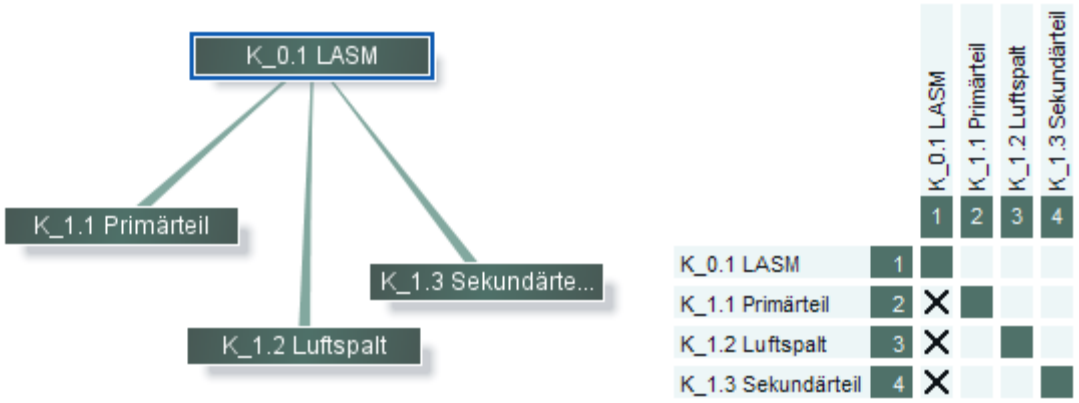


Abb. 71: Graphen- und Matrizendarstellung der Matrix S_{K_H} erster Ebene

Anders als für Funktionen existieren für Komponenten generell keine Abfolgerelationen. Die Realisierungsrelation beschreibt hingegen die Funktionsrealisierung durch die Komponenten und wird über die Systemmatrix S_{K,F_R} in Matrizendarstellung abgebildet (Abb. 72). Die Komponente K_1.1 (Primärteil) realisiert die Funktion F_1.1 (Wandeln von elektrischer Leistung *P* in Wanderfeld $\underline{B}_{Wander}$), während über die Komponente K_1.2 (Luftspalt) das Wanderfeld geleitet wird und damit die Funktion F_1.2 realisiert. Durch die Komponente K_1.3

(Sekundärteil) wird das Wanderfeld in die translatorische Kraft *F* gewandelt, wodurch die Funktion F_1.3 realisiert wird[30].

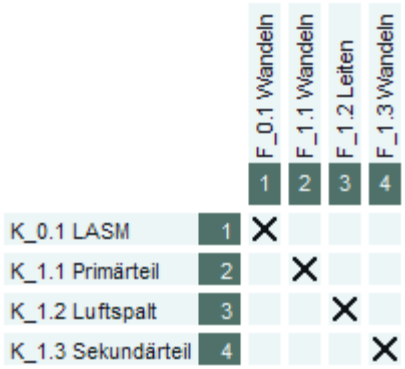


Abb. 72: Matrizendarstellung der Matrix S_{K,F_R} erster Ebene

Die in der Systemmatrix S_{K,F_R} ebenfalls aufgeführten Elemente der nullten Ebene, die Komponente K_0.1 und die Funktion F_0.1, sind nicht erforderlich, um Relationen der ersten Ebene zu beschreiben. Basierend durch die sukzessive Systemkonkretisierung wird die Matrix jedoch schrittweise gefüllt, sodass über mehrere Lösungsebenen in der Matrix Blöcke entstehen, welche sich eindeutig einer Lösungsebene zuordnen lassen. Abb. 73 veranschaulicht dies über einen blauen Block für der nullte Ebene und einen grünen Block für die erste Ebene. Zudem ist über die Indizierung der Elemente als K_x.y, wobei x die Lösungsebene und y die fortlaufende Nummer von Elementen der gleichen Systemsicht und Ebene beschreibt, jederzeit eine eindeutige und transparente Benennung gegeben. Im Umkehrschluss würde eine ebenenreine Matrix erfordern, dass beispielsweise die DMM $S_{K,F}$ je Ebene eine separate Systemmatrix benötigen würde, also $S_{K,F\_R.0}$ und $S_{K,F\_R.1}$ für die Systemmatrix zur Beschreibung einer Realisierungsrelationen zwischen Komponenten und Funktionen nullter bzw. erster Ebene. Dies würde den Modellierungsaufwand und die Komplexität erhöhen und die Übersichtlichkeit verringern.

[30] Bei einer integralen Konstruktion, bei der Funktionen mehrerer Lösungsebenen durch nur eine Komponente erfüllt werden, wird die integrale Komponente als Subsystem verstanden. Auch wenn das Subsystem somit in der Lage wäre, Funktionen zu realisieren, welche auf der Ebene, auf der das Subsystem erstmalig definiert wurde, noch gar nicht existieren, ändert sich nicht die prinzipielle Logik. Auf untergeordneter Ebene ließe sich das Subsystem granularer beschreiben, beispielsweise über Funktionsflächen, welche Bestandteil der integralen Komponente sind und sich nicht baulich, aber logisch weiter zerlegen lassen. Diese Funktionsflächen können dann auf untergeordneter Ebene mit den Funktionen über eine Realisierungsrelation verknüpft werden.

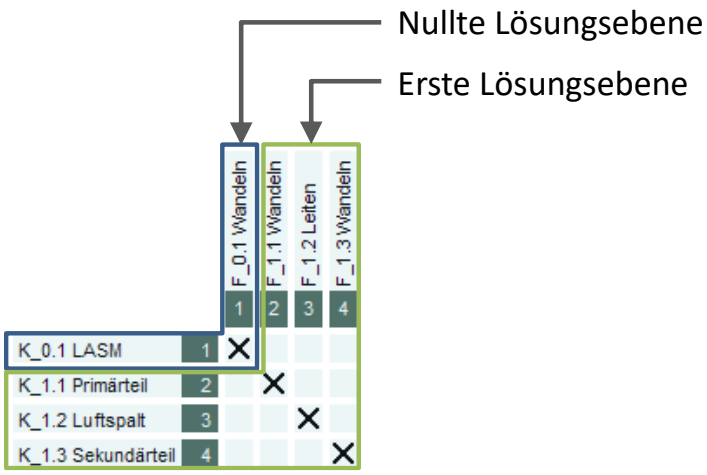


Abb. 73: Lösungsebenen in der Matrizendarstellung S_{K,F_R}

Für die Komponenten erster Ebene K_1.1, K_1.2 und K_1.3 liegt keine Erfordernisrelation vor, da bislang keine Störgrößen existieren, welche abgeleitete Anforderungen erfordern würde.

5.4.2.2.3 Größen und Schnittstellen identifizieren

Durch die Vererbung bestehender Größen und Schnittstellen der Funktionssicht erster Ebene auf die Komponenten erster Ebene und die identifizierte Realisierung der Funktionen F_1.1, F_1.2 und F_1.3 durch die Komponenten K_1.1, K_1.2 und K_1.3 existiert bereits eine Komponentenstruktur, welche die gleichen Größen und Schnittstellen besitzt wie die Funktionen der Ebene. Da keine zusätzlichen Größen oder Schnittstellen für die Komponenten erster Ebene identifiziert werden können, können die Energieflüsse direkt abgleitet und modelliert werden. Die Modellierung ist grundsätzlich in allen Darstellungsvarianten (Symbol, Graph, Matrix) möglich, wobei die Symboldarstellung (Abb. 74) den größtmöglichen Informationsumfang bietet, da sowohl die qualitativen Größen als auch die technisch-physikalischen Schnittstellen zwischen den Komponenten darstellbar sind.

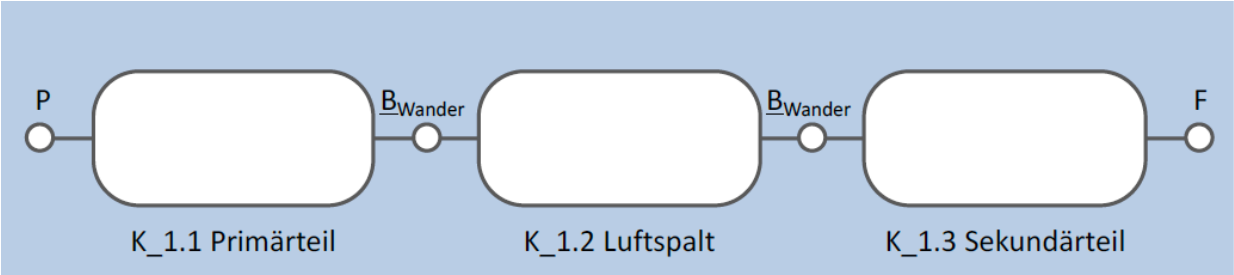


Abb. 74: Symboldarstellung der Größen und Schnittstellen der Komponenten erster Ebene

5.4.2.2.4 Größen und Schnittstellen analysieren

Die elektrische Leistung *P* stellt die erforderliche Eingangsgröße für die Funktionsrealisierung des Primärteils (Komponente K_1.1) dar, demzufolge handelt es sich um eine Sollgröße. Das erzeugte Wanderfeld $\underline{B}_{Wander}$ als Ausgangsgröße der Komponente K_1.1 und Eingangsgröße der Komponente K_1.2 (Luftspalt) ist ebenfalls erforderlich und verursacht keinen negativen Einfluss auf Systembestandteile. Somit wird die Größe als Sollgröße bestätigt. Auch für das Wanderfeld als Ausgangsgröße der Komponente K_1.2 und Eingangsgröße für die Komponente K_1.3 (Sekundärteil) gilt selbiges, entsprechend handelt es sich um eine Sollausgangs- bzw. Solleingangsgröße. Die generierte translatorische Kraft *F* stellt, wie bereits in der nullten

Ebene definiert, die Sollausgangsgröße der Komponente K_1.3 dar. Somit liegen keine Störgrößen oder problematische technisch-physikalische Schnittstellen vor.

5.4.2.3 Abgeleitete Anforderung definieren

Die Ableitung von Anforderungen als Reaktion auf Störgrößen entfällt, da keine Störgrößen identifiziert werden konnten. Dementsprechend werden für die abgeleiteten Anforderungen auch keine Schritte der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse durchlaufen.

5.4.2.4 Optimierungsmaßnahmen implementieren

Die Implementierung von Optimierungsmaßnahmen ist nicht erforderlich, da keine abgeleiteten Anforderungen existieren, welche diese Maßnahmen bedingen würden. Analog zur Definition abgeleiteter Anforderungen sind demnach keine Schritte zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse erforderlich.

5.4.2.5 Ermittlung des Produktreifegrads für die erste Ebene

Mit dem Abschluss des letzten Schrittes zur Systemkonkretisierung und -analyse der ersten Lösungsebene ist der erste Meilenstein erreicht, an dem die Schrittfolge zur Ermittlung des Produktreifegrads durchlaufen wird. Hierfür wird das in Kap. 4.2.3.2 vorgestellte Bewertungsschema genutzt, bei dem die Kriterien einzeln bewertet werden, siehe Tab. 19.

Tab. 19: Bewertung des Produktreifegrads erster Ebene

Schrittfolge	Erwartete Ergebnisse	Bewertung
Konkretisierung des Lösungsraums	Auf der betrachteten Ebene wurde mindestens eine zusätzliche Funktion im Sinne der Systemkonkretisierung definiert	Ja (10 %)
	Für jede Funktion der betrachteten Ebene wurde mindestens eine funktionsrealisierende Komponente festgelegt	Ja (10 %)
	Beim Auftreten von Störgrößen auf der betrachteten Ebene wurden Anforderungen zum Umgang mit diesen abgeleitet	n/a (10 %)
	Beim Vorliegen abgeleiteter Anforderungen wurden Optimierungsmaßnahmen auf der betrachteten Ebene implementiert	n/a (10 %)
Relationsbestimmung und Schnittstellenidentifizierung und -analyse	Funktionen und Komponenten wurden von der übergeordneten Ebene auf die betrachtete Ebene vererbt oder auf der betrachteten Ebene konkretisiert	Ja (10 %)
	Eingangs- und Ausgangsgrößen und Schnittstellen der übergeordneten Ebene wurden auf Elemente der betrachteten Ebene vererbt	Ja (5 %)
	Hierarchische Relationen von Elementen der betrachteten Ebene zu Elementen der übergeordneten Ebene wurden vollständig bestimmt	Ja (5 %)
	Strukturelle Relationen von Elementen der betrachteten Ebene wurden vollständig bestimmt	Ja (5 %)
	Größen und technisch-physikalische Schnittstellen von Elementen der betrachteten Ebene wurden vollständig identifiziert	Ja (5 %)
	Größen und technisch-physikalische Schnittstellen von Elementen der betrachteten Ebene wurden vollständig analysiert	Ja (5 %)
	Auftretende Störgrößen der betrachteten Ebene wurden hinsichtlich ihres Einflusses auf das System bewertet	n/a (5 %)
Systemmodellierung	Funktionen der betrachteten Ebene wurden vollständig modelliert	Ja (5 %)
	Komponenten der betrachteten Ebene wurden vollständig modelliert	Ja (5 %)
	Abgeleitete Anforderungen der betrachteten Ebene wurden vollständig modelliert	n/a (5 %)
	Strukturelle Relationen der betrachteten Ebene und hierarchische Relationen zur übergeordneten Ebene wurden vollständig modelliert	Ja (5 %)
	Größen und technisch-physikalische Schnittstellen der betrachteten Ebene wurden vollständig modelliert	Ja (5 %)
	Gesamtbewertung inkl. Vertrauensintervall	100 %, 4x n/a

Die Gesamtbewertung zum Meilenstein der ersten Lösungsebene zeigt, dass der Produktreifegrad mit 100 % bewertet wurde. Aufgrund von vier n/a-Bewertungen, welche sich durch nicht anwendbare Bewertungskriterien aufgrund der fehlender Störgrößen, abgeleiteten Anforderungen und Optimierungsmaßnahmen ergeben, wird jedoch nur ein gelb bewertetes Vertrauensintervall erreicht. Diese Bewertung des Vertrauensintervalls dient als Indikator für potentielle Risiken der Produktreifegradbewertung durch nicht berücksichtigte Störgrößen oder die fehlende Ableitung von Design Constraints bzw. die fehlende Implementierung von Optimierungsmaßnahmen. Generell bleibt jedoch ein Restrisiko bestehen, da auch bei identifizierten Störgrößen und implementierten Optimierungsmaßnahmen ggf. nicht alle Störgrößen identifiziert wurden. Die Angabe eines Vertrauensintervalls kann somit der Verifizierung der Bewertung dienen, jedoch nicht sämtliche Risiken einer unvollständigen oder falschen Anwendung der Schrittfolge absichern.

Eine kritische Prüfung der durchgeführten Schrittfolgen des Vorgehenskonzepts zeigt, dass alle Analysen vollständig durchgeführt wurden. Das Nichtvorhandensein von Störgrößen, und daraus resultierend dem Fehlen abgeleiteter Anforderungen und Optimierungsmaßnahmen, ist auf die geringe, aber erwartungsgemäße Konkretisierung der ersten Lösungsebene zurückzuführen. Somit kann das Quality Gate zum ersten Meilenstein passiert und die Systemkonkretisierung und -analyse auf der zweiten Lösungsebene fortgeführt werden.

5.4.3 Zweite Lösungsebene

Mit fortschreitender Systemkonkretisierung wächst für die zweite Lösungsebene auch die Anzahl der Systemmatrizen und der darin enthaltenen Elemente und Verknüpfungen. Für die einzelnen Schritte werden daher nur relevante Symbol-, Graphen- oder Matrizendarstellungen abgebildet, die der Validierung des Vorgehenskonzepts dienen. Unabhängig hiervon erfolgt die Modellierung vollständig in LOOMEO.

5.4.3.1 Funktion definieren

Um die bereits existierenden Funktionen erster Ebene zu konkretisieren, ist weiteres Fachwissen zur Funktionsweise von LASM erforderlich. Dieses Fachwissen wird bei der Anwendung des Ansatzes üblicherweise durch die an der Entwicklung beteiligten Domänen bereitgestellt. Bei der Validierung des Vorgehenskonzepts liegt der Fokus jedoch in der Bewertung der Anwendbarkeit der Schrittfolgen. Entsprechend werden für die Funktionsweise der LASM mit zunehmender Konkretisierung vereinfachende Annahmen getroffen.

Gemäß des Wirkprinzips einer LASM muss die elektrische Leistung P zunächst auf drei Phasen in die elektrischen Leistungen P_1, P_2 und P_3 aufgeteilt und in drei elektromagnetische Wechselfelder $\underline{B}_1$, $\underline{B}_2$ und $\underline{B}_3$ gewandelt werden, welche sich dann durch entsprechende Steuerung zu einem Wanderfeld $\underline{B}_{Wander}$ überlagern bzw. vereinigen. Dieses Wanderfeld wird über den Luftspalt induktiv auf den Sekundärteil übertragen. Im Sekundärteil induziert das Wanderfeld $\underline{B}_{Wander}$ eine Spannung, wodurch Wirbelströme fließen, die gemeinsam mit dem Wanderfeld schließlich in die resultierende translatorische Kraft F gewandelt werden, woraus die Bewegung des Sekundärteils resultiert, vgl. [Riekhof et al. 2013b]. Die bestehende Funktionsstruktur der ersten Ebene wird daher auf der zweiten Ebene durch folgende Funktionen konkretisiert:

- F_2.1: Aufteilen der elektrischen Leistung P in P_1, P_2, P_3
- F_2.2: Wandeln der elektrischen Leistung P_1 in Wechselfeld $\underline{B}_1$
- F_2.3: Wandeln der elektrischen Leistung P_2 in Wechselfeld $\underline{B}_2$
- F_2.4: Wandeln der elektrischen Leistung P_3 in Wechselfeld $\underline{B}_3$
- F_2.5: Vereinigen der Wechselfelder $\underline{B}_1$, $\underline{B}_2$, $\underline{B}_3$ zu einem Wanderfeld $\underline{B}_{Wander}$
- F_2.6: Leiten des Wanderfelds $\underline{B}_{Wander}$
- F_2.7: Wandeln des Wanderfelds $\underline{B}_{Wander}$ in translatorische Kraft F

5.4.3.1.1 Elemente, Größen und Beziehungen vererben

Aus der Funktionsstruktur der ersten Ebene müssen die Funktionskategorien, Größen und Beziehungen übernommen und im Systemmodell abgebildet werden. Eine direkte Übernahme von Elementen erfolgt für die Funktionen F_2.6 (Leiten des Wanderfelds $\underline{B}_{Wander}$) und F_2.7 (Wandeln des Wanderfelds $\underline{B}_{Wander}$ in translatorische Kraft F), welche auf der zweiten Ebene nicht

konkretisiert werden und somit inklusive ihrer Größen von den Funktionen erster Ebene (F_1.2 und F_1.3) vererbt werden.

5.4.3.1.2 Relationen identifizieren

Bei der Erfassung der hierarchischen Relationen zeigt sich, dass mehrere Funktionen zweiter Ebene jeweils eine Funktion erster Ebene konkretisieren, vgl. Abb. 75. Gleichzeitig ließen sich über die Matrizendarstellung etwaige Modellierungsfehler unmittelbar erkennen, da durch die gerichteten Relationen, welche gemäß des Ansatzes ausschließlich von Elementen untergeordneter Ebene auf Elemente übergeordneter Ebene bestehen dürfen, keine Relationen über Knoten in der Matrix oberhalb der Diagonalen gesetzt werden dürfen. Dieser Zusammenhang erschließt sich aus der Graphendarstellung hingegen nicht unmittelbar, zumal der Graph im Tool basierend auf den Knoten als Cluster gebildet wird und die dargestellte Struktur erst manuell generiert werden muss. Dennoch bietet die Graphendarstellung eine geeignete Kontrollmöglichkeit, ob die Modellierung in der Matrix S_{F_H} korrekt erfolgte.

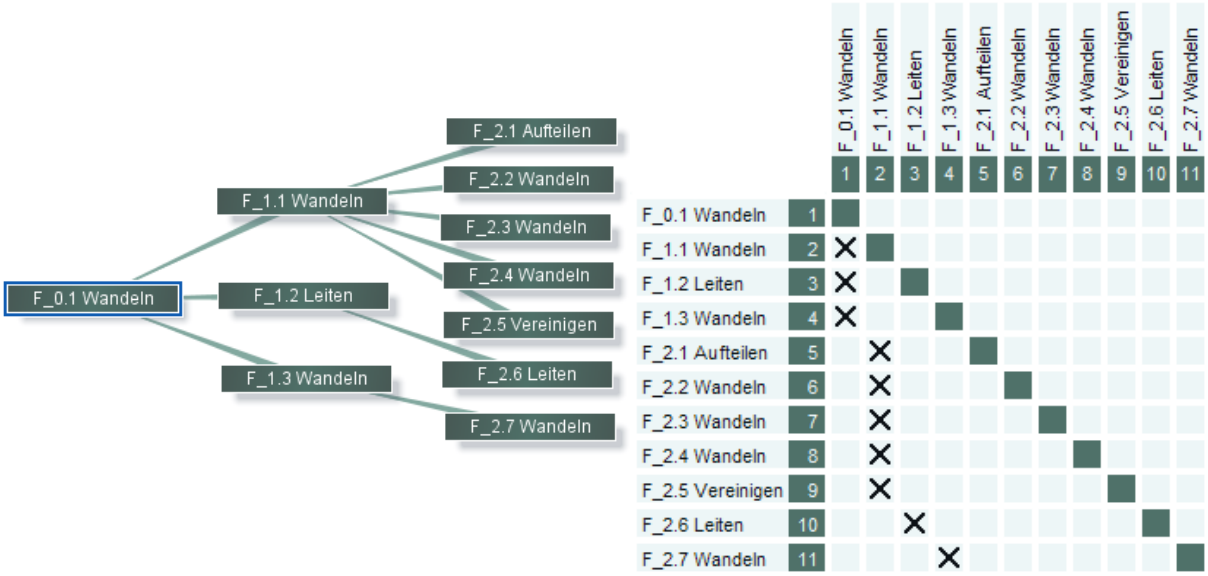


Abb. 75: Graphen- und Matrizendarstellung der Matrix S_{F_H} zweiter Ebene

Neben den hierarchischen Relationen existieren für die Funktionen der zweiten Ebene Abfolgerelationen, welche in der Systemmatrix S_{F_A} erfasst werden, vgl. Abb. 76. Die Abfolgerelation determiniert den logischen Zusammenhang der Funktionen. Ihre Modellierung mittels Graphen- und Matrizendarstellung zeigt die Möglichkeiten und Grenzen der Visualisierung. Während die Graphendarstellung einen verständlichen Eindruck der Funktionsstruktur vermittelt, erschließt sich aus der inhaltsgleichen Matrizendarstellung die Funktionsstruktur nicht unmittelbar. In der rechten Matrizendarstellung sind gegenüber der linken Graphendarstellung die Funktion F_0.1 nullter Ebene sowie die Funktionen F_1.1. F_1.2 und F_1.3 erster Ebene enthalten. Wie bereits erläutert (vgl. Abb. 73) stellt dies keinen Widerspruch zur Forderung dar, dass Abfolgerelationen nicht ebenenübergreifend modelliert werden dürfen. Entsprechend ist in der Matrizendarstellung der Matrix S_{F_A} keine Verknüpfung zwischen der Funktion nullter Ebene und Funktionen erster Ebene bzw. zwischen erster und zweiter Ebene gesetzt.

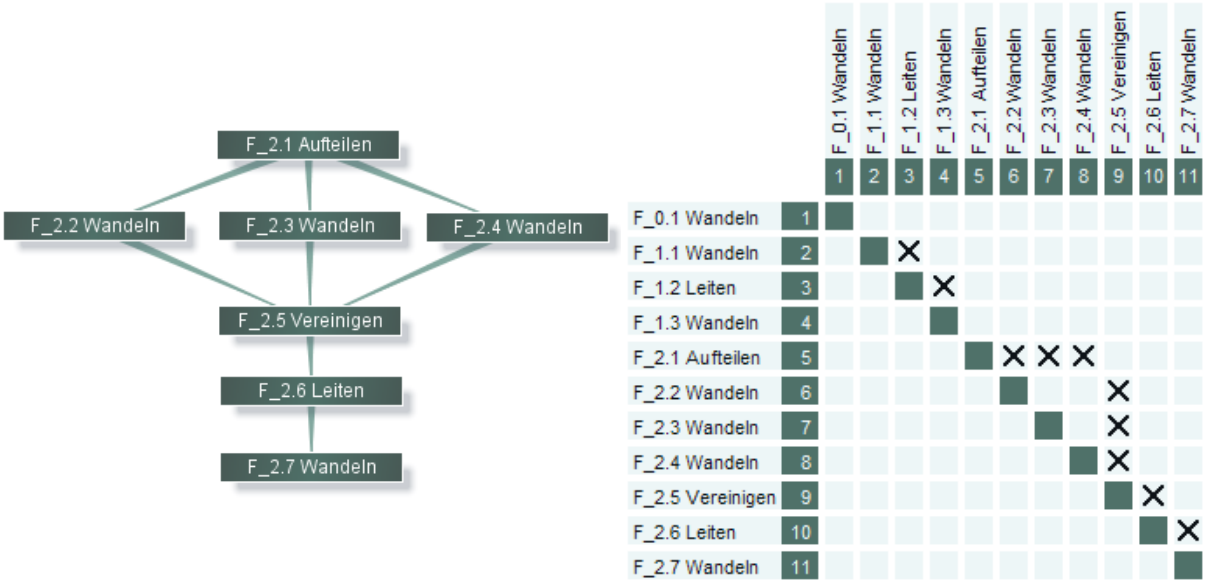


Abb. 76: Graphen- und Matrizendarstellung der Matrix S_{F_A} zweiter Ebene

Für die Funktionen zweiter Ebene existieren somit Hierarchie- und Abfolgerelationen, mangels Störgrößen und abgeleiteten Anforderungen jedoch keine Realisierungsrelation.

5.4.3.1.3 Größen und Schnittstellen identifizieren

Zusätzlich zu den bereits aus der ersten Lösungsebene vererbten Größen und Schnittstellen müssen in diesem Schritt alle neu definierten Größen und Schnittstellen identifiziert werden. Hierfür bietet sich die Symboldarstellung an. Wie bereits für die identifizierten Funktionen in Kap. 5.4.3.1 beschrieben, existieren neben den aus der ersten Ebene vererbten Größen und Schnittstellen (P, $\underline{B}_{Wander}$, F) weitere Größen und Schnittstellen, die entsprechend zu erfassen und modellieren sind. Hierfür bietet sich die Symboldarstellung an, da hierüber bereits die Größen und die Energieflüsse darstellbar sind, vgl. Abb. 77. Die Symboldarstellung stellt in diesem Fall sowohl die Informationen aus der Systemmatrix S_{F_A} zur Abfolgerelationen als auch die Information aus der Systemmatrix S_{F_EF} für die Energieflüsse dar, welche identische Verknüpfungen enthalten.

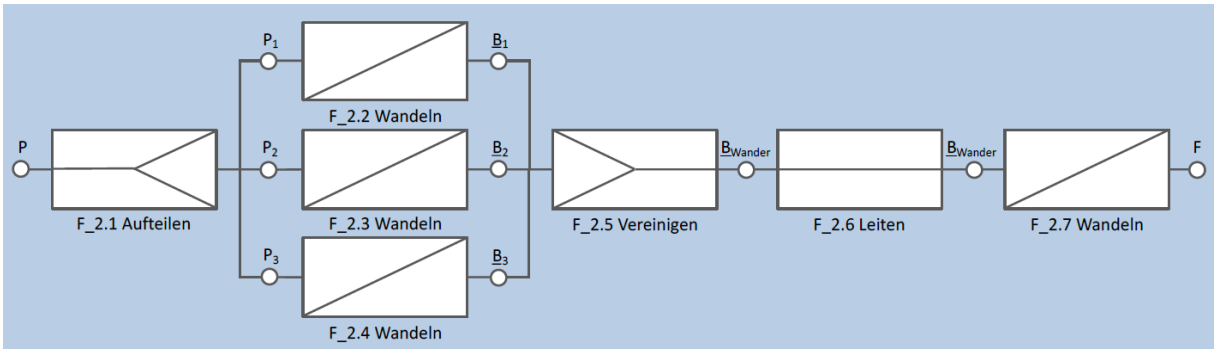


Abb. 77: Symboldarstellung der Größen und Schnittstellen der Funktionen zweiter Ebene

Die elektrische Leistung P wird aufgeteilt in drei elektrische Leistungen P_1, P_2 und P_3 der drei Phasen (Funktion F_2.1). Diese werden durch die Funktionen F_2.2, F_2.3 und F_2.4 in drei elektromagnetische Wechselfelder $\underline{B}_1$, $\underline{B}_2$ und $\underline{B}_3$ gewandelt, welche sich durch Überlagerung zu einem Wanderfeld $\underline{B}_{Wander}$ vereinigen (Funktion F_2.5). Durch die Funktion F_2.6 wird das

Wanderfeld geleitet und letztlich durch die Funktion F_2.7 in die translatorische Kraft *F* gewandelt.

5.4.3.2 Komponente festlegen

Für die Festlegung der Komponenten zweiter Ebene müssen die Komponenten erster Ebene so konkretisiert werden, dass sie mit einer sinnvollen Granularität die Funktionen der zweiten Ebene realisieren können.

Für das Aufteilen der elektrischen Leistung P in P_1, P_2 und P_3 ist eine Leistungselektronik erforderlich. Diese wird als K_2.1 bezeichnet und stellt eine neue Komponente gegenüber der ersten Ebene dar. Die Umwandlung der elektrischen Leistung P_1, P_2 und P_3 in die elektromagnetischen Wechselfelder $\underline{B}_1$, $\underline{B}_2$ und $\underline{B}_3$ erfolgt über die Drehstromwicklung des Primärteils, welche als Phase 1 (Komponente K_2.2), Phase 2 (Komponente K_2.3) und Phase 3 (Komponente K_2.4) bezeichnet werden. Diese Komponenten konkretisieren das Primärteil der ersten Ebene. Zur Vereinfachung wird an dieser Stelle nur ein Streckenabschnitt der intralogistischen Anlage betrachtet, der entsprechend der Länge der Förderstrecke zu duplizieren wäre. Zudem wird auf die Darstellung der mechanischen Grundstruktur verzichtet, da diese zur Erläuterung des Vorgehenskonzepts nicht erforderlich ist.

Die Überlagerung der elektromagnetischen Wechselfelder $\underline{B}_1$, $\underline{B}_2$ und $\underline{B}_3$ in ein Wanderfeld $\underline{B}_{Wander}$ wird durch eine elektrische Regelung bzw. Steuerung (Komponente K_2.5) realisiert. Diese Komponente ist ebenfalls nicht in der Komponentenstruktur der ersten Ebene enthalten, sondern ergibt sich aus der Realisierung der Funktion F_2.5 zweiter Ebene (Vereinigen der Wechselfelder $\underline{B}_1$, $\underline{B}_2$, $\underline{B}_3$ zu einem Wanderfeld $\underline{B}_{Wander}$).

Die induktive Energieübertragung erfolgt über den Luftspalt (Komponente K_2.6), welcher gegenüber der Komponente erster Ebene nicht weiter konkretisiert werden muss. Das Sekundärteil der ersten Ebene (Komponente K_1.3), welches das Wanderfeld $\underline{B}_{Wander}$ in die translatorische Kraft *F* wandelt, kann ebenfalls übernommen werden.

5.4.3.2.1 Elemente, Größen und Beziehungen vererben

Von der Komponentenstruktur erster Ebene werden der Luftspalt (K_1.2) sowie das Sekundärteil (K_1.3) inklusive ihrer Größen und Beziehungen direkt als Elemente vererbt, während das Primärteil (K_1.1) auf der zweiten Ebene über die Phasen (K_2.2, K_2.3, K_2.4) konkretisiert wird. Für sie werden Größen und Beziehungen von der Funktionsstrukur zweiter Ebene vererbt. Der Luftspalt ist als logischer Systembestandteil generell nicht granularer beschreibbar und wird auf alle untergeordneten Ebenen vererbt. Die Größen der hinzugekommenen Komponenten der zweiten Ebene, K_2.1 (Leistungselektronik) und K_2.5 (Regelung/Steuerung), werden ebenfalls von den zugrundeliegenden Funktionen vererbt.

Die Komponentenstruktur zweiter Ebene umfasst somit die Größen der elektrischen Leistung (P, P_1, P_2, P_3), die elektromagnetischen Wechselfelder ($\underline{B}_1$, $\underline{B}_2$, $\underline{B}_3$), das elektromagnetische Wanderfeld ($\underline{B}_{Wander}$) sowie die translatorische Kraft (F), dargestellt in Abb. 78.

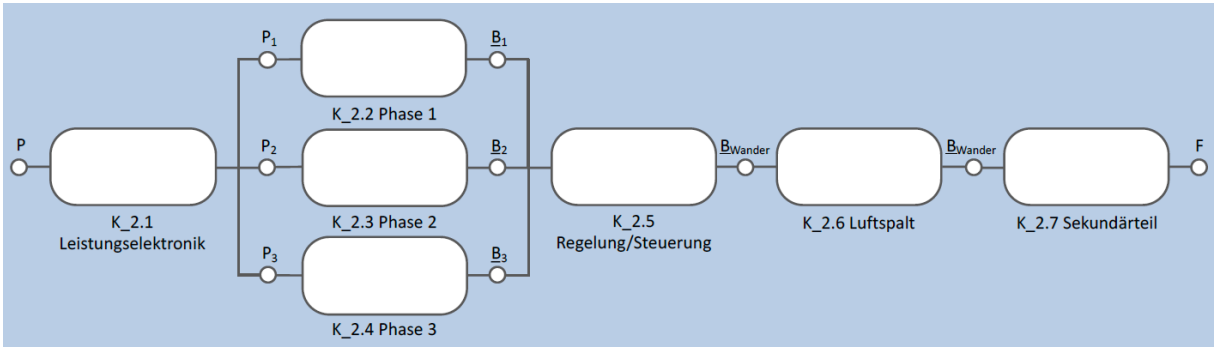


Abb. 78: Symboldarstellung der Größen und Schnittstellen der Komponenten zweiter Ebene

5.4.3.2.2 Relationen identifizieren

Die zunächst erfassten hierarchischen Relationen zeigen eine Besonderheit der Komponentenstruktur zweiter Ebene. Die Komponente K_1.1 (Primärteil) wird durch Konkretisierung granularer beschrieben und die Komponenten K_1.2 (Luftspalt) und K_1.3 (Sekundärteil) werden direkt vererbt. Entsprechend sind die konkretisierten bzw. vererbten Komponenten über eine „Bestandteil von“-Relation mit den Komponenten erster Ebene verknüpft. Bei der Festlegung der Komponenten zweiter Ebene wurden jedoch zwei neue Komponenten identifiziert, welche sich nicht aus einer Konkretisierung der Komponenten übergeordneter Ebene, sondern aus der Realisierung erforderlicher Funktionen ergeben, vgl. Abb. 79. Für diese Komponenten K_2.1 (Leistungselektronik) und K_2.5 (Regelung/Steuerung) besteht daher keine hierarchische Relation zu Komponenten erster Ebene, was in der Graphendarstellung durch die zwei unverknüpften Komponenten K_2.1 und K_2.5 und in der Matrizendarstellung durch fehlende Knoten in den entsprechenden Zeilen dargestellt ist. Außerdem ist zu erkennen, dass das Primärteil (K_1.1) durch die Komponenten K_2.2, K_2.3 und K_2.4 konkretisiert wird, während der Luftspalt (K_2.6) und das Sekundärteil (K_2.7) aus den identischen Komponenten (K_1.2 bzw. K_1.3) erster Ebene übernommen wurden.

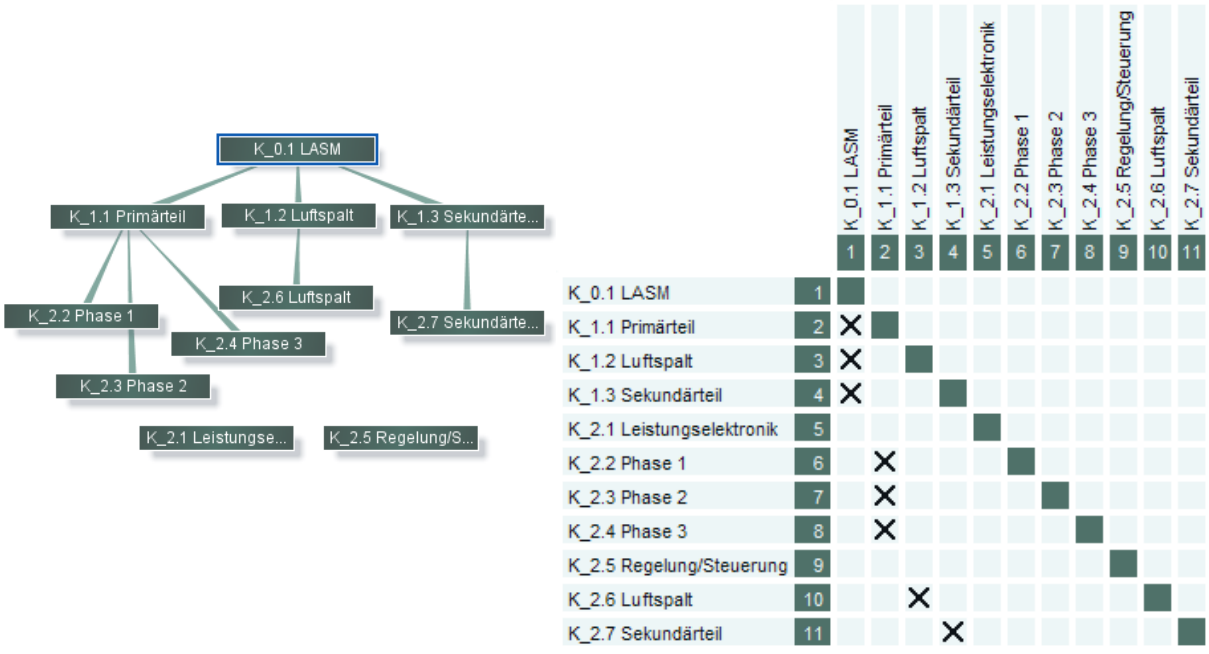


Abb. 79: Graphen- und Matrizendarstellung der Matrix SK_H zweiter Ebene

Zwischen den Komponenten und den Funktionen zweiter Ebene können weiterführend Realisierungsrelationen in der Systemmatrix S_{K,F_R} identifiziert werden, vgl. Abb. 80. Wie in anderen Matrizendarstellung sind auch hier die Komponenten und Funktionen aller Ebenen enthalten. Die Komponenten realisieren jeweils nur eine Funktion, wobei die Realisierungsrelation nur zwischen Elementen gleicher Lösungsebene existieren darf.

		F_0.1 Wandeln	F_1.1 Wandeln	F_1.2 Leiten	F_1.3 Wandeln	F_2.1 Aufteilen	F_2.2 Wandeln	F_2.3 Wandeln	F_2.4 Wandeln	F_2.5 Vereinigen	F_2.6 Leiten	F_2.7 Wandeln
		1	2	3	4	5	6	7	8	9	10	11
K_0.1 LASM	1	X										
K_1.1 Primärteil	2		X									
K_1.2 Luftspalt	3			X								
K_1.3 Sekundärteil	4				X							
K_2.1 Leistungselektronik	5					X						
K_2.2 Phase 1	6						X					
K_2.3 Phase 2	7							X				
K_2.4 Phase 3	8								X			
K_2.5 Regelung/Steuerung	9									X		
K_2.6 Luftspalt	10										X	
K_2.7 Sekundärteil	11											X

Abb. 80: Matrizendarstellung der Matrix S_{K,F_R} zweiter Ebene

Ein auffälliges Merkmal der Matrix S_{K,F_R} zweiter Ebene ist, dass die Realisierungsrelation über die Diagonale der Matrix beschrieben wird. Dies ist dem sukzessiven Vorgehen bei der Systemkonkretisierung geschuldet. So wurde für die Bestimmung der Komponenten zweiter Ebene die Funktionsstruktur herangezogen. Hierbei wurden schrittweise für alle Funktion funktionsrealisierende Komponenten identifiziert und modelliert, sodass die Knoten der Matrix in der Diagonalen erfasst sind.

Weitere mögliche Relationen, d.h. die Realisierung von Anforderungen (Matrix S_{K,A_R}) oder eine Erfordernisrelation (Matrix S_{K,A_E}), können für die Komponenten zweiter Ebene nicht identifiziert werden.

5.4.3.2.3 Größen und Schnittstellen identifizieren

Für die zielgerichtete Identifizierung von Größen und Schnittstellen bietet sich die Symboldarstellung an. Hierbei sind die Größen der Funktionsstruktur zu berücksichtigen, welche auf die Komponenten vererbt wurden, sowie neu identifizierte Größen und Schnittstellen.

Vereinfachend wird angenommen, dass bei der Überlagerung der Wechselfelder $\underline{B}_1$, $\underline{B}_2$ und $\underline{B}_3$ zum Wanderfeld $\underline{B}_{Wander}$ ein Energiefluss zwischen den beteiligten Komponenten, d.h. von K_2.2, K_2.3 und K_2.4 auf K_2.5 und hiervon auf K_2.6, genauso erfolgt, wie bei den Funktionen (F_2.2, F_2.3 und F_2.4 auf F_2.5, F_2.5 auf F_2.6), auch wenn dies für die Komponente K_2.5 (Regelung/Steuerung) de facto nicht zutrifft. Die Vereinfachung ermöglicht jedoch eine Darstellung analog zur Symboldarstellung der Funktionen zweiter Ebene (Abb. 77).

Bei der detaillierten Betrachtung der Wirkprinzipien und resultierenden Größen zeigt sich, dass durch die Spannungsinduktion des Wanderfelds $\underline{B}_{Wander}$ in das Sekundärteil Wirbelströme erzeugt werden, die mit dem magnetischen Feld des Wanderfelds eine Lorentzkraft erzeugen. Hierbei wird nicht nur die translatorische Kraft F erzeugt, sondern es resultieren aus dem Widerstand des Sekundärteils Stromwärmeverluste (Wicklungsverluste), wobei die Verlustenergie in Wärmeenergie umgesetzt wird [Riekhof et al. 2013b]. Der Wärmestrom $\dot{Q}$ stellt somit eine neue, gegenüber der zugrundeliegenden Funktion F_2.7 zusätzliche Ausgangsgröße der Komponente K_2.7 dar. Weitere zusätzliche Größen lassen sich nicht identifizieren.

Die Betrachtung der Größen zeigt, dass die Komponenten zweiter Ebene prinzipiell die gleichen Größen und Schnittstellen (Energiefluss) wie die Funktionen zweiter Ebene besitzen. Da bei der Funktionsrealisierung durch die Stromwärmeverluste jedoch eine zusätzliche Ausgangsgröße durch die Komponenten verursacht wird, unterscheidet sich die Komponentenstruktur in diesem Detail von der Funktionsstruktur. Die Symboldarstellung der Systemmatrix S_{K_EF} veranschaulicht diesen Zusammenhang, vgl. Abb. 81.

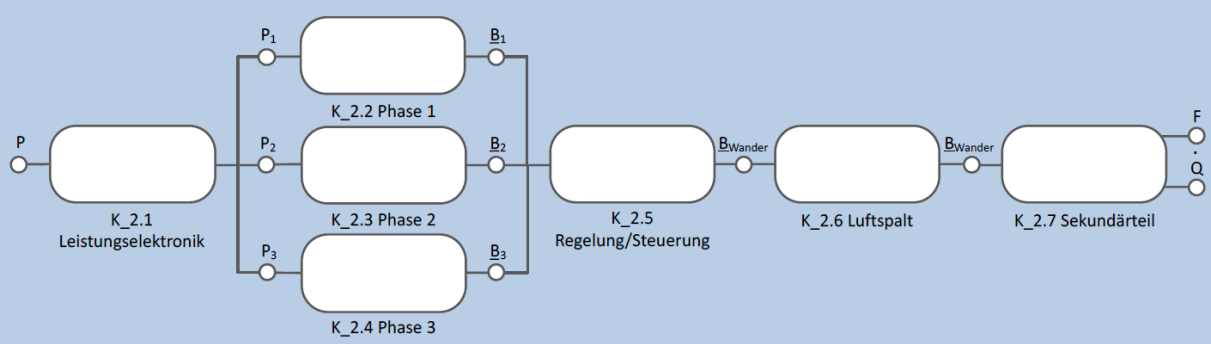


Abb. 81: Symboldarstellung der Größen und Schnittstellen der Komponenten zweiter Ebene (Ergänzung)

5.4.3.2.4 Größen und Schnittstellen analysieren

Erwartungsgemäß stellen die von den Funktionen übernommenen Größen Solleingangs- bzw. Sollausgangsgrößen der Komponenten dar, d.h. ihre Analyse begründete keinen Verdacht eines negativen Einflusses auf Systembestandteile. Eine Ausnahme bildet der resultierende Wärmestrom $\dot{Q}$. Unabhängig von der quantitativen Ausprägung, welche für den vorliegenden Grad der Systemkonkretisierung nicht determiniert ist, zeigt die Analyse der Ausgangsgröße der Komponente K_2.7, dass der Wärmestrom $\dot{Q}$ potentiell negative Auswirkungen besitzt. Einerseits bedeuten Stromwärmeverluste Leistungsverluste, da Energie nicht für den geplanten Zweck des Systems aufgewendet wird. Andererseits können Systembestandteile bzw. das Fördergut der intralogistischen Anlage durch Wärmeeinwirkung geschädigt werden. Der Wärmestrom stellt demzufolge eine Störausgangsgröße dar, auf welche entwicklungsseitig reagiert werden muss. Für die Quantifizierung des Einflusses der Störgröße müssen Berechnungen oder Simulationen zum Thermomanagement eingesetzt werden. Die Verknüpfung mit diesen Methoden ist expliziter Bestandteil des Vorgehenskonzepts.

Im Projekt Q-ELF wurde zur Quantifizierung der Störgröße die Erwärmung des Sekundärteils für den Nenn- und Dauerbetrieb auf Basis der vorgegebenen Betriebsparameter simuliert. Während im Nennbetrieb lokale Höchstwerte von 112,89 °C resultieren, können im Worst-case-Szenario, d.h. dem Betrieb bei mechanischer Blockierung der Förderstrecke, Temperaturen auf dem Sekundärteil von bis zu 335,74 °C entstehen [Willing et al. 2014, S. 29ff]. Da die

simulierte Erwärmung des Sekundärteils Systembestandteile oder das Fördergut selbst schädigen kann, ist der Wärmestrom $\dot{Q}$ als Störausgangsgröße bestätigt. Entsprechend muss gemäß der Schrittfolge auf die Störgröße reagiert werden, indem eine abgeleitete Anforderung definiert wird.

5.4.3.3 Abgeleitete Anforderung definieren

Als Reaktion auf die Störgröße $\dot{Q}$ muss eine Anforderung als sog. Design Constraint abgeleitet werden. Generell besteht beim Auftreten von Störgrößen das primäre Ziel in der Eliminierung der Störgröße durch geeignete Maßnahmen. Da physikalisch bedingte Stromwärmeverluste nicht eliminierbar sind, muss stattdessen der Einfluss der Störgröße bestmöglich reduziert werden. Entsprechend muss in diesem Schritt ein Design Constraint definiert werden, dessen Notwendigkeit sich ausschließlich über die Existenz der Störgröße ergibt. Diese abgeleitete Anforderung beschreibt das Ziel im Umgang mit der Störgröße.

Für die Stromwärmeverluste bzw. den Wärmestrom besteht das Ziel einerseits in der Erhöhung des Wirkungsgrads des Systems, andererseits in der Abführung der Wärme, da die Störgröße nicht eliminiert, sondern nur reduziert werden kann und entsprechend auch durch einen erhöhten Wirkungsgrad der Einfluss der Störgröße fortbesteht. Demzufolge lassen sich die zwei Anforderungen A_2.1 (Wirkungsgrad erhöhen) und A_2.2 (Wärme abführen) ableiten.

5.4.3.3.1 Relationen identifizieren

Die abgeleiteten Anforderungen A_2.1 und A_2.2 werden zunächst standardgemäß in der Systemmatrix S_{A_H} erfasst. Da die Anforderungen als Design Constraints kein Bestandteil der Anforderung nullter Ebene (A_0.1 Stückgut befördern) sind, wird keine hierarchische Relation modelliert, vgl. Abb. 82. Dementsprechend enthält die Matrix keine Knoten.

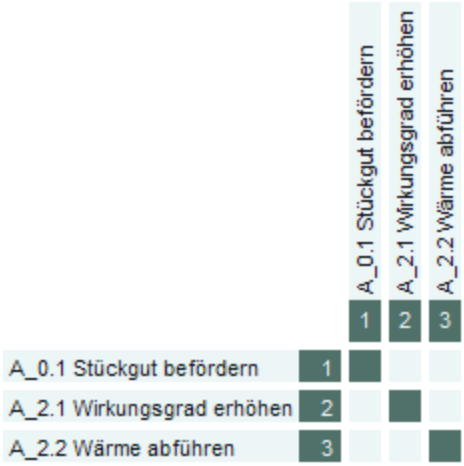


Abb. 82: Matrizendarstellung der Matrix S_{A_H} zweiter Ebene

Das Vorhandensein der abgeleiteten Anforderungen ist stattdessen ausschließlich auf das Vorhandensein der Störgröße zurückzuführen. Demzufolge besteht zwischen der Komponente K_2.7, welche die Störfunktion verursacht, und den abgeleiteten Anforderungen eine Erfordernisrelation, welche in der Systemmatrix S_{K,A_E} dargestellt ist, vgl. Abb. 83.

		A_0.1 Stückgut befördern	A_2.1 Wirkungsgrad erhöhen	A_2.2 Wärme abführen
		1	2	3
K_0.1 LASM	1			
K_1.1 Primärteil	2			
K_1.2 Luftspalt	3			
K_1.3 Sekundärteil	4			
K_2.1 Leistungselektronik	5			
K_2.2 Phase 1	6			
K_2.3 Phase 2	7			
K_2.4 Phase 3	8			
K_2.5 Regelung/Steuerung	9			
K_2.6 Luftspalt	10			
K_2.7 Sekundärteil	11		X	X

Abb. 83: Matrizendarstellung der Matrix S_{K,A_E} zweiter Ebene

5.4.3.4 Optimierungsmaßnahmen implementieren

Um die abgeleiteten Anforderungen realisieren zu können, müssen zunächst Funktionen definiert werden, deren Zweck sich auf den Umgang mit der Störgröße bezieht. Hierfür sind bei Bedarf Größen und Beziehungen zu vererben und alle Relationen sowie Größen und Schnittstellen zu identifizieren. Anschließend müssen geeignete Komponenten zur Funktionsrealisierung festgelegt werden und auch für diese die Schritte zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse durchgeführt werden. Für Komponenten sind zusätzlich Schnittstellen zu analysieren, um auszuschließen, dass durch die Implementierung der Optimierungsmaßnahme neue Störgrößen resultieren. Der Schritt zur Implementierung von Optimierungsmaßnahmen umfasst somit prinzipiell die Schritte *Funktion definieren* und *Komponente definieren* sowie die zugehörigen Schritte der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse.

Für die identifizierte Störgröße bezieht sich die Anforderung A_2.1 auf die Erhöhung des Wirkungsgrads. Hierfür muss die Effizienz bei der Wandlung des Wanderfelds $\underline{B}_{Wander}$ in die translatorische Kraft F gesteigert und somit die Störgröße durch die Reduzierung der Stromwärmeverluste bestmöglich reduziert werden. Dieser Zweck wird mit der Funktion F_2.8 „Verkleinern der Stromwärmeverluste“ beschrieben, welche die bestehenden Funktionen der zweiten Lösungsebene ergänzt. Für die Anforderung A_2.2 (Wärme abführen) wird die Funktion F_2.9 „Leiten von Wärme“ definiert. Diese sieht das Ableiten der entstehenden Wärme in die Systemumwelt vor, um die negative Auswirkung der Störgröße zu reduzieren.

Zur Realisierung der Funktionen werden zwei Komponenten benötigt, welche das Sekundärteil konkretisieren und für die zweite Lösungsebene ersetzen. Entsprechend kann das Sekundärteil in eine Kupferplatte (K_2.7) und einen sog. eisernen Rückschluss mit Kühlrippen zerlegt

werden. Die Kupferplatte (K_2.7) übernimmt anstelle des Sekundärteils die Realisierung der Funktion F_2.7 (Wandeln des Wanderfelds $\underline{B}_{Wander}$ in translatorische Kraft F). Der eiserne Rückschluss mit Kühlrippen (K_2.8) erhöht den Wirkungsgrad und realisiert damit die Funktion F_2.8 („Verkleinern der Stromwärmeverluste“). Zudem wird durch die Kühlrippen die verbleibende Wärme über die vergrößerte Oberfläche in die Systemumwelt ableitet und somit die Funktion F_2.9 („Leiten von Wärme“) realisiert.

5.4.3.4.1 Elemente, Größen und Beziehungen vererben

Für die Funktionen müssen keine Elemente, Größen oder Beziehungen vererbt werden, da die Funktionen F_2.8 und F_2.9 die bestehende Funktionsstruktur zweiter Ebene ergänzen.

In der Komponentenstruktur ersetzt die Kupferplatte (K_2.7) den Platz des bisher auf der zweiten Lösungsebene genutzten Sekundärteils. Entsprechend werden die Größen des Sekundärteils, die Eingangsgröße $\underline{B}_{Wander}$ und die Ausgangsgröße F, auf die Kupferplatte vererbt. Auch die Störgröße $\dot{Q}$ wird als Ausgangsgröße auf die Kupferplatte vererbt. Der eiserne Rückschluss mit Kühlrippen (K_2.8) ergänzt die Komponentenstruktur, weshalb für diese Komponente keine Vererbung stattfindet.

5.4.3.4.2 Relationen identifizieren

Für die abgeleiteten Anforderungen, die hieraus resultierenden Funktionen und die funktionsrealisierenden Komponenten sind nun die Relationen zu erfassen.

Da die Notwendigkeit der abgeleiteten Anforderungen ausschließlich auf das Vorhandensein der Störgröße zurückzuführen ist, welche aus der Komponente K_2.7 resultiert, resultiert eine Erfordernisrelation zwischen der Komponente und den Anforderungen A_2.1 und A_2.2. Diese wird in der Systemmatrix S_{K,A_E} dargestellt. Im Unterschied zur bisherigen Darstellung (vgl. Abb. 83) wird die Bezeichnung der Komponente K_2.7 lediglich ausgetauscht, d.h. die Kupferplatte wird anstelle des Sekundärteils genannt.

Bei den Funktionen wird die Hierarchiedarstellung um die Funktionen F_2.8 und F_2.9 ergänzt (Matrix S_{F_H}). Eine Besonderheit besteht darin, dass die Funktionen F_2.8 und F_2.9, welche ausschließlich der Realisierung der Design Constraints dienen, nicht hierarchisch mit Funktionen der ersten Lösungsebene verbunden sind, da sie keine Konkretisierung von Funktionen übergeordneter Ebene leisten, vgl. Abb. 84. Während alle Funktionen der ersten Ebene mit der nullten Ebene verknüpft sind, liegt eine hierarchische Relation der Funktionen zweiter Ebene zu denen der ersten Ebene somit nur für die Funktionen vor, welche sich durch Konkretisierung oder Vererbung durch die übergeordnete Ebene ergeben, nicht jedoch für Funktionen, deren Notwendigkeit sich aus einem Design Constraint ergibt. Auf einer untergeordneten Ebene hingegen müssten F_2.8 und F_2.9 – wie alle Funktion der übergeordneten Ebene – hierarchisch mit den Elementen der dritten Lösungsebene verknüpft werden, da sie entweder auf die untergeordnete Ebene vererbt oder dort konkretisiert werden müssen.

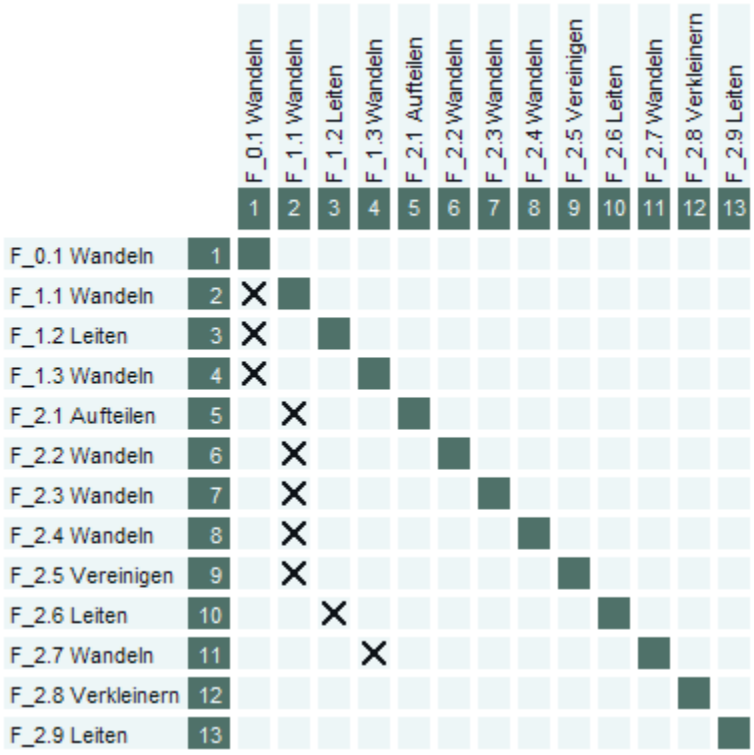


Abb. 84: Matrizendarstellung der Matrix S_{F_H} zweiter Ebene nach Optimierung

Auch für die Abfolgerelation innerhalb der Funktionssicht (Matrix S_{F_A}) gilt, dass die Funktionen F_2.8 und F_2.9 nicht mit den anderen Funktionen der zweiten Lösungsebene verknüpft werden, da innerhalb der Funktionssicht die hierfür erforderliche Störgröße $\dot{Q}$ nicht existiert. Untereinander sind sie verknüpft, da die Ausgangsgröße der Funktion F_2.8 die Eingangsgröße der Funktion F_2.9 ist und diese auf F_2.8 folgt. In der Matrizendarstellung (Abb. 85) wird dies durch den fehlenden Knoten zwischen der Funktion F_2.7 und der Funktion F_2.9 beschrieben.

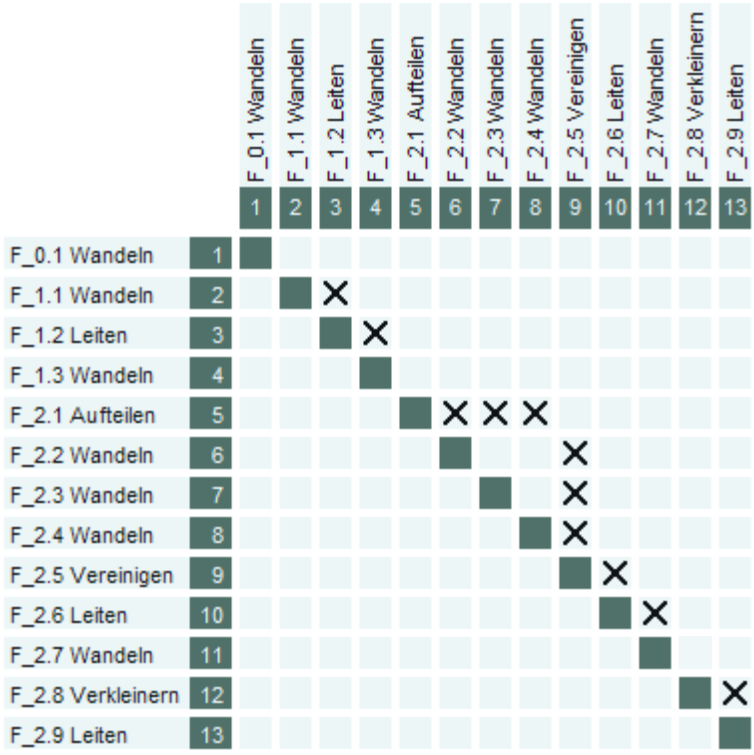


Abb. 85: Matrizendarstellung der Matrix S_{F_A} zweiter Ebene nach Optimierung

Für die Funktionen F_2.8 und F_2.9 kann außerdem eine Realisierungsrelation identifiziert werden, welche zwischen ihnen und den zugrundenliegenden Anforderungen A_2.1 und A_2.2 besteht und in der Systemmatrix S_{F,A_R} beschrieben wird. Eine Erfordernisrelation zwischen Anforderungen und Funktionen, welche prinzipiell zulässig ist, liegt hingegen nicht vor, da diese nur gegeben wäre, wenn Anforderungen eine konkrete Funktion direkt fordern. Dies ist hier nicht der Fall, da die Funktionen reguläre anforderungsrealisierende Elemente sind.

Für Komponenten können zunächst hierarchische Relationen identifiziert werden. Anders als bei den Funktionen sind die Komponenten K_2.7 und K_2.8 mit denen der übergeordneten Ebene hierarchisch verknüpft, da sie eine Konkretisierung des Systems erster Lösungsebene darstellen. Wie Abb. 87 in der Graphen- und Matrizendarstellung zeigt, ersetzt die Kupferplatte das Sekundärteil als K_2.7. Zusätzlich wird der eiserne Rückschluss als K_2.8 gebildet. Beide Komponenten sind Bestandteil des Sekundärteils der ersten Ebene, d.h. der Komponente K_1.3.

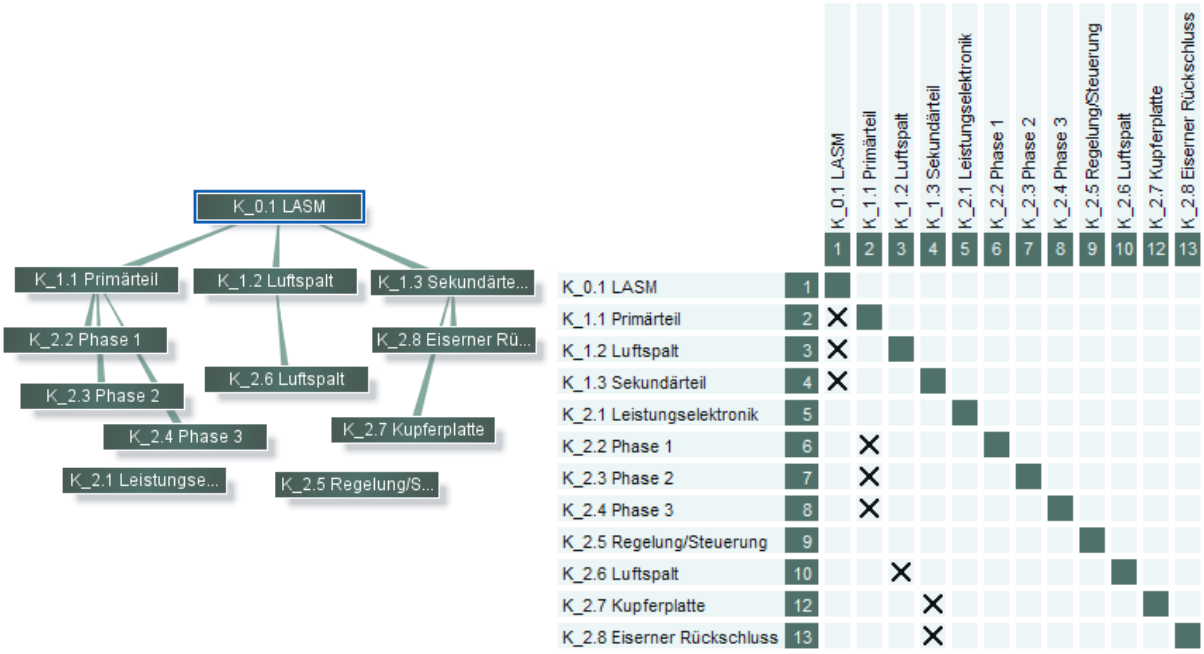


Abb. 86: Graphen- und Matrizendarstellung der Matrix S_{K_H} zweiter Ebene nach Optimierung

Auch bei der Abfolgerelation können K_2.7 und K_2.8 in die bestehende Struktur eingebunden werden. Da auf die Kupferplatte K_2.7 die Größen des zuvor beschriebenen Sekundärteils vererbt wurden, folgt sie auf K_2.6. Der eiserne Rückschluss K_2.8 nimmt wiederrum die Ausgangsgrößen von K_2.7 als Eingangsgröße auf und folgt damit dieser Komponente.

Hinsichtlich der Realisierung bestehen neue Relationen zwischen K_2.7 und K_2.8 und den Funktionen, die sie realisieren, d.h. F_2.7, F_2.8 und F_2.9, dargestellt in Abb. 87.

		F_0.1 Wandeln	F_1.1 Wandeln	F_1.2 Leiten	F_1.3 Wandeln	F_2.1 Aufteilen	F_2.2 Wandeln	F_2.3 Wandeln	F_2.4 Wandeln	F_2.5 Vereinigen	F_2.6 Leiten	F_2.7 Wandeln	F_2.8 Verkleinern	F_2.9 Leiten
		1	2	3	4	5	6	7	8	9	10	11	12	13
K_0.1 LASM	1	X												
K_1.1 Primärteil	2		X											
K_1.2 Luftspalt	3			X										
K_1.3 Sekundärteil	4				X									
K_2.1 Leistungselektronik	5					X								
K_2.2 Phase 1	6						X							
K_2.3 Phase 2	7							X						
K_2.4 Phase 3	8								X					
K_2.5 Regelung/Steuerung	9									X				
K_2.6 Luftspalt	10										X			
K_2.7 Kupferplatte	12											X		
K_2.8 Eiserner Rückschluss	13												X	X

Abb. 87: Matrizendarstellung der Matrix S_{K,F_R} zweiter Ebene nach Optimierung

Bei der Erfordernisrelation zwischen Komponenten und Anforderungen (Matrix S_{K,A_E}) wurde bereits die hinterlegte Verknüpfung für die Komponente K_2.7 vom Sekundärteil auf die Kupferplatte vererbt, sodass diese durch die auftretende Störgröße die abgeleiteten Anforderungen erforderlich macht. Für die Komponenten K_2.8 lässt sich keine Erfordernisrelation identifizieren, da sie keine Störgröße verursacht.

5.4.3.4.3 Größen und Schnittstellen identifizieren

Für die zielgerichtete Identifizierung und Modellierung der Größen und Schnittstellen hat sich die Symboldarstellung bewährt. Die Funktionsstruktur nach Optimierung zeigt, dass die Funktionen F_2.8 und F_2.9 ergänzt wurden und jeweils die Eingangs- und Ausgangsgröße $\dot{Q}$ besitzen, siehe Abb. 88. Sie sind jedoch nicht mit den restlichen Funktionen der zweiten Ebene verbunden, da die Störgröße $\dot{Q}$ keine Ausgangsgröße der Wandlungsfunktion F_2.7 darstellt, sondern nur der sie realisierenden Komponente. Bezogen auf die Funktionen F_2.8 und F_2.9 stellt der Wärmestrom $\dot{Q}$ eine Sollgröße dar, da die Funktionen den Umgang mit ihr beschreiben. Weitere Größen für die Funktionssicht konnten nicht identifiziert werden.

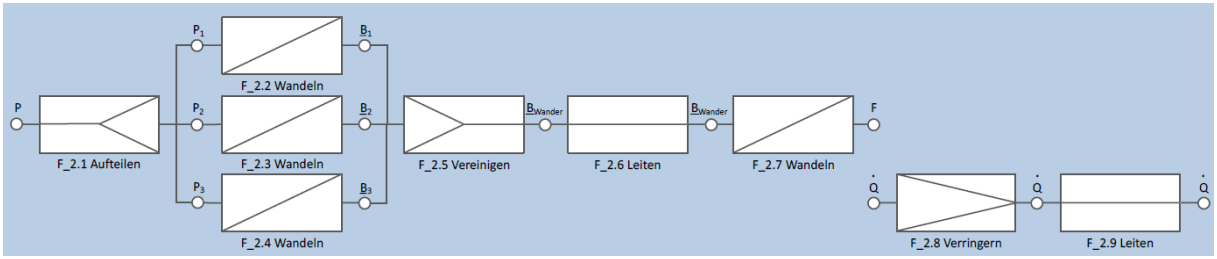


Abb. 88: Symboldarstellung der Größen und Schnittstellen der Funktionen zweiter Ebene nach Optimierung

Für die Komponentensicht stellt $\dot{Q}$ eine Ausgangsgröße der Komponente K_2.7 dar und ist entsprechend mit den Komponenten über einen Energiefluss verknüpft, welcher sich in der

Systemmatrix S_{K_EF} beschreiben lässt. Als transparente Modellierungsform bietet sich, wie bei der Funktionsstruktur, die Symboldarstellung an, vgl. Abb. 89. Gegenüber der Komponentenstruktur vor Identifizierung der Störgröße bzw. nach der Ergänzung um diese sind hier die geänderte Komponente K_2.7 und die zusätzliche Komponente K_2.8 dargestellt. K_2.7 zeigt den Wärmestrom $\dot{Q}$ als zusätzliche Ausgangsgröße, welche K_2.9 als Eingangsgröße dient. Da sie durch die Implementierung eines eisernen Rückschlusses mit Kühlrippen zwar reduziert, aber nicht eliminiert wird, stellt $\dot{Q}$ auch die Ausgangsgröße von K_2.9 dar.

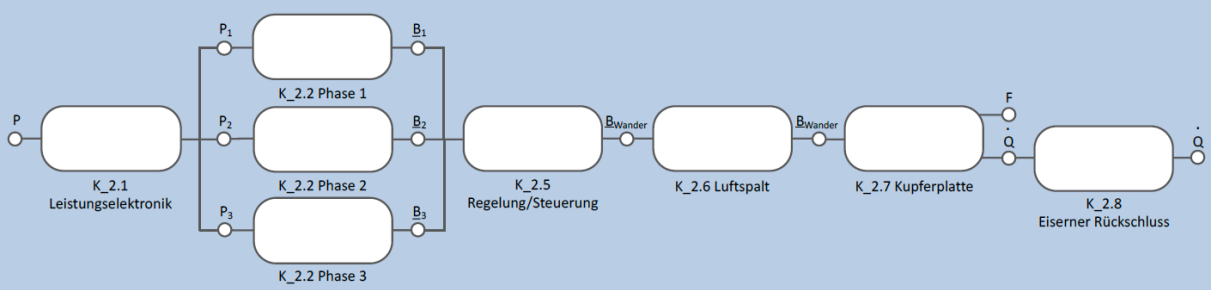


Abb. 89: Symboldarstellung der Größen und Schnittstellen der Komponenten zweiter Ebene nach Optimierung

Auch für die Komponentensicht konnten keine neuen Größen, welche sich aus der Optimierungsmaßnahme selbst ergeben, identifiziert werden.

5.4.3.5 Größen und Schnittstellen analysieren

Die Analyse der Größen und Schnittstellen für die implementierten Optimierungsmaßnahmen dient der Sicherstellung, dass durch die Optimierung keine neuen Störgrößen resultieren bzw. im Falle ihres Auftretens die Schrittfolge erneut durchlaufen wird, beginnend mit der Ableitung von Anforderungen. Für die dargestellte Komponentenstruktur zweiter Ebene konnten keine zusätzlichen Größen, ergo auch keine neuen Störgrößen, identifiziert werden.

Durch die Einbindung von Simulationen konnte gezeigt werden, dass für die beschriebenen Anwendungsfälle durch die Ausführung des eisernen Rückschlusses mit 5 mm breiten Kühlrippen die maximale Oberflächentemperatur im Nennbetrieb von 112,89 °C auf 52,14 °C und bei mechanischer Blockierung von 335,74 °C auf 220,88 °C gesenkt werden konnte [Willing et al. 2014, S. 29ff]. Sollten diese Werte weiterhin einen störenden Einfluss besitzen, ließe sich das Entwärmverhalten durch eine geringere Zahnbreite und somit eine vergrößerte Oberfläche der Kühlrippen optimieren, wobei hierbei die mechanische Stabilität im Sinne der Tragfähigkeit des Förderguts den limitierenden Faktor darstellt. Für die Veranschaulichung des Vorgehenskonzepts wird diese Reduzierung der Störgröße als ausreichend angesehen, sodass der thematische Block zum Umgang mit der Störgröße abgeschlossen ist und zurück in die Schrittfolge zur Konkretisierung der Lösungsebene gesprungen werden kann, in diesem Fall der Abschluss der Lösungsebene mit der Ermittlung des Produktreifegrads.

5.4.3.6 Ermittlung des Produktreifegrads für die zweite Ebene

Während die Durchführung der Schrittfolgen zur Konkretisierung des Lösungsraums bzw. zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse für die erste Ebene nur wenige Elemente, Relationen und Schnittstellen enthielt, welche zu identifizieren, analysieren und zu modellieren waren, wächst mit steigender Produktreife auch der Umfang durchzu-

führender Tätigkeiten. Um die Bewertungskriterien zu erfüllen, müssen entsprechend deutlich umfangreichere Schritte durchgeführt werden. Dies stellt die Skalierbarkeit bei der Bewertung des Produktreifegrads sicher.

Für die zweite Lösungsebene wurden die beiden Schrittfolgen für die Konkretisierung des Lösungsraums und die Relationsbestimmung und Schnittstellenidentifizierung und -analyse erstmals vollständig durchlaufen. Hinsichtlich der Systemkonkretisierung wurden neue Funktionen und Komponenten definiert. Da Störgrößen identifiziert werden konnten, wurden Anforderungen abgeleitet und Optimierungsmaßnahmen implementiert. Ebenso wurden für alle Systemelemente der Lösungsebene Relationen und Schnittstellen identifiziert und analysiert, sodass auch hier alle erwarteten Ergebnisse vorliegen. Diese Ergebnisse wurden in der Matrizen-, Graphen- und/oder der Symboldarstellung modelliert und auszugsweise dargestellt. In Summe wurden damit alle erwarteten Ergebnisse erreicht, was zu einer Gesamtbewertung von 100 % führt, vgl. Tab. 20. Da kein Kriterium ausgelassen und als nicht anwendbar (n/a) gekennzeichnet wurde, resultiert ein grünes Vertrauensintervall. Das Quality Gate kann somit passiert und die Systemkonkretisierung und -analyse könnte für die dritte Lösungsebene fortgesetzt werden.

Tab. 20: Bewertung des Produktreifegrads zweiter Ebene

Schrittfolge	Erwartete Ergebnisse	Bewertung
Konkretisierung des Lösungsraums	Auf der betrachteten Ebene wurde mindestens eine zusätzliche Funktion im Sinne der Systemkonkretisierung definiert	Ja (10 %)
	Für jede Funktion der betrachteten Ebene wurde mindestens eine funktionsrealisierende Komponente festgelegt	Ja (10 %)
	Beim Auftreten von Störgrößen auf der betrachteten Ebene wurden Anforderungen zum Umgang mit diesen abgeleitet	Ja (10 %)
	Beim Vorliegen abgeleiteter Anforderungen wurden Optimierungsmaßnahmen auf der betrachteten Ebene implementiert	Ja (10 %)
Relationsbestimmung und Schnittstellenidentifizierung und -analyse	Funktionen und Komponenten wurden von der übergeordneten Ebene auf die betrachtete Ebene vererbt oder auf der betrachteten Ebene konkretisiert	Ja (10 %)
	Eingangs- und Ausgangsgrößen und Schnittstellen der übergeordneten Ebene wurden auf Elemente der betrachteten Ebene vererbt	Ja (5 %)
	Hierarchische Relationen von Elementen der betrachteten Ebene zu Elementen der übergeordneten Ebene wurden vollständig bestimmt	Ja (5 %)
	Strukturelle Relationen von Elementen der betrachteten Ebene wurden vollständig bestimmt	Ja (5 %)
	Größen und technisch-physikalische Schnittstellen von Elementen der betrachteten Ebene wurden vollständig identifiziert	Ja (5 %)
	Größen und technisch-physikalische Schnittstellen von Elementen der betrachteten Ebene wurden vollständig analysiert	Ja (5 %)
	Auftretende Störgrößen der betrachteten Ebene wurden hinsichtlich ihres Einflusses auf das System bewertet	Ja (5 %)
Systemmodellierung	Funktionen der betrachteten Ebene wurden vollständig modelliert	Ja (5 %)
	Komponenten der betrachteten Ebene wurden vollständig modelliert	Ja (5 %)
	Abgeleitete Anforderungen der betrachteten Ebene wurden vollständig modelliert	Ja (5 %)
	Strukturelle Relationen der betrachteten Ebene und hierarchische Relationen zur übergeordneten Ebene wurden vollständig modelliert	Ja (5 %)
	Größen und technisch-physikalische Schnittstellen der betrachteten Ebene wurden vollständig modelliert	Ja (5 %)
	Gesamtbewertung inkl. Vertrauensintervall	100 %, 0x n/a

Wie zu Beginn der Validierung erläutert, wird der Durchlauf der Schrittfolge auf die nullte, erste und zweite Lösungsebene beschränkt, da diese das Vorgehenskonzept bereits vollumfänglich veranschaulichen können. Es wäre zu vermuten, dass die Systemkonkretisierung nach einer oder zwei weiteren Lösungsebenen in ihrer Granularität soweit fortgeschritten ist, dass eine Übergabe des Systementwurfs an die Phase des domänenspezifischen Entwurfs erfolgen könnte und die frühen Phasen der Produktentwicklung damit abgeschlossen wären.

5.5 Ermittlung des Prozessreifegrads

Die Grundlage bei Prozessreifegradermittlung stellen die in Kap. 3.1.1.2.2 vorgestellten abgeleiteten Anforderungen des CMMI-DEV dar. Die Bewertung erfolgt gemäß den abgeleiteten Anforderungen, vgl. Tab. 8. Die Anforderungen werden nachfolgend besprochen und ihre Erfüllung bewertet, was die Grundlage für die Ermittlung des Prozessreifegrads bildet.

Die Bewertungssystematik folgt prinzipiell dem Aufbau der Bewertungsstufen der spezifischen Ziele und Praktiken, siehe Tab. 6. Dementsprechend wird der Prozessreifegrad über vier Stufen bewertet, wobei die Stufen anstelle der Relevanz den Erfüllungsgrad der abgeleiteten Anforderungen bewerten. Zur Ermittlung des resultierenden Prozessreifegrads ist den Bewertungsstufen ein prozentualer Erfüllungsgrad zugeordnet, wobei dieser semi-quantitativ ist. Die Bewertungsstufen der abgeleiteten Anforderungen sind „keine Anforderungserfüllung“ (0 %), „geringe Anforderungserfüllung“ (33 %), „hohe Anforderungserfüllung“ (67 %) und „vollständige Anforderungserfüllung“ (100 %).

Die nachfolgend vorgestellte Bewertung der Anforderungserfüllung wird zur besseren Übersichtlichkeit unterteilt in die Bewertungskriterien „Frühe Entwicklungsphasen“, „Schnittstellenidentifizierung und -analyse“, „Komplexität“, „Zuverlässigkeit“, „Reifegradabsicherung“ sowie „nicht kategorisierte Anforderungen“.

5.5.1 Bewertung der Erfüllung abgeleiteter Anforderungen

5.5.1.1 Frühe Entwicklungsphasen

Gemäß den abgeleiteten Anforderungen muss in den frühen Entwicklungsphasen ein Schnittstellenmanagement implementiert werden. Diese Anforderung ist dem Prozessgebiet Produktintegration (PI), der Praktik SP 2.2 und den Bewertungskriterien „Frühe Entwicklungsphasen“ sowie „Schnittstellenidentifizierung und -analyse“ zugeordnet. Hinsichtlich der Anforderungserfüllung lässt sich feststellen, dass das vorgestellte Vorgehenskonzept auf die Phase des Systementwurfs gemäß des V-Modells fokussiert und damit explizit frühe Entwicklungsphasen. Für diese Phasen sind in der Schrittfolge die Schnittstellenidentifizierung und die Schnittstellenanalyse über die Schritte „Größen und Schnittstellen identifizieren“ bzw. „Größen und Schnittstellen analysieren“ prozesstechnisch verankert. Hierbei werden Schnittstellen gezielt identifiziert und bezüglich möglicher Störgrößen sowie deren Einfluss auf das System bewertet. Ergänzend erfolgt in der Schrittfolge für die Systemmodellierung die modelltechnische Abbildung identifizierter Schnittstellen. Ein auf diese Weise implementiertes Schnittstellenmanagement bewertet vollumfänglich technisch-physikalische Schnittstellen und stellt ihren Einfluss auf das System dar. Neben Schnittstellen, welche über das Vorgehenskonzept umfangreich abgedeckt sind, ist auch die Berücksichtigung von Benutzerschnittstellen gefordert, vgl. [SEI 2011, S. 267]. Diese Schnittstellen werden durch den Ansatz nicht behandelt, sodass keine vollständige, sondern eine hohe Anforderungserfüllung (67 %) resultiert.

Ein Vorgehenskonzept muss zudem eine frühzeitige Validierung von Anforderungen sicherstellen, d.h. die Umsetzung einer Bewertung im Prozess, ob die Systemmerkmale des Systementwurfs die an das System gestellten Anforderungen erfüllen. Diese Anforderung ist dem Bewertungskriterium „Frühe Entwicklungsphasen“ und, bezogen auf das CMMI-DEV, dem Prozessgebiet Anforderungsentwicklung (RD) und der spezifischen Praktik SP 3.5 zugeordnet. Gemäß des V-Modells erfolgt die Eigenschaftsabsicherung in der Phase der Systemintegration (vgl. VDI 2206). Für die Phase des Systementwurfs enthält das vorgestellte Vorgehenskonzept keinen direkten Abgleich mit den Anforderungen, welche beispielsweise in Form des Lastenhefts vorliegen können. Das Vorgehenskonzept sieht jedoch die Validierung von Design Constraints durch die Verknüpfung mit den realisierenden Funktionen und/oder Komponenten vor. Darüber hinaus kann die Realisierungsrelation gezielt genutzt werden, um den vorhandenen

Anforderungen des Lastenhefts im Systemmodell die anforderungsrealisierenden Elemente zuzuordnen. Dies würde eine vollständige Erfüllung der abgeleiteten Anforderungen zur frühzeitigen Validierung darstellen. Da Anforderungen in den fokussierten Prozessschritten nicht explizit behandelt werden und demzufolge bis auf die Verknüpfung des Anforderungstyps Design Constraint kein Schritt zur Validierung der Anforderungsbasis enthalten ist, wird die Anforderung mit einem geringen Erfüllungsgrad (33 %) bewertet. Es ist jedoch festzuhalten, dass bereits durch die Anwendung der Systemmodellierung mit Beginn des Anforderungsmanagements der Grundstein für eine vollständige Anforderungserfüllung gelegt wäre und die Verknüpfung der Produktanforderungen mit dem Systementwurf durch die Relationsbestimmung proaktiv unterstützt wird.

Die abgeleitete Anforderung, Risiken frühzeitig zu identifizieren, bezieht sich auf das Prozessgebiet Risikomanagement (RSKM) und die spezifische Praktik SP 1.1 und ist ebenfalls den Bewertungskriterien mit Bezug zu frühen Entwicklungsphasen zugeordnet. Die im Vorgehenskonzept vorgesehene systematische Identifizierung und Analyse von Störgrößen stellt einen Ansatz dar, technische Risiken mit Bezug auf die Funktionsfähigkeit bzw. die Zuverlässigkeit des Systems bereits in der Phase des Systementwurfs zu identifizieren. Da das CMMI-DEV ein Risikomanagement fordert, welches „sowohl interne als auch externe und sowohl technische als auch nichttechnische Quellen von Kosten-, Termin-, Leistungs- und anderen Risiken“ [SEI 2011, S. 358] berücksichtigt, wird bezogen auf die Anwendungsfelder nur ein Teilbereich potentieller Risiken erfasst. Dementsprechend wird die zugrundeliegende Anforderung mit einer geringen Erfüllung (33 %) bewertet.

Inhaltlich stark an die Forderung der frühzeitigen Risikobewertung angelehnt ist die abgeleitete Anforderung, eine Risikomanagementstrategie frühzeitig festzulegen, welche ebenfalls unter das Bewertungskriterium „Frühe Entwicklungsphasen“ fällt und eine Umsetzung der spezifischen Praktik 1.3 des Prozessgebiets Risikomanagement (RSKM) darstellt. Gemäß gängiger Definitionen bestehen Risikomanagementstrategien in der Risikovermeidung, -minimierung, -akzeptanz oder dem Risikotransfer [Versteegen 2003, S. 167]. Eine Risikomanagementstrategie, welche nicht nur technische Risiken abdeckt, sondern vollumfänglich sämtliche Risiken im Projekt umfasst, wie beispielsweise monetäre Risiken, Versorgungsrisiken oder terminliche Risiken, muss über die verschiedenen Fachbereiche eines Unternehmens festgelegt werden. Der im Vorgehenskonzept vorgestellte Ansatz fokussiert Produktentwicklungsprozesse, bei denen in Abhängigkeit festgestellter technischer Risiken der Umgang mit ihnen festgelegt wird. Um eine ganzheitliche Risikomanagementstrategie festzulegen, müssen darüber hinaus alle relevanten Unternehmensbereiche involviert werden, was das Vorgehenskonzept nicht umfasst. Dementsprechend resultiert eine geringe Anforderungserfüllung (33 %).

5.5.1.2 Schnittstellenidentifizierung und -analyse

Das Bewertungskriterium „Schnittstellenidentifizierung und -analyse“ umfasst sieben abgeleitete Anforderungen, wobei die Anforderung zur frühzeitigen Implementierung eines Schnittstellenmanagements aus dem Prozessgebiet Produktintegration (PI, SP 2.2) bereits mit einem hohen Erfüllungsgrad (67 %) bewertet wurde, da dieses zwei Bewertungskriterien zugeordnet ist. Für diese Anforderung erfolgt somit keine Neubewertung, sie fließt jedoch in die Prozessreifegradbewertung ein.

Eine weitere Anforderung aus dem Prozessgebiet Produktintegration (PI) wurde für die spezifische Praktik SP 2.1 abgeleitet. Demnach müssen Schnittstellen vollständig erfasst werden. Wie bereits für das Bewertungskriterium „Frühe Entwicklungsphasen“ bei der spezifischen Praktik SP 2.2 erläutert, fordert das CMMI-DEV neben der Erfassung technischer Schnittstellen auch die Berücksichtigung von Benutzerschnittstellen, welche nicht Betrachtungsgegenstand der vorliegenden Arbeit sind. Hinsichtlich der Vollständigkeit der Erfassung struktureller Relationen und technisch-physikalischer Schnittstellen bietet das Vorgehenskonzept hingegen einen umfassenden Ansatz, der Schnittstellen detailliert beschreibt, die Systemelemente über sie verknüpft und bei Bedarf tiefergehende Analysen fordert, wenn Störeinflüsse vermutet werden. Aufgrund der nicht vorhandenen Erfassung von Benutzerschnittstellen werden hinsichtlich der Anforderungserfüllung Abzüge vorgenommen, sodass letztlich eine hohe Anforderungserfüllung (67 %) resultiert.

Der Nachweis der Schnittstellenkompatibilität am Gesamtsystem bzw. an den Subsystemen wird durch das Prozessgebiet Produktintegration (PI) und die SP 3.3 gefordert. Das Vorgehenskonzept unterstützt die Schnittstellenkompatibilität durch die Erfassung, Analyse und Darstellung von Größen, welche zwischen Systembestandteilen existieren. Ein auf diese Weise entwickelter Systementwurf bzw. domänenspezifischer Entwurf ermöglicht in der Phase der Systemintegration die Überprüfung, ob die (technisch-physikalischen) Schnittstellen zueinander kompatibel sind oder ob problematische Störeinflüsse resultieren können. Der explizite Nachweis dieser Schnittstellenkompatibilität ist für die Phase des Systementwurfs jedoch nicht vorgesehen, weshalb die Anforderung mit einem geringen Erfüllungsgrad (33 %) bewertet wird.

Aus dem Prozessgebiet Technische Umsetzung (TS) und der spezifischen Praktik SP 1.2 ist die Forderung abgeleitet, dass Schnittstellen zwischen Produktbestandteilen beschrieben werden müssen. Über das Vorgehenskonzept werden Beziehungen zwischen Produktbestandteilen in strukturelle Relationen und technisch-physikalische Schnittstellen unterteilt. Die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse sieht in den Schritten „Relationen identifizieren“ und „Größen und Schnittstellen identifizieren“ diese geforderte Beschreibung vor. Zudem erfolgt über die Systemmodellierung die gezielte modelltechnische Darstellung der Relationen sowie der Schnittstellen in Symbol-, Graphen- oder Matrizendarstellung. Die Forderung zur Beschreibung der Schnittstellen wird daher im Prozess als vollständig erfüllt (100 %) angesehen.

Die Forderung zur Identifizierung aller externen und der wichtigen internen Schnittstellen (Prozessgebiet Technische Umsetzung (TS), SP 2.1) wird, wie bereits dargelegt, nur bedingt erfüllt, da externe Schnittstellen nicht identifiziert und analysiert werden. Bei den internen Schnittstellen erfolgt keine Gewichtung in wichtig oder unwichtig, da die Bewertung, ob Schnittstellen relevant sind, oftmals auf einer Vermutung basiert und somit fehleranfällig ist. Die Identifizierung interner Schnittstellen erfolgt vollumfänglich. Bezogen auf die Anforderung wird somit eine geringe Erfüllung erreicht (33 %).

Ebenfalls dem Prozessgebiet Technische Umsetzung (TS) und hierbei der spezifischen Praktik SP 2.3 zugeordnet ist die abgeleitete Anforderung, dass Schnittstellenkategorien anhand definierter Kriterien entwickelt werden müssen. Wie in Kap. 4.2.1.2.1 definiert, werden strukturelle Relationen über vier mögliche Relationsarten (Hierarchie, Abfolge, Realisierung, Erfordernis) und die technisch-physikalischen Schnittstellen über die Kategorien Energie-, Stoff- und

Signalflüsse differenziert. Diese Kategorisierung setzt die Forderung zur Verwendung definierter Kriterien vollständig um. Dementsprechend wird die abgeleitete Anforderung als vollständig erfüllt (100 %) bewertet.

Aus dem Prozessgebiet Anforderungsentwicklung (RD) und der spezifischen Praktik SP 2.3 folgt die Anforderung, dass identifizierte Schnittstellen genutzt werden müssen, um hieraus Anforderungen an das System abzuleiten. Diese Anforderung wird durch die Schrittfolge zur Konkretisierung des Lösungsraums sowie die Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse zum Teil erfüllt. Diese schreiben vor, dass im Fall identifizierter Störgrößen, welche sich aus Schnittstellen ergeben, Anforderungen in Form von Design Constraints abgeleitet werden müssen. Eine darüberhinausgehende Nutzung der Schnittstellen zur Ableitung von Anforderungen ist nicht vorgesehen, weshalb keine vollständige, jedoch eine hohe Anforderungserfüllung (67 %) resultiert.

5.5.1.3 Komplexität

Das Bewertungskriterium „Komplexität“ umfasst eine abgeleitete Anforderung, welche dem Prozessgebiet Technische Umsetzung (TS) und der spezifischen Praktik SP 1.1 zugeordnet ist. Demnach muss die Komplexität von Subsystemen bei der Auswahl von Lösungen berücksichtigt werden. Gemäß dem Fokus der vorliegenden Arbeit stellt die Komplexität technischer Systeme ein inhärentes Merkmal dar. Komplexität muss demnach beherrscht werden, wobei keine Aussage darüber getroffen wird, ob das Ziel im Umgang mit Komplexität beispielsweise deren Reduzierung darstellt. Es erfolgt auch keine Empfehlung, ob bei der Auswahl von Lösungen solche mit geringer oder hoher Komplexität zu favorisieren sind. Dennoch leistet insbesondere die Systemmodellierung einen Beitrag zur Komplexitätsbeherrschung und damit auch der Erfassung der Komplexität möglicher Lösungen. Die Anforderung wird daher als gering erfüllt (33 %) bewertet.

5.5.1.4 Zuverlässigkeit

Die Anforderungen des Bewertungskriteriums Zuverlässigkeit wurden aus dem Prozessgebiet Ursachenanalyse und -beseitigung (CAR) abgeleitet. Hierbei wurde aus den spezifischen Praktiken SP 1.2 und SP 2.1 eine gemeinsame Anforderung gebildet. Demnach müssen für relevante Ergebnisse die Ursachen ermittelt und geeignete Maßnahmen entwickelt und umgesetzt werden. Im CMMI-DEV werden hier unter Ergebnissen Fehler und Probleme verstanden, die einen negativen Einfluss auf die sog. Prozessleistung besitzen. Um deren Ursachen zu ermitteln, werden gemäß der Schrittfolge zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse die Schnittstellen analysiert. Dies sieht explizit eine Ursachenermittlung mittels Simulationen oder Berechnungsmodellen vor und wurde exemplarisch für die Simulation der Oberflächentemperatur vor und nach Optimierung durchgeführt. Sind die Ursachen ermittelt und Ereignisse hinsichtlich ihres Einflusses auf das System bewertet, sieht die Schrittfolge zur Konkretisierung des Lösungsraums die Definition abgeleiteter Anforderungen, sog. Design Constraints, sowie die Implementierung von Optimierungsmaßnahmen vor. Dies entspricht der abgeleiteten Anforderung, weshalb diese als vollständig erfüllt (100 %) bewertet wird.

Aus der spezifischen Praktik SP 2.2 ist die Anforderung abgeleitet, dass die Effektivität getroffener Maßnahmen gemessen und analysiert werden muss. Über die Schrittfolge zur Konkretisierung des Lösungsraums wird zwar die Implementierung von Optimierungsmaßnahmen

vorgegeben, eine explizite Messung der Effektivität ist hierbei jedoch nicht vorgesehen. Zwar ist es über die iterative Anwendung der Schrittfolgen möglich, auch die Effektivität von Optimierungsmaßnahmen hinsichtlich ihrer prinzipiellen Eignung und ggf. der Verursachung neuer Störgrößen zu bewerten, dies stellt allerdings keine quantitative Effektivitätsmessung und -analyse dar. Dementsprechend liegt eine geringe Anforderungserfüllung (33 %) vor.

Die spezifische Praktik SP 2.3 fordert die Dokumentation von Erkenntnissen der Ursachenanalyse, die als Lessons Learned für andere Projekte aufbereitet werden müssen. Während über die Schrittfolge für die Systemmodellierung die Darstellung von Relationen und Schnittstellen und somit die Dokumentation generell sichergestellt ist, ist es nicht vorgesehen, diese Erkenntnisse separat zu dokumentieren und anderen Projekten zur Verfügung zu stellen. Gleichwohl bietet die Systemmodellierung eine fundierte Dokumentationsgrundlage, welche durch andere Projekte genutzt werden kann. Da dies jedoch über die Schrittfolge nicht explizit umgesetzt ist, wird die Anforderung mit einem geringen Erfüllungsgrad (33 %) bewertet.

5.5.1.5 Reifegradabsicherung

Aus den beiden Prozessgebieten Quantitatives Projektmanagement (QPM, SP 1.1) und Projektverfolgung und -steuerung (PMC, SP 1.6) ist die Anforderung abgeleitet, dass geeignete Qualitätsziele definiert und auf einzelne Projektphasen heruntergebrochen werden müssen, um die Zielerreichung projektbegleitend zu überwachen. Die Bewertung von Qualitätszielen in einzelnen Projektphasen entspricht der Festlegung von Quality Gates im PEP, an denen der geforderte Projektfortschritt bzw. Produktreifegrad mit den vorhandenen Ergebnissen verglichen wird. Die Meilensteine, welche im Vorgehenskonzept nach dem Abschluss jeder Lösungsebene mit Ausnahme der nullten Ebene vorgesehen sind, stellen dabei Quality Gates dar. Diese sind allerdings global formuliert und nicht an projektspezifische Bedarfe und Rahmenbedingungen ausgerichtet. Da trotz der Einbindung von Quality Gates an definierten Meilensteinen das Vorgehenskonzept keine projektspezifischen Quality Gates vorsieht und diese, wenn durch das Projekt definiert, nicht ohne weiteres in das Vorgehenskonzept eingebunden werden können resultiert letztlich eine hohe Anforderungserfüllung (67 %).

Die Anforderung, Meilensteine im Projekt zu definieren, an denen die Produktreife bewertet wird, um hieraus Entscheidungen über den weiteren Projektablauf abzuleiten, ergibt sich über die spezifischen Praktiken SP 1.3 und SP 2.1 des Prozessgebiets Projektplanung (PP) sowie der SP 1.7 des Prozessgebiets Projektverfolgung und -steuerung. Diese Anforderung ist wie zuvor dargestellt über die Meilensteinfestlegung nach dem Abschluss des Durchlaufs der Schrittfolge für jede Lösungsebene umgesetzt und wird daher als vollständig erfüllt (100 %) bewertet.

Aus dem Prozessgebiet Zulieferungsmanagement (SAM, SP 2.1) ist die Anforderung abgeleitet, den Reifegrad zugekaufter Subsysteme projektbegleitend zu überwachen. Das vorgestellte Vorgehenskonzept ist zwar prinzipiell geeignet, um dieses auch für zugekaufte Subsysteme anzuwenden, dies ist jedoch nicht über die Schrittfolge abgebildet, da die Lieferkette im Ansatz nicht fokussiert wird. Dementsprechend erfolgt keine Reifegradüberwachung von Subsystemen, welche durch Lieferanten entwickelt werden. Die zugehörige Anforderung ist daher nicht erfüllt (0 %).

Die Definition geeigneter Kriterien zur Durchführung einer Reifegradbewertung fordert die spezifische Praktik SP 2.1 des Prozessgebiets Technische Umsetzung (TS). Diese

Bewertungskriterien sind in Tab. 18 vorgestellt und beziehen sich auf die Schrittfolgen des Vorgehenskonzepts. Mittels dieser Kriterien wird der konkrete Produktreifegrad an definierten Meilensteinen erfasst und somit die Reifegradbewertung umgesetzt. Die Anforderung zur Definition geeigneter Kriterien ist hierdurch vollständig erfüllt (100 %).

5.5.1.6 Nicht kategorisierte Anforderungen

Neben den Anforderungen, welche für die Bewertungskriterien (Frühe Entwicklungsphasen, Schnittstellenidentifizierung und -analyse, Komplexität, Zuverlässigkeit und Reifegradabsicherung) abgeleitet wurden, konnte eine zusätzliche Anforderung identifiziert werden, welche aus Sicht des CMMI-DEV relevant für die Erreichung eines hohen Prozessreifegrads ist und eine wichtige Anforderung für das Vorgehenskonzept darstellt. Demnach muss eine funktionale Architektur gebildet werden, wie in der spezifischen Praktik SP 3.2 des Prozessgebiets Anforderungsentwicklung (RD) gefordert.

Die Schrittfolge für die Systemmodellierung sieht hierfür insbesondere die Verknüpfung der Funktionen über die Symboldarstellung vor, bei der Funktionsnetze gebildet werden, um die funktionale Architektur abzubilden. Über die Relationsbestimmung werden Funktionen zusätzlich über strukturelle Beziehungen miteinander verknüpft, um beispielsweise hierarchische oder abfolgeorientierte Relationen darzustellen. Die Bildung einer Funktionsarchitektur ist daher für die funktionsorientierte Produktentwicklung von zentraler Bedeutung und ein integraler Bestandteil des Vorgehenskonzepts. Entsprechend wird die abgeleitete Anforderung als vollständig erfüllt (100 %) angesehen.

5.5.2 Prozessreifegradbewertung

Zur Bewertung des Prozessreifegrads wird der Prozess, d.h. das vorgestellte Vorgehenskonzept, bewertet. Da hierbei der Reifegrad eines Ansatzes mit generischem Reifegradbezug bewertet wird, erfolgt die Bewertung ausschließlich auf Basis des in Kap. 4 vorgestellten Vorgehenskonzepts. Ob der Prozess wie gefordert angewendet wird, ist nicht Gegenstand der Bewertung, da die Umsetzung in einem konkreten Entwicklungsvorhaben unabhängig vom definierten Vorgehenskonzept ist.

Für die Prozessreifegradbewertung wird der Anforderungserfüllungsgrad der definierten und zuvor bewerteten Bewertungskriterien gemittelt. Sofern eine abgeleitete Anforderung Forderungen mehrerer Prozessgebiete oder Praktiken erfüllt, wird der Erfüllungsgrad separat gewertet. Tab. 21 fasst die Ergebnisse zusammen.

Tab. 21: Zusammenfassung der Anforderungserfüllung

Kriterium	Prozessgebiet und Praktik	Abgeleitete Anforderung	Erfüllungsgrad
Frühe Entwicklungsphasen	PI, SP 2.2	Ein Schnittstellenmanagement muss frühzeitig implementiert werden	67 %
	RD, SP 3.5	Anforderungen müssen frühzeitig validiert werden	33 %
	RSKM, SP 1.1	Risiken müssen frühzeitig identifiziert werden	33 %
	RSKM, SP 1.3	Eine Risikomanagementstrategie muss frühzeitig festgelegt werden	33 %
Schnittstellenidentifizierung und -analyse	PI, SP 2.1	Schnittstellen müssen vollständig erfasst werden	67 %
	PI, SP 2.2	Ein Schnittstellenmanagement muss frühzeitig implementiert werden[31]	67 %
	TS, SP 1.2	Schnittstellen zwischen Produktbestandteilen müssen beschrieben werden	100 %
	TS, SP 2.1	Es müssen alle externen Schnittstellen und die wichtigen internen Schnittstellen identifiziert werden	33 %
	TS, SP 2.3	Schnittstellenkategorien müssen anhand definierter Kriterien entwickelt werden	100 %
	RD, SP 2.3	Identifizierte Schnittstellen müssen genutzt werden, um hieraus Anforderungen an das System abzuleiten	67 %
	PI, SP 3.3	Die Schnittstellenkompatibilität muss am Gesamtsystem bzw. den Subsystemen nachgewiesen werden	33 %
Komplexität	TS, SP 1.1	Die Komplexität von Subsystemen muss bei der Auswahl von Lösungen berücksichtigt werden	33 %
Zuverlässigkeit	CAR, SP 1.2	Für relevante Ergebnisse müssen die Ursachen ermittelt werden und geeignete Maßnahmen entwickelt und umgesetzt werden	100 %
	CAR, SP 2.1		100 %
	CAR, SP 2.2	Die Effektivität getroffener Maßnahmen muss gemessen und analysiert werden	33 %
	CAR, SP 2.3	Erkenntnisse der Ursachenanalyse müssen dokumentiert und als Lessons Learned für andere Projekte aufbereitet werden	33 %
Reifegradabsicherung	QPM, SP 1.1	Es müssen geeignete Qualitätsziele definiert und auf einzelne Projektphasen heruntergebrochen werden, um die Zielerreichung projektbegleitend zu überwachen	67 %
	PMC, SP 1.6		67 %
	PP, SP 1.3	Es müssen Meilensteine im Projekt definiert werden, an denen die Produktreife bewertet wird, um hieraus Entscheidungen über den weiteren Projektablauf abzuleiten	100 %
	PMC, SP 1.7		100 %
	PP, SP 2.1		100 %
	SAM, SP 2.1	Der Reifegrad zugekaufter Subsysteme muss projektbegleitend überwacht werden	0 %
	TS, SP 2.1	Es müssen geeignete Kriterien definiert werden, mit denen eine Reifegradbewertung durchführbar ist	100 %
n/a	RD, SP 3.2	Es muss eine funktionale Architektur gebildet werden	100 %

[31] Diese Anforderung ist ebenfalls den frühen Entwicklungsphasen zugeordnet

Es zeigt sich, dass der Erfüllungsgrad der abgeleiteten Anforderungen stark divergiert. Während eine Anforderung nicht erfüllt und mit 0 % Anforderungserfüllung bewertet ist, existieren mehrere Anforderungen, welche eine vollständige Anforderungserfüllung (100 %) erreichen. Der resultierende Prozessreifegrad, der sich aus dem Mittelwert der einzelnen Erfüllungsgrade ergibt, liegt damit bei 65,25 % und entspricht als übergreifender Prozessreifegrad des Vorgehenskonzepts somit annähernd einem hohen Anforderungserfüllungsgrad.

Werden die Erfüllungsgrade für die sechs Bewertungskriterien jeweils im Mittelwert betrachtet, zeigen sich Unterschiede. So betragen die erreichten Partialprozessreifegrade für frühe Entwicklungsphasen 41,5 %, bei der Schnittstellenanalyse und -identifizierung 66,7 %, für Komplexität 33 % und für das Thema Zuverlässigkeit 66,5 %. Die Reifegradabsicherung liegt bei 76,3 % und die nicht-kategorisierten Anforderungen bei 100 %, wobei hier nur eine Anforderung zugrunde liegt. Während die Themen Zuverlässigkeit, Reifegradabsicherung, Schnittstellenidentifizierung und -analyse sowie die nicht-kategorisierte Anforderung eine hohe bzw. vollständige Anforderungserfüllung und somit einen hohen Prozessreifegrad besitzen, sind die erreichten Prozessreifegrade der Bewertungskriterien Frühe Entwicklungsphasen und Komplexität unterdurchschnittlich und erreichen nur eine geringe Anforderungserfüllung.

Die Ursache für die geringen Partialprozessreifegrade ist über den Fokus des Vorgehenskonzepts begründet. Werden Anforderungen nicht erfüllt, ist dies nicht gleichbedeutend mit einer geringen Eignung des Ansatzes. Stattdessen zeigt dies, dass das Vorgehensmodell weitere Aspekte umfassen muss, wie beispielsweise das Anforderungs-, Projekt- und Lieferantenmanagement, um einen ganzheitlichen Ansatz für die Produktentwicklung zu bieten. Mit Blick auf die Anwendung agiler Vorgehensweisen in der Produktentwicklung konstatiert das CMMI-DEV beispielsweise, dass diese „weder notwendig noch hinreichend sind, um das Prozessgebiet umzusetzen“ [SEI 2011, S. 73]. Konkrete Methoden und Vorgehensweise leisten zwar einen Beitrag zur Erreichung eines hohen Prozessreifegrads, hierbei ist jedoch kein bestimmter Ansatz vorgeschrieben. Das CMMI-DEV fordert daher vielmehr, dass gute Praktiken synergetisch kombiniert werden, um einen bestmöglichen Prozessreifegrad zu erreichen. Bedingt durch die Fokussierung der vorliegenden Arbeit wurden daher nicht alle Aspekte betrachtet, welche das CMMI-DEV als relevant definiert. Ein ganzheitlicher Ansatz müsste demzufolge sowohl die vom vorgestellten Vorgehenskonzept abgedeckten Bereiche umfassen, als auch weitere, für die Produktentwicklung relevante Sichtweisen und Unternehmensbereiche und -funktionen. Unter dem Fokus einer systemmodellbasierten Reifegradbewertung in den frühen Phasen einer funktionsorientierten Produktentwicklung konnte durch das Vorgehenskonzept dennoch ein Ansatz präsentiert werden, welcher die Schließung der methodischen Lücke zwischen Reifegradmodellen mit generischem Prozessbezug und Methoden der Reifegradabsicherung mit konkretem Produktbezug ermöglicht. Zur weiteren Steigerung des Prozessreifegrads ist dennoch erforderlich, den Ansatz um weitere Bestandteile zu ergänzen und somit aufzuzeigen, wie konkrete Schrittfolgen den Produktentwicklungsprozess unterstützen können und dabei in der Lage sind, generische Anforderungen bezüglich des Prozessreifegrads vollumfänglich zu erfüllen.

6 Fazit

6.1 Zusammenfassung

Für frühe Phasen der Produktentwicklung konnte über die Schrittfolge zur Systemkonkretisierung und -analyse gezeigt werden, wie Ansätze zur Verbesserung der Prozessreife mit denen zur Messung und Optimierung der Produktreife kombiniert werden können. Basierend auf den drei Säulen Systemkonkretisierung und -analyse, Systemmodellierung und Reifegradbewertung wurde hierfür ein Vorgehenskonzept entwickelt, welches technische Systeme über die Sichten Anforderungen, Funktionen und Komponenten modelliert und dabei gezielt strukturelle Relationen und technisch-physikalische Schnittstellen zwischen Systemelementen analysiert. Deren unzureichende Betrachtung bzw. die Unkenntnis über Systemzusammenhänge in frühen Entwicklungsphasen konnten zuvor als ein zentrales Problem der Produktentwicklung identifiziert werden. Im Kern wurden so die drei grundlegenden Herausforderungen adressiert: die insbesondere in frühen Entwicklungsphasen geringe Informationsverfügbarkeit über das System und seine Wechselwirkungen, die Komplexität, deren mangelnde Beherrschung ursächlich für Entwicklungsfehler ist, und die Zuverlässigkeit, für deren Erreichung bereits frühzeitig geeignete Analysen zur Identifizierung von Störgrößen implementiert werden müssen.

Bestehende Ansätze aus dem Bereich der Fähigkeits- und Reifegradmodelle leiten zwar Anforderungen an den Produktentwicklungsprozess ab, allerdings sind die Anforderungen mit generischem Prozessbezug zu Allgemeingültig, um hieraus konkrete Handlungsanleitungen abzuleiten. Ihnen gegenüber stehen Methoden der Reifegradabsicherung. Diese Ansätze mit konkretem Produktbezug leiten Forderungen ab, welche Kriterien erfüllt sein müssen, um einen Produktreifegrad zu ermitteln. Auch sie liefern kein Vorgehenskonzept, wie Produktentwicklungsprozesse gestaltet werden müssen. Als Reaktion auf diese methodische Lücke wurde das vorgestellte Vorgehenskonzept entwickelt, dessen Kern auf einer Verknüpfung einer Schrittfolge zur Systemkonkretisierung und -analyse mit dem angepassten DeCoDe-Systemmodell und einer meilensteinbezogenen Produktreifegradbewertung beruht.

Die Nutzung von Lösungsebenen unterstützt hierbei die sukzessive Systemkonkretisierung und stellt sicher, dass durch einen klar abgrenzbaren Entwicklungsfokus innerhalb einer Ebene die Komplexität beherrschbar bleibt. Durch die gezielte Erfassung und Analyse der Systembestandteile und der Beziehungen zwischen ihnen, werden systematisch Informationen über das System generiert. Diese liefern Ansatzpunkte für die Kopplung mit Methoden der Zuverlässigkeitsanalyse oder Berechnungen bzw. Simulationen, um so frühzeitig Störgrößen zu erkennen, zu bewerten und ihren Einfluss zu eliminieren oder bestmöglich zu reduzieren. Am Beispiel der LASM des DFG-Projekts Q-ELF konnte so aufgezeigt werden, wie eine Störgröße, die durch ein induziertes Wanderfeld verursachte Wärmeverlustleistung, gezielt identifiziert wird und als Reaktion hierauf Anforderungen abgeleitet und Optimierungsmaßnahmen implementiert werden. Das Vorgehenskonzept bietet dabei den Ausgangspunkt und die Schnittstelle zu Simulationen, um Störgrößen oder ihren Einfluss auf Zuverlässigkeitsparameter zu quantifizieren.

Die Bewertung des Produktreifegrads für die erste und zweite Lösungsebene zeigte, dass die Umsetzung der Schrittfolge zur Systemkonkretisierung und -analyse trotz unterschiedlicher Vertrauensintervalle jeweils 100 % erreichte. Die erreichbare Produktreife bezieht sich hierbei jedoch ausschließlich auf die Vollständigkeit des Systementwurfs zur jeweiligen Lösungsebene

und enthält keine Gegenüberstellung mit konkreten Produktanforderungen. Die Angabe eines Vertrauensintervalls, welches über die Anzahl erwarteter Ergebnisse, die als nicht anwendbar gekennzeichnet wurden, die getätigten Angaben kritisch hinterfragt, dient der Selbstkontrolle der Anwender. Eine vollständige Sicherheit lässt sich jedoch nicht erreichen, da beispielsweise alle erwarteten Ergebnisse vorliegen können, resultierend in einem grünen Vertrauensintervall, jedoch nur einige und nicht alle Störgrößen erfasst worden sein können. Zudem gilt die Bewertung ausschließlich für die methodische Produktreife, d.h. die Durchführung der Schrittfolgen. Für eine vollumfängliche Bewertung des Produktreifegrads ist zusätzlich die konkrete, auf der Produktspezifikation basierende Produktreife zu erfassen.

Über die Ermittlung des Prozessreifegrads konnte belegt werden, dass zentrale Anforderungen mit generischem Produktbezug durch das Vorgehenskonzept umgesetzt werden. Während das CMMI-DEV einen ganzheitlichen Ansatz unter Einbeziehung aller Unternehmensbereiche und insbesondere unter Berücksichtigung der Erfordernisse der Projektmanagements vorsieht, besitzt der in dieser Arbeit vorgestellte Ansatz einen anderen Fokus, welcher nicht alle Unternehmensbereiche und -prozesse abdeckt, sondern für frühe Entwicklungsphasen konzeptioniert ist. Für diesen Kernbereich konnte jedoch belegt werden, dass das Vorgehenskonzept eine Lösungsmöglichkeit zur konkreten Umsetzung bietet.

Für die Systemmodellierung hat sich das angepasste DeCoDe-Systemmodell als geeignet erwiesen und ermöglicht über Sichten Anforderungen, Funktionen und Komponenten eine standardisierbare Systembeschreibung, ohne die Beziehungen zwischen Systemelementen nicht transparent erfassbar wären. Die Kombination aus Graphen-, Matrizen- und Symboldarstellung ist zeitaufwändig, sichert aber die Dokumentation. Bei der Anwendung hat sich gezeigt, dass insbesondere die Symboldarstellung die Erfassung und Bewertung von Größen und Schnittstellen unterstützt hat. Die Nutzung der Symboldarstellung sollte somit als Ausgangspunkt der Modellierung im Entwicklungsteam genutzt werden, während die Graphen- und Matrizendarstellung in einem geeigneten Tool wie LOOMEO die Ergebnisse langfristig sichert. Wie zuvor erläutert, sind sowohl die Modellierungssprache als auch das Tool ein Mittel zum Zweck, weshalb die Anwendung anderer Modellierungssprachen zur Systemmodellierung und anderer Tools zu deren Visualisierung grundsätzlich möglich gewesen wäre.

In Summe leistet das Vorgehenskonzept einen Beitrag zur Lösung der beschriebenen Herausforderungen der Produktentwicklung. Die Schrittfolgen zur Konkretisierung des Lösungsraums sowie zur Relationsbestimmung und Schnittstellenidentifizierung und -analyse ermöglichen hierbei ein strukturiertes Vorgehen, welches Transparenz, Nachverfolgbarkeit und Reproduzierbarkeit sicherstellt. Während sich eine enge Verzahnung der Schrittfolgen als zielführend erwiesen hat, ist der sequentielle Ablauf für versierte Anwender weniger intuitiv und kann in Mehraufwand resultieren. Eine Aufweichung der Schrittfolge zugunsten einer intuitiveren Anwendbarkeit ist daher denkbar, sofern inhaltlich die Forderungen der Einzelschritte umgesetzt werden. Aufgrund des erforderlichen Mehraufwands für die Durchführung der Schrittfolge empfiehlt sich das Vorgehenskonzept eher für komplexe, insbesondere mechatronische Systeme, bei denen Entwicklungsteams verschiedener Domänen zusammenarbeiten.

6.2 Bewertung der Anforderungserfüllung durch das Vorgehenskonzept

Um die Eignung des Vorgehenskonzepts detailliert zu bewerten, erfolgt ein Abgleich mit den in Kap. 1.3 aufgestellten Anforderungen, vgl. Tab. 2. Demnach müssen durch das Vorgehenskonzept komplexitäts- und zuverlässigkeitsrelevante Informationen kategorisiert werden, um sie anschließend identifizieren zu können. Komplexitätsrelevante Informationen beziehen sich auf strukturelle Relationen zwischen Systemelementen, während sich zuverlässigkeitsrelevante Informationen auf Wechselwirkungen zwischen Systemelementen beziehen, also technisch-physikalische Schnittstellen. Beide Informationsarten wurden durch eine abschließende Beschreibung über die möglichen Relationsarten (Hierarchie, Abfolge, Realisierung, Erfordernis) und Schnittstellen (Energie-, Stoff-, Signalfluss) kategorisiert. Im Systemmodell wurden die jeweils relevanten Matrizen (DSM, DMM) benannt, für welche die Relationen und Schnittstellen existieren können. Die Anforderung wird somit durch das Vorgehenskonzept vollständig erfüllt (100 %). Gleiches gilt für die Erfassung und Modellierung der Systemelemente, Relationen und Schnittstellen. Über die Schrittfolge ist sichergestellt, dass eine Erfassung stattfindet und über das verknüpfte Systemmodell erfolgt die Modellierung.

Hinsichtlich der Kriterien für die Identifizierung von Störgrößen wurde definiert, wie Eingangs- und Ausgangsgrößen zu erfassen sind und welche Fälle existieren, für die eine Größe als Störgröße zu behandeln ist, vgl. Abb. 49: Störeinflüsse einer Größe. Für die quantitative Bewertung, ob ein Störeinfluss vorliegt, ist die Verknüpfung mit geeigneten Methoden oder Simulationen vorgesehen und wurde exemplarisch in die Validierung eingebunden. Die Anforderung wird daher ebenfalls als vollständig erfüllt (100 %) bewertet.

Bei der Auswahl und Adaption eines geeigneten Systemmodells wurde das DeCoDe-Modell ausgewählt und bedarfsgerecht angepasst, d.h. die Systemsichten wurden auf die relevanten Sichten Anforderungen, Funktionen und Komponenten reduziert und beschreibbare Relationen und Schnittstellen festgelegt. Der Fokus lag hierbei auf dem *Proof of Concept* des Vorgehenskonzepts anstatt der vergleichenden Bewertung von Systemmodellen oder Modellierungssprachen. Entsprechend erfolgte kein direkter Vergleich mit alternativen Modellen wie der SysML, welche im Bereich des Systems Engineering verbreitet ist. Aufgrund dieser Einschränkung wird ein hoher Erfüllungsgrad angenommen (67 %).

Die gewählten Visualisierungsformen der Modellierung, die Graphen-, Matrizen- und Symboldarstellung, sind bezogen auf Graph und Matrix generisch, da hier unmittelbar Vorgaben der Graphentheorie greifen. Detaillierte Vorgaben, wie beispielsweise die Benennung der Systemmatrizen oder die Festlegung zur ebenenübergreifenden Systembeschreibung, sind speziell auf die Belange des Vorgehenskonzepts zugeschnitten. Die Symboldarstellung stellt eine zusätzliche Granularisierung der Modellierung dar und zielt auf die optimale Darstellung von Größen ab. Allerdings gilt bei der Frage der Eignung die gleiche Einschränkung wie für das gewählte Systemmodell selbst: durch die Auswahl des DeCoDe-Modells ist die unmittelbare Umsetzbarkeit auch hinsichtlich der Darstellungsform nicht in einem Maße gegeben, wie sie eine etablierte Modellierungssprache ermöglicht hätte. Es liegt daher ein hoher Erfüllungsgrad (67 %) der Anforderung vor.

Die Forderung, dass das Vorgehenskonzept und das Systemmodell Meilensteine basierend auf den Lösungsebenen definieren muss, um an diesen eine Reifegradbewertung durchzuführen und bei Bedarf Optimierungsmaßnahmen zu implementieren, wird durch die Kombination der

Schrittfolge zur Systemkonkretisierung und -analyse, der Systemmodellierung und der Durchführung eines Quality Gates als Meilenstein nach dem Durchlauf der Schrittfolge je Lösungsebene umgesetzt. Die Anforderung wird daher als vollständig erfüllt (100 %) angesehen.

Hinsichtlich der zu entwickelnden Schrittfolge, welche konkrete Tätigkeiten vorgibt und Transparenz und Reproduzierbarkeit der Ergebnisse sicherstellt kann festgehalten werden, dass die im Vorgehenskonzept entwickelte Schrittfolge den generischen Anspruch durch eine Anwendbarkeit unabhängig vom zu entwickelnden Produkt erfüllt. Sowohl die Schrittfolge als auch die zugehörige Systemmodellierung sichern dabei die Transparenz des Entwicklungsergebnisses ab, indem die einzelnen Schritte je Lösungsebene in gleicher Art und Weise erfolgen und auch eine rekursive Betrachtung im Sinne des Reverse Engineerings ermöglichen. Wird die Schrittfolge konsequent durchgeführt, stellt dies ebenfalls die Reproduzierbarkeit sicher, zumindest in Bezug auf die Betrachtung von Relationen und Schnittstellen, nicht jedoch bezogen auf getroffene Entwicklungsentscheidungen zugunsten einer konkreten Lösung. Während die Anforderung bezüglich Transparenz und Reproduzierbarkeit als vollständig erfüllt (100 %) bewertet wird, trifft das Vorgehenskonzept hinsichtlich der Vorgabe von Entwicklungstätigkeiten generische Vorgaben bezogen auf den prinzipiellen Ablauf und die Analyse der Relationen und Schnittstellen, jedoch keine konkreten Vorgaben, wie der Lösungsraum durch eine erforderliche Konkretisierung einzuschränken ist. Folglich liegt ein geringer Erfüllungsgrad dieses Teilaspekts (33 %) vor. Da die gesamte Anforderung zu bewerten ist, resultiert für beide Aspekte in Summe eine hoher Erfüllungsgrad (67 %).

Die Anforderungen, dass das Vorgehensmodell auf generischen Systemsichten beruhen muss, wird vollständig erfüllt (100 %). Über die verschiedensten Ansätze bilden Anforderungen, Funktionen und Komponenten die wesentlichen Systemsichten für die Produktentwicklung. Diese Sichten lassen sich bedarfsgerecht erweitern, beispielsweise um Prozesse, welche für die vorliegenden Arbeit außerhalb des Fokus lagen.

Die Umsetzung der Best Practices des CMMI-DEV erfolgte durch eine Identifizierung relevanter Prozessgebiete bzw. den zugehörigen spezifischen Zielen und Praktiken. Aus diesen wurden Anforderungen abgeleitet, welche bereits hinsichtlich ihrer Anforderungserfüllung bewertet wurden, vgl. Tab. 21. Bezogen auf die 24 Praktiken resultiert ein Erfüllungsgrad von 65,25 %. Gerundet auf die Stufen des Erfüllungsgrads liegt eine hohe Anforderungserfüllung (67 %) vor.

Tab. 22: Erfüllung der Anforderungen durch das Vorgehenskonzept

Anforderungen	Erfüllungsgrad
Komplexitäts- und zuverlässigkeitsrelevante Informationen müssen kategorisiert werden, um identifiziert werden zu können	100 %
Systemelementen, Relationen und Schnittstellen müssen erfasst und modelliert werden	100 %
Kriterien für die Identifizierung von Störgrößen müssen definiert werden	100 %
Ein geeignetes Systemmodell muss ausgewählt und bei Bedarf adaptiert werden	67 %
Für die funktionsorientierte Produktentwicklung müssen geeignete Darstellungsformen gewählt werden, welche der Bedeutung der Funktionen Rechnung tragen	67 %
Vorgehenskonzept und Systemmodell müssen darauf ausgerichtet sein Lösungsebenen zur Definition geeigneter Meilensteine zu nutzen, den Reifegrad zu bewerten und bei Abweichungen vom geforderten Reifegrad Gegenmaßnahmen implementieren zu können	100 %
Es ist eine Schrittfolge zu entwickeln, welche wesentliche Entwicklungstätigkeiten im Sinne der Transparenz benennt und durch ihren generischen Ansatz die Reproduzierbarkeit von Entwicklungsergebnissen sicherstellt	67 %
Das Vorgehensmodell muss auf generischen Systemsichten beruhen	100 %
Best Practices des Reifegradmodells sind herauszuarbeiten und im Vorgehenskonzept umzusetzen	67 %

Es zeigt sich, dass die postulierten Anforderungen entweder vollständig oder mit einem hohen Erfüllungsgrad durch das Vorgehenskonzept erfüllt werden. Dabei liegen Abweichungen primär in der Auswahl des Systemmodells und der Modellierungssprache begründet. Hier wurde einer generischen Anwendbarkeit der Vorzug gegenüber einer praxisnahen Modellierungssprache gegeben, um die prinzipielle Eignung des Ansatzes nachzuweisen. Zudem fokussiert der Ansatz die Entwicklung eines Vorgehenskonzepts, welches basierend auf Lösungsebenen die modellbasierte Produktentwicklung in den frühen Phasen der Produktentwicklung unterstützen soll. Eine Vorgabe durchzuführender Entwicklungstätigkeiten erfolgt entsprechend unter dieser Prämisse. Die Unterstützung bei der Identifizierung von konkreten funktionellen Lösungsmöglichkeiten, Prinziplösungen oder gestalterischen Lösungen ist nicht Bestandteil dieser Arbeit, sondern die Absicherung der Lösungen im Sinne der Identifizierung, Analyse und Dokumentation von Systemelementen, Relationen und Schnittstellen.

Das Ziel des Ansatzes, einen Beitrag zur Schließung einer methodischen Lücke und zur synergetischen Einbindung bestehender Ansätze zu leisten, wird als erfüllt angesehen. Wie basierend auf dieser Erkenntnis das Vorgehenskonzept in andere Ansätze einzubinden wäre bzw. welche zusätzlichen Anwendungsfelder identifiziert werden können, beschreibt der Ausblick.

6.3 Ausblick

Die Anwendung des vorgestellten Ansatzes benötigt Entwicklungsressourcen, welche prinzipiell effektiv und effizient eingesetzt werden müssen, um innerhalb eines Projekts mit limitierten Ressourcen einen Mehrwert zu generieren. Demzufolge ist es zielführend, Anwendungsszenarien und -erweiterungen zu thematisieren, welche über den Fokus dieser Arbeit hinausgehen.

Obwohl der vorgestellte Ansatz die frühen Produktentwicklungsphasen fokussiert, ist das Vorgehenskonzept grundsätzlich über den gesamten PEP einsetzbar. Diese Erweiterung auf alle Phasen des V-Modells adressiert die Forderung von BERTSCHE ET AL., nach der zur Effizienzsteigerung Modelle auch in nachgelagerten Phasen des Systementwurfs anwendbar sein müssen [Bertsche et al. 2009, S. 53]. Zur Einbeziehung aller PEP-Phasen muss das Vorgehenskonzept nicht grundlegend geändert werden. Über diese Idee hinausgehend fordern SITTE/WINZER, dass Systeme ganzheitlich und über den gesamten Produktlebenszyklus abzubilden sind [Sitte/Winzer 2011]. Die Erweiterung des Ansatzes über die Produktentwicklung hinaus stellt insbesondere für die Nutzungsphase eine interessante Option dar. So unterliegen beispielsweise in der Automobilzulieferindustrie in Serienfertigung befindliche Produkte permanenten Änderungen, welche aus Kosteneinsparungsmaßnahmen oder der Abkündigung von elektronischen Standardbauteilen resultieren können. Im Rahmen des Engineering Change Managements müssen Änderungen erfasst, bewertet und qualifiziert werden, bevor sie für die Serienfertigung freigegeben werden können. Im Bereich der Hardware bestehen hierfür Ansätze wie die Delta Qualification Matrix[32] des ZVEI[33], welche Änderungen kategorisiert und standardisierte Prüfungen definiert. Betreffen Änderungen die Hardware oder mechanische Komponenten ist zu klären, wie sich die Änderungen auf das bestehende System auswirken. Eine vorhandene Systemmodellierung bietet hierfür einen geeigneten Ansatz, da die betroffenen Systemelemente unmittelbar über die vorhandenen Relationen und Schnittstellen identifiziert und bezüglich der Auswirkung der Änderung überprüft werden können. Das Vorgehenskonzepts kann hierfür den wesentlichen Input über das modellierte System inklusive der Relationen und Schnittstellen bereitstellen. Auch eine Fortführung der Relationsbeschreibung ist für das ECM sinnvoll, da hierüber beispielsweise Erfordernisrelationen und Design Constraints transparent beschrieben werden. Dies sichert sowohl die Beibehaltung von Elementen, deren Nutzen über eine Erfordernisrelation eindeutig beschrieben ist, als auch ein klares Kriterium, wann bestimmte Systembestandteile nicht mehr erforderlich sind, nämlich wenn sie keine Funktion realisieren oder sie als Reaktion auf ein entfallenes Design Constraint obsolet werden.

Der Ansatz zur Nutzung von Lösungsebenen kann ebenfalls im Hinblick auf die Idee des *Agile Development* einen Mehrwert bieten. Bei der dominanten Scrum-Methode werden Inkremente, d.h. Teilergebnisse, in sogenannten *Sprints* erarbeitet. Sprints besitzen einen festgelegten Zeitraum sowie einen definierten Entwicklungsumfang [Preußig 2018, S. 144ff]. Eine Kopplung mit agilen Methoden ließe sich durch die Vorgabe erreichen, dass je Sprint die Systemkonkretisierung über eine Lösungsebene zu generieren ist. Die bereits definierten Meilensteine zur Reifegradbewertung dienen dabei dem Sprint als Abschlusskriterium sowie als bereits implementiertes Review des Sprint-Ergebnisses. Da mit zunehmender Produktkonkretisierung auch der Umfang der erwarteten Ergebnisse zunimmt, ließen sich die Sprints zunächst auf die gesamte Systemkonkretisierung innerhalb einer Lösungsebene anwenden, während untergeordnete, umfangreichere Ebenen auf mehrere Sprints aufgeteilt werden können, welche dann das Ziel haben einzelne Aspekte der Schrittfolge wie die Schnittstellenanalyse umzusetzen.

Die grundsätzliche Frage, ob und wie das Vorgehenskonzept für die Belange der Produktentwicklung im industriellen Kontext zugeschnitten werden kann, sollte im Rahmen von

[32] https://www.zvei.org/en/subjects/mobility/product-process-change-notification-method-in-automotive-electronics, zuletzt aufgerufen am 18.04.2021
[33] Zentralverband Elektrotechnik- und Elektroindustrie e.V. (ZVEI)

Forschungs- und Entwicklungsprojekten gemeinsam durch Industrie und Wissenschaft evaluiert werden. Die unverändert hohen Anforderungen an Investitionsgüter sowie die fortbestehenden Herausforderungen bei der Entwicklung komplexer technischer Systeme unterstreichen jedoch die Notwendigkeit, bestehende Ansätze weiterzuentwickeln und neuen Ansätzen gegenüber aufgeschlossen zu sein. Das Vorgehenskonzept liefert hierfür einen ersten Ansatz, indem bestehende Methoden inkludiert werden, um einen Mehrwert über verschiedene PLC-Phasen hinaus zu schaffen.

7 Literaturverzeichnis

[Akkasoglu 2013] Akkasoglu, Gökhan: *Methodik zur Konzeption und Applikation anwendungsspezifischer Reifegradmodelle unter Berücksichtigung der Informationsunsicherheit*. Dissertation. Erlangen-Nürnberg: Technische Fakultät der Friedrich-Alexander-Universität, 2013.

[Andrásfai 1991] Andrásfai, B.: *Graph Theory: Flows, Matrices*. Bristol: Adam Hilger, 1991.

[Artischewski/ Sommerhoff 2015] Artischewski, Felix; Sommerhoff, Benedikt: *Qualitätssicherung 4.0*. In: Winzer, Petra (Hrsg.): Trends zur Handhabung von Komplexität im Qualitätsingenieurwesen. Aachen: Shaker Verlag, 2014. S. 55 – 68.

[Balla et al. 2020] Balla, Katalin; MiaoMiao, Tang; Mowat, Paul; Rasking, Matthias; Shang, Chaobo; van Veenendaal, Erik; Hongbao, Zhai: *Changes in CMMI 2.0 and how they can affect TMMi*. White paper TMMi Foundation, 2020. Quelle: https://www.tmmi.org/tm6/wp-content/uploads/2020/12/CMMI-2.0-whitepaper-V1.0.pdf, zuletzt aufgerufen am 05.05.2022.

[Bauer et al. 2012] Bauer, Frank; Gausemeier, Jürgen; Köchling, Daniel; Oestersötebier, Felix: *Simulative Absicherung mechatronischer Systeme in der frühen Phase der Produktentstehung*. In: Tag des Systems Engineerings 2012. Gesellschaft für Systems Engineering e.V. (GfSE). München: Carl Hanser Verlag, 2012. S. 197 – 206.

[Becker et al. 2009] Becker, J.; Knackstedt, R.; Pöppelbuß, J.: *Dokumentationsqualität von Reifegradmodellentwicklungen*. In: Becker, J.; Hellingrath, B.; Klein, S.; *Kuchen*, H.; Müller-Funk, U.; Vossen, G. (Hrsg.): Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 123. Münster: Westfälische Wilhelms-Universität, 2009

[Bender 2005] Bender, Klaus (Hrsg.): *Embedded Systems – qualitätsorientierte Entwicklung*. Berlin, Heidelberg: Springer, 2005.

[Bertsche et al. 2009] Bertsche, Bernd; Göhner, Peter; Jensen, Uwe; Schinköthe, Wolfgang; Wunderlich, Hans-Joachim: *Zuverlässigkeit mechatronischer Systeme. Grundlagen und Bewertung in frühen Entwicklungsphasen*. Berlin, Heidelberg: Springer-Verlag, 2009.

[Burghardt 2013] Burghardt, Manfred: *Einführung in das Projektmanagement. Definition, Planung, Kontrolle, Abschluss*. 6., überarbeitete und erweiterte Auflage, 2013. Erlangen: Publicis, 2013.

[CMMI Institute 2018] CMMI Institute: *CMMI V2.1 to V1.3 Practice Mapping*. CMMI Institute, 2018. Quelle: https://cmmiinstitute.com/getattachment/c191ea71-6614-405b-9368-6eb5047069f5/attachment.aspx, zuletzt aufgerufen am 11.06.2022.

[Crosby 1979] Crosby, Philip B.: *Quality is free*. New York: McGraw-Hill, 1979.

[Crostack/Klute 2008] Crostack, Horst-Artur; Klute, Sandra: *Die Bedeutung von Fehlerfolgekosten im Rahmen des Fehlermanagements*. In: ZWF Zeitschrift für wirtschaftlichen Fabrikbetrieb 06/2008, S. 404-406. München: Carl Hanser Verlag, 2008.

[Cusick 2020] Cusick, James J.: *A Survey of Maturity Models from Nolon to DevOps and Their Applications in Process Improvement*. 2020. Quelle: https://arxiv.org/pdf/1907.01878.pdf, zuletzt aufgerufen am 28.06.2022.

[Danner/Lindemann 1996] Danner, Stefan; Lindemann, Udo (Hrsg.): *Ganzheitliches Anforderungsmanagement für marktorientierte Entwicklungsprozesse*. Aachen: Shaker, 1996.

[DAT 2000] DAT Deutsche Automobil Treuhand GmbH: *DAT-Veedol-Report 2000*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2000/1559.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2001] DAT Deutsche Automobil Treuhand GmbH: *DAT-Veedol-Report 2001*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2001/1560.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2002] DAT Deutsche Automobil Treuhand GmbH: *DAT-Veedol-Report 2002*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2002/1561.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2003] DAT Deutsche Automobil Treuhand GmbH: *DAT-Veedol-Report 2003*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2003/1562.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2004] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2004*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/%202004/1563.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2005] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2005*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2005/1564.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2006] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2006*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2006/1565.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2007] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2007*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2007/1566.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2008] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2008*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2008/1567.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2009] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2009*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2009/2019.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2010] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2010*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2010/2674.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2011] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2011*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2011/3228.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2012] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2012*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2012/3965.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2013] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2013*. Quelle: http://www.dat.de/uploads/media/DAT-Report_2013_kfz-betrieb.pdf, zuletzt aufgerufen am 25.06.2013.

[DAT 2014] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2014*. Quelle: http://www.dat.de/uploads/DATReport_2014/pubData/source/804.pdf, zuletzt aufgerufen am 25.01.2015.

[DAT 2015] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2015*. Quelle: https://www.dat.de/fileadmin/media/download/DAT-Report_2015.pdf, zuletzt aufgerufen am 27.09.2016.

[DAT 2016] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2016*. Quelle: http://www.dat.de/fileadmin/media/download/DAT-Report/DAT-Report-2016.pdf, zuletzt aufgerufen am 27.09.2016.

[DAT 2017] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2017*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2017/8268.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2018] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2018*. Quelle: https://files.vogel.de/vogelonline/vogelonline/issues/kfz/sonst/2018/8803.pdf, zuletzt aufgerufen am 04.11.2019.

[DAT 2019] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2019*. Ostfildern: Deutsche Automobil Treuhand GmbH, 2019.

[DAT 2020] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2020*. Ostfildern: Deutsche Automobil Treuhand GmbH, 2020.

[DAT 2021] DAT Deutsche Automobil Treuhand GmbH: *DAT-Report 2021*.Ostfildern, Deutsche Automobil Treuhand GmbH, 2021.

[Eberlin/Hock 2014] Eberlin, Stefan; Hock, Barbara: *Zuverlässigkeit und Verfügbarkeit technischer Systeme. Eine Einführung in die Praxis*. Wiesbaden: Springer Vieweg, 2014.

[Delligatti 2014] Delligatti, Lenny: *SysML distilled. A brief guide to the systems modeling language*. Upper Saddle River, NJ, Boston, Indianapolis, San Francisco, New York, Toronto, Montreal, London, Munich, Paris, Madrid, Capetown, Sydney, Tokyo, Singapore, Mexico City: Addison-Wesley, 2014.

[Delonga 2007] Delonga, Melani: *Zuverlässigkeitsmanagement auf der Basis von Felddaten*. Dissertation. Stuttgart: Institut für Maschinenelemente, 2007.

[Dietz et al. 1998] Dietz, P.; Ort, A.; Penschke, S.: *Classification of Product Knowledge – An Approach to Optimal Feedback Strategies for Design*. In: Jacucci, Gianni; Olling, Gustav J.; Preiss, Kenneth; Wozny, Michael J. (Ed.): Globalization of Manufacturing in the Digital Communications Era of the 21st Century. Innovation, Agility and the Virtual Enterprise. Proceedings of the Tenth International IFIP WG5.2/5.3 International Conference PROLAMAT 98 Trento, ITALY September 9-11 & 12, 1998. New York: Springer Science+Business Media, 1998. S. 783 – 799.

[DIN 60050] DIN Deutsches Institut für Normung e.V.: *Internationales Elektrotechnisches Wörterbuch – Teil 351: Leittechnik* (IEC 1/2201/FDIS:2012). DIN IEC 60050-351:2013 (Entwurf). Berlin: Beuth Verlag, 2013.

[DIN 60812] DIN Deutsches Institut für Normung e.V.: *Analysetechniken für die Funktionsfähigkeit von Systemen – Verfahren für die Fehlzustandsart- und -auswirkungsanalyse (FMEA)* (IEC 60812:2006). DIN EN 60812:2006. Berlin: Beuth Verlag, 2006.

[DIN 61025] DIN Deutsches Institut für Normung e.V.: *Fehlzustandsbaumanalyse* (IEC 61025:2006). DIN EN 61025:2007. Berlin: Beuth Verlag, 2007.

[DIN 61078] DIN Deutsches Institut für Normung e.V.: *Techniken für die Analyse der Zuverlässigkeit – Zuverlässigkeitsblockdiagramm und Boole'sche Verfahren* (IEC 61078:2006). DIN EN 61078:2006. Berlin: Beuth Verlag, 2006.

[DIN 61165] DIN Deutsches Institut für Normung e.V.: *Anwendung des Markoff-Verfahrens* (IEC 61165:2006). DIN EN 61165:2007. Berlin: Beuth Verlag, 2007.

[DIN 61649] DIN Deutsches Institut für Normung e.V.: *Weibull-Analyse* (IEC 61649:2008). DIN EN 61649:2009. Berlin: Beuth Verlag, 2009.

[DIN EN 61160:2006] DIN Deutsches Institut für Normung e.V.: *Entwicklungsbewertung* (IEC 61160:2005). DIN EN 61160:2006. Berlin: Beuth Verlag, 2006.

[DIN EN ISO 9001:2000] DIN Deutsches Institut für Normung e.V.: *Qualitätsmanagementsysteme Anforderungen* (ISO 9001:2000). DIN EN ISO 9001:2000. Berlin: Beuth Verlag, 2000.

[DIN EN ISO 9000:2015] DIN Deutsches Institut für Normung e.V.: *Qualitätsmanagementsysteme – Grundlagen und Begriffe (ISO 9000:2015)*. DIN EN ISO 9000:2015. Berlin: Beuth Verlag, 2015.

[Dorociak 2012] Dorociak, Rafal: *Early Probabilistic Reliability Analysis of Mechatronic Systems*. In: Reliability and Maintainability Symposium (RAMS), 2012 Proceedings. 23-26 January 2012, pp. 1-6.

[Dourado et al. 2013] Dourado, João Paulo; Silva, Rui; Silva, Ângela: *Development of new Products using APQP and Quality Gates*. In: International Journal of Engineering and Industrial Management (IJEIM). No. 5 (2013). ISSN 1647-578X. S. 79 – 91. Quelle: http://repositorio.ulusiada.pt/bitstream/11067/1377/1/IJEIM_n5_5.pdf, zuletzt aufgerufen am 24.01.2016.

[Edler 2001] Edler, Andreas: *Nutzung von Felddaten in der qualitätsgetriebenen Produktentwicklung und im Service*. Dissertation. TU Berlin, 2001.

[Ehrlenspiel 2009] Ehrlenspiel, Klaus: *Integrierte Produktentwicklung. Denkabläufe, Methodeneinsatz, Zusammenarbeit*. 4., überarbeitete Auflage. München, Wien: Carl Hanser Verlag, 2009.

[Feldhusen/Grote 2021] Feldhusen, Jörg; Grote, Karl-Heinrich: *Qualitätssicherung in der Produktentwicklung und Konstruktion*. In: Bender, Beate; Gericke, Kilian (Hrsg.): Pahl/Beitz Konstruktionslehre. Methoden und Anwendung erfolgreicher Produktentwicklung. 9. Auflage. Berlin, Heidelberg: Springer Vieweg, 2021.

[Feldhusen/Grote 2013] Feldhusen, Jörg; Grote, Karl-Heinrich (Hrsg.): *Pahl/Beitz. Konstruktionslehre. Methoden und Anwendung erfolgreicher Produktentwicklung*. 8., vollständig überarbeitete Auflage 2013. Berlin, Heidelberg: Springer, 2013.

[Gausemeier 2010] Gausemeier, Jürgen.: *Frühzeitige Zuverlässigkeitsanalyse mechatronischer Systeme*. München: Hanser, 2010.

[Gericke et al. 2021] Gericke, Kilian; Bender, Beate; Pahl, Gerhard; Beitz, Wolfgang; Feldhusen, Jörg; Grote, Karl-Heinrich: *Der Produktentwicklungsprozess*. In: Bender, Beate; Gericke, Kilian (Hrsg.): Pahl/Beitz Konstruktionslehre. Methoden und Anwendung erfolgreicher Produktentwicklung. 9. Auflage. Berlin, Heidelberg: Springer Vieweg, 2021.

[Hentzschel/Jung 2002] Hentzschel, Jürgen; Jung, Harald: *Die Integration von Felddatenerfassung und Zuverlässigkeitsmanagement im Produktentstehungsprozess und Kundenservice. Anforderungen, Problembereiche und Lösungen*. In: VDI-Gesellschaft Systementwicklung und Projektgestaltung (Hrsg.): TTZ 2002 – Zuverlässige Produkte – Basis für hohe

Kundenzufriedenheit. Tagung Stuttgart 10. und 11. Oktober 2002. VDI-Berichte 1713. Düsseldorf: VDI Verlag, 2002. S. 139 – 160.

[Herrmann et al. 2013] Herrmann, Andrea; Knauss, Eric; Weißbach, Rüdiger (Hrsg.): *Requirements Engineering und Projektmanagement*. Berlin, Heidelberg: Springer-Vieweg, 2013.

[Isermann 2008] Isermann, Rolf: *Mechatronische Systeme. Grundlagen*. 2. Auflage. Berlin, Heidelberg: Springer-Verlag, 2008.

[Koller/Kastrup 1998] Koller, Rudolf; Kastrup, Norbert: *Prinziplösungen zur Konstruktion technischer Systeme*. 2. Auflage. Berlin, Heidelberg: Springer-Verlag, 1998.

[Kremer/Bertsche 2018] Kremer, Alexander; Bertsche, Bernd: *Beschleunigte Zuverlässigkeitstests unter Berücksichtigung von Vertrauensbereichen*. In: Forschung im Ingenieurwesen. Ausgabe 4/018. Ausgewählte Beiträge der 59. Tribologie-Fachtagung 2018 – Reibung, Schmierung. Verschleiß. Springer-Verlag, 2018. S. 395 – 409.

[Krehmer et al. 2009] Krehmer, Hartmut; Meerkamm, Harald; Wartzack, Sandro: *The Product's Degree of Maturity as a Measurement fort he Efficiency of Design Iterations*. In: International Conference on Engineering Design, ICED 09. 24. – 27. August 2009, Stanford University, USA. S. 181 – 192.

[Kreimeyer/Lindemann 2011] Kreimeyer, Matthias; Lindemann, Udo: *Complexity Metrics in Engineering Design. Managing the Structure of Design Processes*. Berlin, Heidelberg: Springer-Verlag, 2011.

[Kroll 2013] Kroll, Andreas: *Computational Intelligence. Eine Einführung in Probleme, Methoden und technische Anwendungen*. München: Oldenbourg Wissenschaftsverlag, 2013.

[Lashin 2021] Lashin, Gamal: *Technisches Änderungsmanagement*. In: Bender, Beate; Gericke, Kilian (Hrsg.): Pahl/Beitz Konstruktionslehre. Methoden und Anwendung erfolgreicher Produktentwicklung. 9. Auflage. Berlin, Heidelberg: Springer Vieweg, 2021.

[Lee 2003] Lee, Taesik: *Complexity Theory in Axiomatic Design*. Dissertation. Department of Mechanical Engineering, Massachusetts Institute of Technology, 2003.

[Leopold 2012] Leopold, Tobias: *Ganzheitliche Datenerfassung für verbesserte Zuverlässigkeitsanalysen*. Dissertation. Stuttgart: Institut für Maschinenelemente, 2012.

[Lindemann et al. 2009] Lindemann, Udo; Maurer, Maik; Braun, Thomas: *Structural Complexity Management. An Approach for the Field of Product Design*. Berlin, Heidelberg: Springer-Verlag, 2009.

[Linß 2011] Linß, Gerhard: *Qualitätsmanagement für Ingenieure*. 3., aktualisierte und erweiterte Auflage. München: Hanser, 2011.

[Magnus et al. 2008] Magnus, Kurt; Popp, Karl; Sextro, Walter: *Schwingungen. Eine Einführung in die physikalischen Grundlagen und die theoretische Behandlung von Schwingungsproblemen*. 8., überarbeitete Auflage. Wiesbaden: Vieweg+Teubner, 2008.

[Mamrot 2014] Mamrot, Michel: *Entwicklung eines Ansatzes zur modellbasierten Felddatenrückführung in die Produktentwicklung*. Dissertation. Reihe: Berichte zum Generic-Management, Band 01/2014. Aachen: Shaker Verlag, 2014.

[Mamrot et al. 2012] Mamrot, Michel; Marchlewitz, Stefan; Nicklas, Jan-Peter; Riekhof, Florian; Schlüter, Nadine; Seider, Gabriele; Winzer, Petra: *Begriffe im Kontext des Generic Systems Engineering-Ansatzes*. In: Winzer, Petra (Hrsg.): Generic Systems Engineering als Basis für die Weiterentwicklung des WGMK-Modells. Berichte zum Generic-Management, Band 02/2012. Aachen: Shaker Verlag, 2012. Seite 21 – 30.

[Marenbach et al. 2020] Marenbach, Richard; Jäger, Johann; Nelles, Dieter: *Elektrische Energietechnik. Grundlagen, Energieversorgung, Antriebe und Leistungselektronik*. 3., aktualisierte Auflage. Wiesbaden: Springer Vieweg, 2020.

[Maurer 2007] Maurer, Maik S.: *Structural Awareness in Complex Product Design*. Dissertation. Fakultät für Maschinenwesen, Technische Universität München. München: Verlag Dr. Hut, 2007.

[Mayer-Bachmann 2007] Mayer-Bachman, Roland: *Integratives Anforderungsmanagement: Konzept und Anforderungsmodell am Beispiel der Fahrzeugentwicklung*. Dissertation. Karlsruhe: Universitätsverlag, 2007.

[Meinsen 2003] Meinsen, Stefan: *Konstruktivistisches Wissensmanagement. Wie Wissensarbeiter ihre Arbeit organisieren*. Weinheim, Basel, Berlin: Beltz Verlag, 2003.

[Meschede 2006] Meschede, Dieter (Hrsg.): *Gerthsen Physik*. 23. Auflage. Berlin, Heidelberg, New York: Springer, 2006.

[Mescheder/Sallach 2012] Mescheder, Berhard; Sallach, Christian: *Wettbewerbsvorteile durch Wissen. Knowledge Management, CRM und Change Management verbinden*. Berlin, Heidelberg: Springer-Verlag, 2012.

[Meyer/Reher 2016] Meyer, Helga; Reher, Heinz-Josef: *Projektmanagement. Von der Definition über die Projektplanung zum erfolgreichen Abschluss*. Wiesbaden: Springer Gabler, 2016.

[Meyna/Pauli 2010] Meyna, Arno; Pauli, Bernhard: *Zuverlässigkeitstechnik: Quantitative Bewertungsverfahren*. 2., überarbeitete und erweiterte Auflage. München, Wien: Carl Hanser Verlag, 2010.

[Mistler 2021] Mistler, Marian: *Entwicklung eines Vorgehenskonzepts zum modellbasierten Anforderungsmanagement (Requirement Engineering und*

Requirements Management) für Organisationen – REMOt. Dissertation. Bergische Universität Wuppertal, 2021.

[Müller 2007] Müller, Marco: *Reifegradbasierte Optimierung von Entwicklungsprozessen am besonderen Beispiel der produktionsbezogenen Produktabsicherung in der Automobilindustrie*. Dissertation. Universität des Saarlandes. Schriftenreihe Produktionstechnik, Band 42. Saarbrücken: Lehrstuhl für Konstruktionstechnik/CAD, 2007.

[Müller 2013] Müller, Jörg R.: *Ein präzises Terminologiegebäude der technischen Zuverlässigkeit*. In: VDI-Gesellschaft Produkt- und Prozessgestaltung (Hrsg.): Technische Zuverlässigkeit 2013. Entwicklung und Betrieb zuverlässiger Produkte. VDI-Berichte 2210. 26. Fachtagung Technische Zuverlässigkeit 2013. Leonberg bei Stuttgart, 23. Und 24. April 2013. Düsseldorf: VDI Verlag, 2013.

[Nollau 2009] Nollau, Reiner: *Modellierung und Simulation technischer Systeme. Eine praxisnahe Einführung*. Berlin, Heidelberg: Springer-Verlag, 2009.

[Pahl et al. 2021] Pahl, Gerhard; Beitz, Wolfgang; Gericke, Kilian; Bender, Beate; Feldhusen, Jörg; Grote, Karl-Heinrich: *Grundlagen technischer Systeme*. In: Bender, Beate; Gericke, Kilian (Hrsg.): Pahl/Beitz Konstruktionslehre. Methoden und Anwendung erfolgreicher Produktentwicklung. 9. Auflage. Berlin, Heidelberg: Springer Vieweg, 2021.

[Paulk et al. 1993] Paulk, Mark C.; Curtis, Bill; Chrissis, Mary Beth; Wever, Charles V.: *Capability Maturity Model^{SM} for Software, Version 1.1*. Technical Report CMU/SEI-93-TR-024. ESC-TR-93-177. February 1993

[Paulk 2009] Paulk, Mark C.: *A History of the Capability Maturity Model for Software*. In: *ASQ Software Quality Professional*, Vol. 12, No. 1, December 2009, pp. 5-19. Quelle: http://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.216.199; zuletzt aufgerufen am 03.04.2016.

[Patzak 1982] Patzak, Gerold: *Systemtechnik. Planung komplexer innovativer Systeme: Grundlagen, Methoden, Techniken*. Berlin: Springer, 1982.

[Pfeifer 2001] Pfeifer, Tilo: *Qualitätsmanagement. Strategien, Methoden, Techniken*. 3., völlig überarbeitete und erweiterte Auflage. München, Wien: Carl Hanser Verlag, 2001.

[Ponn/Lindemann 2011] Ponn, J.; Lindemann, U.: *Konzeptentwicklung und Gestaltung technischer Produkte. Systematisch von Anforderungen zu Konzepten und Gestaltlösungen*. 2. Auflage. Berlin, Heidelberg: Springer-Verlag, 2011

[Preußig 2018] Preußig, Jörg: *Agiles Projektmanagement. Agilität und Scrum im klassischen Projektumfeld.* 1. Auflage. Freiburg: Haufe Gruppe, 2018.

[Rakowsky/ Richardson 2001] Rakowsky, Uwe Kay; Richardson, Nicole: *Wörterbuch der Zuverlässigkeit*. Hagen: Life-Long Learning Verlag, 2001.

[Riekhof et al. 2012] Riekhof, Florian; Winzer, Petra; Wörner, Linus; Kulig, Stefan: *Funktionsorientierte Auslegung eines Linearantriebs*. In: Jumar, Ulrich; Schnieder, Eckehard; Diedrich, Christian (Hrsg.): Entwurf komplexer Automatisierungstechnik – EKA 2012. Beschreibungsmittel, Methoden, Werkzeuge und Anwendungen. 12. Fachtagung mit Tutorium und Toolausstellung. Magdeburg: ifak Institut für Automation und Kommunikation e.V., 2012.

[Riekhof et al. 2013a] Riekhof, Florian; Bielefeld, Ovidiu; Esser, Christian; Winzer, Petra: *Anforderungen an intralogistische Anlagen*. In: ZWF – Zeitschrift für wirtschaftlichen Fabrikbetrieb. 108. Jahrgang, Heft 10/2013. München: Carl Hanser Verlag, 2013. S. 753 – 757.

[Riekhof et al. 2013b] Riekhof, Florian; Winzer, Petra; Wörner, Linus; Kulig, Stefan: *Ansatz zur lösungsneutralen Funktionsmodellierung für das Komplexitätsmanagement und Zuverlässigkeitsanalysen bei mechatronischen Systemen*. In: VDI-Gesellschaft Produkt- und Prozessgestaltung: 26. Fachtagung Technische Zuverlässigkeit 2013. Entwicklung und Betrieb zuverlässiger Produkte. VDI-Berichte 2210. Düsseldorf: VDI Verlag, 2013.

[Riekhof/Winzer 2011] Riekhof, Florian; Winzer, Petra: *Zuverlässigkeits- und Innovationsgraderhöhung technischer Systeme durch die Nutzung eines erweiterten Funktionsverständnisses*. In: Petersen, Britgitte; Raab Verena (Hrsg.): Berichte zum Qualitätsmanagement Band 12/2011. Qualitätskommunikation. Bericht zur GQW-Jahrestagung 2011 in Bonn. Aachen: Shaker Verlag, 2011. S. 207 – 228.

[von Regius 2006] von Regius, Bernd: Qualität in der Produktentwicklung. Vom Kundenwunsch bis zum fehlerfreien Produkt. München, Wien: Carl Hanser Verlag, 2006.

[Schienmann 2002] Schienmann, Bruno: *Kontinuierliches Anforderungsmanagement: Prozesse – Techniken – Werkzeuge*. München: Addison-Wesley, 2002.

[Schlund 2011] Schlund, Sebastian: *Anforderungsaktualisierung in der Produktentwicklung. Entwicklung einer Methodik durch die Einbindung anforderungsrelevanter Ereignisse.* Berichte zum Generic-Management, Band 01/2011. Aachen: Shaker, 2011.

[Schlund et al. 2009] Schlund, Sebastian; Riekhof, Florian; Winzer, Petra: *Probleme bei der Entwicklung mechatronischer Systeme – Ergebnisse einer Industriebefragung*. In: ZWF – Zeitschrift für wirtschaftlichen Fabrikbetrieb, 01-02/2009. München: Carl Hanser Verlag, 2009. S. 54 – 59.

[Schlund/Winzer 2010] Schlund, Sebastian; Winzer, Petra: *DeCoDe-Modell zur anforderungsgerechten Produktentwicklung*. In: Bandow, Gerhard; Holzmüller, Helmut H. (Hrsg.): „Das ist gar keine Modell!“ Unterschiedliche Modelle und Modellierungen in Betriebswirtschaftslehre und Ingenieurwissenschaften. Wiesbaden: Gabler, 2010. S. 277 – 293.

[Schoeneberg 2014] Schoeneberg, Klaus-Peter (Hrsg.): *Komplexitätsmanagement in Unternehmen. Herausforderungen im Umgang mit Dynamik, Unsicherheit und Komplexität meistern*. Wiesbaden: Springer Gabler, 2014.

[Schuh et al. 2008] Schuh, Günther; Stölzle, Wolfgang; Straube, Frank (Hrsg.): *Anlaufmanagement in der Automobilindustrie erfolgreich umsetzen. Ein Leitfaden für die Praxis*. Berlin, Heidelberg: Springer-Verlag, 2008.

[SEI 2011] Software Engineering Institute (SEI): CMMI® für Entwicklung, Version 1.3. SEI-sanctioned GERMAN translation of CMMI-DEV, V1.3. CMMI Product Team. Prozessverbesserung für die Entwicklung besserer Produkte und Dienstleistungen. November 2011. Technical Report. Quelle: http://www.sei.cmu.edu/library/assets/whitepapers/10tr033de_v11.pdf, zuletzt aufgerufen am 15.11.2015.

[Siebertz et al. 2010] Siebertz, Karl; van Bebber, David; Hochkirchen, Thomas: Statistische Versuchsplanung. Design of Experiments (DoE). Berlin, Heidelberg: Springer, 2010.

[Sitte/Winzer 2011] Sitte, Joaquin; Winzer, Petra: *Demand Compliant Design*. In: IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans, Volume 41, No. 3 (2011).

[SN 29500-2] Siemens Norm SN 29500-2: *Ausfallraten Bauelemente, Teil 2: Erwartungswerte von integrierten Schaltkreisen*. Ausgabe 2004-12. Siemens AG, 2004.

[Ständer 2011] Ständer, Tobias: *Eine modellbasierte Methode zur Objektivierung der Risikoanalyse nach ISO 26262*. Dissertation. Technische Universität Carolo-Wilhelmina zu Braunschweig, 2011.

[Steger 2007] Steger, Angelika: *Diskrete Strukturen. Band 1: Kombinatorik, Graphentheorie, Algebra*. 2. Auflage. Berlin, Heidelberg: Springer, 2007.

[Taucher 2014] Taucher, Fritz: *Überholte Q-Methoden?* In: QZ – Qualität und Zuverlässigkeit. 59. Jahrgang, Heft 12/2014. München: Carl Hanser, 2014. Editorial.

[Turau 2009] Turau, Volker: *Algorithmische Graphentheorie*. 3. Auflage. München: Oldenburg Wissenschaftsverlag, 2009.

[Vajna et al. 1994] Vajna, Sándor; Weber, Christian; Schlingensiepen, Jürgen; Schlottmann, Dietrich: *CAD/CAM für Ingenieure. Hardware – Software – Strategien*. Braunschweig/Wiesbaden: Vieweg, 1994.

[VDA 2009] Verband der Automobilindustrie e. V. (VDA): *Das gemeinsame Qualitätsmanagement in der Lieferkette. Produktentstehung: Reifegradabsicherung für Neuteile. Methoden, Messkriterien, Dokumentationen*. 2. Überarbeitete Auflage, Oktober 2009. ISSN: 0943-9412.

[VDA 2023] VDA QMC Working Group 13: *Automotive SPICE Process Assessment / Reference Model*. Version 3.991. Draft status. Quelle: Automotive-SPICE-PAM-40-Gelbbandrelease.pdf (vda-qmc.de), zuletzt aufgerufen am 07.02.2024.

[VDI 2206] Verein Deutscher Ingenieure; VDI-Gesellschaft Entwicklung Konstruktion Vertrieb: *Entwicklungsmethodik für mechatronische Systeme*. VDI-Richtlinie 2206. Berlin: Beuth Verlag, 2004.

[VDI 2221] Verein Deutscher Ingenieure; VDI-Gesellschaft Entwicklung, Konstruktion, Vertrieb: *Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte*. VDI-Richtlinie 2221. Berlin: Beuth Verlag, 1993.

[VDI 2247] Verein Deutscher Ingenieure; VDI-Gesellschaft Entwicklung, Konstruktion, Vertrieb: *Qualitätsmanagement in der Produktentwicklung*. VDI-Richtlinie 2247 (Entwurf). Berlin: Beuth Verlag, 1994.

[Versteegen 2003] Versteegen, Gerhard (Hrsg.): *Risikomanagement in IT-Projekten*. Berlin, Heidelberg: Springer, 2013.

[Wedel et al. 2007] Wedel, Michael; Göhner, Peter; Gäng, Jochen; Bertsche, Bernd; Arnaout, Talal; Wunderlich, Hans-Joachim: *Domänenübergreifende Zuverlässigkeitsbewertung in frühen Entwicklungsphasen unter Berücksichtigung von Wechselwirkungen*. In: 5. Paderborner Workshop "Entwurf mechatronischer Systeme". HNI Verlagsschriftenreihe Band 210. Paderborn: HNI Verlag, 2007. S. 257-272.

[Westkämper 2006] Westkämper, Engelbert*: Einführung in die Organisation der Produktion*. Berlin, Heidelberg: Springer-Verlag, 2006.

[Wild 2007] Wild, Michael: *Ermittlungen von Interventionen für den Wissens- und Informationsaustausch mittels Intranet bei der RCB*. Norderstedt: Grin Verlag, 2007.

[Willing et al. 2014] Willing, Marén; Riekhof, Florian; Wörner, Linus: *Qualitätsorientierter Methodenworkflow für die Produktneuentwicklung eines Linearantriebs in der Fördertechnik*. In: Winzer, Petra (Hrsg.): Trends zur Handhabung von Komplexität im Qualitätsingenieurwesen. Berichte zum Generic Management, Band 2/2014. Aachen: Shaker Verlag, 2014. S. 19 – 42.

[Winzer 2014] Winzer, Petra: *Neue Tools braucht das Land!* In: QZ – Qualität und Zuverlässigkeit. 59. Jahrgang, Heft 12/2014. München: Carl Hanser, 2014. S. 16 – 17.

[Winzer 2016] Winzer, Petra: *Generic Systems Engineering – Ein methodischer Ansatz zur Komplexitätsbewältigung*. 2. Auflage. Berlin, Heidelberg: Springer Vieweg Verlag, 2016.

[Winzer et al. 2007] Winzer, Petra; Schlund, Sebastian; Kulig, Stefan; Rosendahl, Jens: *Methodischer Ansatz zur Anforderungsgerechten Entwicklung vernetzter mechatronischer Systeme in intralogistischen Anlagen*. In: SFB 696: Forderungsgerechte Auslegung von intralogistischen Systemen, Logistics on Demand, 2. Kolloquium am 10. Oktober 2007. Dortmund, Verlag Praxiswissen, 2007.

[Winzer et al. 2009] Winzer, Petra; Kulig, Stefan; Rosendahl, Jens; Schlund, Sebastian: *Methodenworkflow zur Entwicklung mechatronischer Systeme*. In: SFB 696: Forderungsgerechte Auslegung von intralogistischen Systemen. Logistics on Demand. 3. Kolloquium, 30. September 2009. Dortmund: Technische Universität Dortmund, 2009. S. 63 – 79.

[Wörner 2013] Wörner, Linus: *Ein heuristisches Verfahren zum automatisierten Eingrenzen des Lösungsraums in frühen Phasen der Produktentwicklung*. In: Winzer, Petra: Von der Produktentwicklung bis zur Business Excellence. Berichte zum Generic Management, Band 4/2013. Aachen: Shaker Verlag, 2013. S. 83 – 99.

[Wörner et al. 2014] Wörner, Linus; Kulig, Stefan; Willing, Marén; Winzer, Petra: *Genetic algorithm embedded into a quality-oriented workflow of methods for the development of a linear drive used in intralogistic systems*. In: Polish Academy of Sciences: Archives of Electrical Engineering. Vol. 63 (4). pp. 647 – 665. Poznan, 2014. Quelle: http://journals.pan.pl/dlibra/publication/98581/edition/85018/content/genetic-algorithm-embedded-into-a-quality-oriented-workflow-of-methods-for-the-development-of-a-linear-drive-used-in-intralogistic-systems-worner-linus-kulig-stefan-willing-maren-winzer-petra?language=en, zuletzt aufgerufen am 19.10.2020.

8 Anhang

Anhang 1: Bewertung des CMMI-DEV

Tab. 23: Bewertung des CMMI-DEV – SG 1

Akro-nym	Kategorie	Reife-grad	Prozessgebiet	Relevanz Prozess-gebiet	SG 1	Rele-vanz SG 1
CAR	Unter-stützung	5	Ursachenanalyse und -beseitigung	hoch	Ursachen für ausgewählte Ergebnisse ermitteln	gering
CM	Unter-stützung	2	Konfigurations-management	keine	Baselines etablieren	keine
DAR	Unter-stützung	3	Entscheidungs-findung	keine	Alternativen bewerten	keine
IPM	Projekt-management	3	Fortgeschrittenes Projektmanagement	keine	Projektspezifisch definierte Prozesse verwenden	keine
MA	Unter-stützung	2	Messung und Analyse	keine	Mess- und Analysetätigkeiten ausrichten	keine
OPD	Prozess-management	3	Organisationsweite Prozessentwicklung	keine	Prozess-Assests der Organisation etablieren	keine
OPF	Prozess-management	3	Organisationsweite Prozessausrichtung	keine	Prozessverbesserungs-möglichkeiten bestimmen	keine
OPM	Prozess-management	5	Organisationsweites Leistungs-management	keine	Betriebliche Leistung führen	keine
OPP	Prozess-management	4	Organisationsweite Prozessleistung	keine	Prozessleistungsbaselines und -modelle etablieren	keine
OT	Prozess-management	3	Organisationsweite Aus- und Weiterbildung	keine	Organisationsweite Fähigkeit zur Aus- und Weiterbildung etablieren	keine
PI	Entwicklung	3	Produktintegration	hoch	Produktintegration vorbereiten	keine
PMC	Projekt-management	2	Projektverfolgung und -steuerung	gering	Das Projekt gegenüber dem Plan überwachen	hoch
PP	Projekt-management	2	Projektplanung	hoch	Schätzungen etablieren	gering
PPQA	Unter-stützung	2	Prozess- und Produkt-Qualitätssicherung	keine	Arbeitsabläufe und -ergebnisse objektiv bewerten	keine
QPM	Projekt-management	4	Quantitatives Projektmanagement	gering	Quantitatives Management vorbereiten	gering
RD	Entwicklung	3	Anforderungs-entwicklung	hoch	Kundenanforderungen entwickeln	keine
REQM	Projekt-management	2	Anforderungs-management	keine	Anforderungen verwalten	keine
RSKM	Projekt-management	3	Risikomanagement	gering	Risikomanagement vorbereiten	hoch
SAM	Projekt-management	2	Zulieferungs-management	gering	Vereinbarungen mit Lieferanten treffen	keine
TS	Entwicklung	3	Technische Umsetzung	hoch	Lösungen für Produktbestandteile auswählen	hoch
VAL	Entwicklung	3	Validierung	keine	Validierung vorbereiten	keine
VER	Entwicklung	3	Verifizierung	keine	Verifizierung vorbereiten	keine

Tab. 24: Bewertung des CMMI-DEV – SP 1.1

SP 1.1						
Bezeichnung	**SP 1.1 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
Ergebnisse für die Analyse auswählen	keine	nein	nein	nein	nein	nein
Konfigurationseinheiten festlegen	keine	nein	nein	nein	nein	nein
Richtlinien zur Entscheidungsanalyse etablieren	keine	nein	nein	nein	nein	nein
Projektspezifisch definierte Prozesse etablieren	keine	nein	nein	nein	nein	nein
Messziele etablieren	keine	nein	nein	nein	nein	nein
Standardprozesse etablieren	keine	nein	nein	nein	nein	nein
Prozesserfordernisse der Organisation etablieren	keine	nein	nein	nein	nein	nein
Geschäftsziele unterstützen	keine	nein	nein	nein	nein	nein
Qualitäts- und Prozessleistungsziele etablieren	keine	nein	nein	nein	nein	nein
Strategischen Aus- und Weiterbildungsbedarf etablieren	keine	nein	nein	nein	nein	nein
Integrationsstrategie etablieren	keine	nein	nein	nein	nein	nein
Projektplanungsparameter überwachen	keine	nein	nein	nein	nein	nein
Umfang des Projekts abschätzen	keine	nein	nein	nein	nein	nein
Arbeitsabläufe objektiv bewerten	keine	nein	nein	nein	nein	nein
Projektziele etablieren	gering	nein	nein	nein	nein	ja
Bedürfnisse herausfinden	keine	nein	nein	nein	nein	nein
Anforderungen verstehen	keine	nein	nein	nein	nein	nein
Risikoquellen und -kategorien festlegen	gering	ja	nein	nein	nein	nein
Beschaffungsart festlegen	keine	nein	nein	nein	nein	nein
Alternative Lösungen und Auswahlkriterien entwickeln	gering	nein	nein	ja	nein	nein
Produkte zur Validierung auswählen	keine	nein	nein	nein	nein	nein
Arbeitsergebnisse zur Verifizierung auswählen	keine	nein	nein	nein	nein	nein

Tab. 25: Bewertung des CMMI-DEV – SP 1.2

SP 1.2						
Bezeichnung	**SP 1.2 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitts-tellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
Ursachen analysieren	gering	nein	nein	nein	ja	nein
Konfigurations-managementsysteme etablieren	keine	nein	nein	nein	nein	nein
Bewertungskriterien etablieren	keine	nein	nein	nein	nein	nein
Prozess-Assets der Organisation für die Planung der Projekttätigkeiten verwenden	keine	nein	nein	nein	nein	nein
Kennzahlen festlegen	keine	nein	nein	nein	nein	nein
Beschreibungen von Phasenmodellen etablieren	keine	nein	nein	nein	nein	nein
Prozesse der Organisation begutachten	keine	nein	nein	nein	nein	nein
Prozessleistungsdaten analysieren	keine	nein	nein	nein	nein	nein
Prozesse selektieren	keine	nein	nein	nein	nein	nein
Festlegen, welcher Aus- und Weiterbildungsbedarf in der Verantwortung der Organisation liegt	keine	nein	nein	nein	nein	nein
Produktintegrationsumgebung etablieren	keine	nein	nein	nein	nein	nein
Zusagen überwachen	keine	nein	nein	nein	nein	nein
Schätzungen der Attribute von Arbeitsergebnissen und Aufgaben etablieren	keine	nein	nein	nein	nein	nein
Arbeitsergebnisse objektiv bewerten	keine	nein	nein	nein	nein	nein
Definierte Prozesse zusammenstellen	keine	nein	nein	nein	nein	nein
Bedürfnisse der Stakeholder in Kundenanforderungen überführen	keine	nein	nein	nein	nein	nein
Zusagen zu Anforderungen einholen	keine	nein	nein	nein	nein	nein
Risikoparameter definieren	keine	nein	nein	nein	nein	nein
Lieferanten auswählen	keine	nein	nein	nein	nein	nein
Lösungen für Produktbestandteile auswählen	gering	nein	ja	nein	nein	nein
Validierungsumgebungen etablieren	keine	nein	nein	nein	nein	nein
Verifizierungsumgebung etablieren	keine	nein	nein	nein	nein	nein

Tab. 26: Bewertung des CMMI-DEV – SP 1.3

SP 1.3						
Bezeichnung	**SP 1.3 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
Baseline erstellen und freigeben	keine	nein	nein	nein	nein	nein
Alternative Lösungen identifizieren	keine	nein	nein	nein	nein	nein
Arbeitsumgebung des Projekts etablieren	keine	nein	nein	nein	nein	nein
Verfahren zur Datenerfassung und -speicherung identifzieren	keine	nein	nein	nein	nein	nein
Tailoring-Kriterien und Tailoring-Guidelines etablieren	keine	nein	nein	nein	nein	nein
Prozessverbesserungen der Organisation identifizieren	keine	nein	nein	nein	nein	nein
Mögliche Bereiche für Verbesserungen identifzieren	keine	nein	nein	nein	nein	nein
Kennzahlen zur Prozessleistung etablieren	keine	nein	nein	nein	nein	nein
Einen organisationsweiten operativen Aus- und Weiterbildungsplan etablieren	keine	nein	nein	nein	nein	nein
Verfahren und Kriterien zur Produktintegration etablieren	keine	nein	nein	nein	nein	nein
Projektrisiken überwachen	keine	nein	nein	nein	nein	nein
Projektphasen definieren	gering	nein	nein	nein	nein	ja
-	n/a	n/a	n/a	n/a	n/a	n/a
Teilprozesse und Attribute auswählen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Anforderungsänderungen verwalten	keine	nein	nein	nein	nein	nein
Strategie für das Risikomanagement etablieren	gering	ja	nein	nein	nein	nein
Vereinbarungen mit Lieferanten etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Verfahren und Kriterien zur Validierung etablieren	keine	nein	nein	nein	nein	nein
Verfahren und Kriterien zur Verifizierung etablieren	keine	nein	nein	nein	nein	nein

Tab. 27: Bewertung des CMMI-DEV – SP 1.4

SP 1.4						
Bezeichnung	**SP 1.4 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Bewertungsverfahren auswählen	keine	nein	nein	nein	nein	nein
Pläne integrieren	keine	nein	nein	nein	nein	nein
Analyseverfahren spezifizieren	keine	nein	nein	nein	nein	nein
Messablage der Organisation etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Prozessleistung analysieren und Prozessleistungsbaselines etablieren	keine	nein	nein	nein	nein	nein
Eine Fähigkeit zur Aus- und Weiterbildung etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Datenmanagement überwachen	keine	nein	nein	nein	nein	nein
Aufwand und Kosten schätzen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Kennzahlen und Analyseverfahren selektieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Bidirektionale Nachverfolgbarkeit von Anforderungen aufrechterhalten	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 28: Bewertung des CMMI-DEV – SP 1.5

SP 1.5						
Bezeichnung	**SP 1.5 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Alternative Lösungen bewerten	keine	nein	nein	nein	nein	nein
Das Projekt unter Verwendung integrierte Pläne managen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Bibliothek der Prozess-Assets der Organisation etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Prozessleistungsmodelle etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Einbeziehung von Stakeholdern überwachen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Abstimmung zwischen Projektarbeit und Anforderungen sicherstellen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 29: Bewertung des CMMI-DEV – SP 1.6

SP 1.6						
Bezeichnung	**SP 1.6 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Lösungen auswählen	keine	nein	nein	nein	nein	nein
Teams etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Standards zur Arbeitsumgebung etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Fortschrittsprüfungen durchführen	gering	nein	nein	nein	nein	ja
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 30: Bewertung des CMMI-DEV – SP 1.7

SP 1.7						
Bezeichnung	**SP 1.7 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Zu den Prozess-Assets der Organisation beitragen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Regeln und Richtlinien für Teams etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Meilensteinprüfungen durchführen	gering	nein	nein	nein	nein	ja
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 31: Bewertung des CMMI-DEV – SG 2

Akro-nym	Kategorie	Reife-grad	Prozessgebiet	Relevanz Prozess-gebiet	SG 2	Rele-vanz SG 2
CAR	Unter-stützung	5	Ursachenanalyse und -beseitigung	hoch	Ursachen für ausgewählte Ergebnisse angehen	hoch
CM	Unter-stützung	2	Konfigurations-management	keine	Änderungen verfolgen und lenken	keine
DAR	Unter-stützung	3	Entscheidungs-findung	keine	-	n/a
IPM	Projekt-management	3	Fortgeschrittenes Projektmanagement	keine	Koordination von und Zusammenarbeit mit relevanten Stakeholdern	keine
MA	Unter-stützung	2	Messung und Analyse	keine	Messergebnisse bereitstellen	keine
OPD	Prozess-management	3	Organisationsweite Prozessentwicklung	keine	-	n/a
OPF	Prozess-management	3	Organisationsweite Prozessausrichtung	keine	Prozessverbesserungen planen und umsetzen	keine
OPM	Prozess-management	5	Organisationsweites Leistungs-management	keine	Verbesserungen auswählen	keine
OPP	Prozess-management	4	Organisationsweite Prozessleistung	keine	-	n/a
OT	Prozess-management	3	Organisationsweite Aus- und Weiterbildung	keine	Aus- und Weiterbildung bereitstellen	keine
PI	Entwicklung	3	Produktintegration	hoch	Schnittstellenkompatibilität sicherstellen	hoch
PMC	Projekt-management	2	Projektverfolgung und -steuerung	gering	Korrekturmaßnahmen zum Abschluß führen	keine
PP	Projekt-management	2	Projektplanung	hoch	Projektpläne erstellen	gering
PPQA	Unter-stützung	2	Prozess- und Produkt-Qualitätssicherung	keine	Objektiven Einblick geben	keine
QPM	Projekt-management	4	Quantitatives Projektmanagement	gering	Projekte quantitativ führen	keine
RD	Entwicklung	3	Anforderungs-entwicklung	hoch	Produktanforderungen entwickeln	gering
REQM	Projekt-management	2	Anforderungs-management	keine	-	n/a
RSKM	Projekt-management	3	Risikomanagement	gering	Risiken erkennen und analysieren	keine
SAM	Projekt-management	2	Zulieferungs-management	gering	Vereinbarungen mit Lieferanten erfüllen	gering
TS	Entwicklung	3	Technische Umsetzung	hoch	Designs entwickeln	hoch
VAL	Entwicklung	3	Validierung	keine	Produkte oder Produktbestandteile validieren	keine
VER	Entwicklung	3	Verifizierung	keine	Peer-Reviews durchführen	keine

Tab. 32: Bewertung des CMMI-DEV – SP 2.1

SP 2.1						
Bezeichnung	**SP 2.1 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
Vorgeschlagene Maßnahmen umsetzen	gering	nein	nein	nein	ja	nein
Änderungsanträge verfolgen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Einbeziehung der Stakeholder managen	keine	nein	nein	nein	nein	nein
Messwerte ermitteln	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Pläne zur Prozessverbesserung etablieren	keine	nein	nein	nein	nein	nein
Verbesserungsvorschläge erarbeiten	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Aus- und Weiterbildungsmaßnahmen durchführen	keine	nein	nein	nein	nein	nein
Schnittstellenbeschriebungen auf Vollständigkeit prüfen	gering	nein	ja	nein	nein	nein
Problematische Punkte analysieren	keine	nein	nein	nein	nein	nein
Budget und Terminplan etablieren	gering	nein	nein	nein	nein	ja
Abweichungen kommunizieren und beseitigen	keine	nein	nein	nein	nein	nein
Leistung ausgewählter Teilprozesse überwachen	keine	nein	nein	nein	nein	nein
Anforderungen an Produkte und Produktbestandteile etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Risiken erkennen	keine	nein	nein	nein	nein	nein
Vereinbarungen mit Lieferanten ausführen	gering	nein	nein	nein	nein	ja
Produkte oder Produktbestandteile entwerfen	mittel	nein	ja	nein	nein	ja
Validierung durchführen	keine	nein	nein	nein	nein	nein
Peer-Reviews vorbereiten	keine	nein	nein	nein	nein	nein

Tab. 33: Bewertung des CMMI-DEV – SP 2.2

SP 2.2						
Bezeichnung	**SP 2.2 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
Auswirkungen von umgesetzten Maßnahmen bewerten	gering	nein	nein	nein	ja	nein
Konfigurationseinheiten lenken	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Mit Abhängigkeiten umgehen	keine	nein	nein	nein	nein	nein
Messwerte analysieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Pläne zur Prozessverbesserung umsetzen	keine	nein	nein	nein	nein	nein
Verbesserungsvorschläge analysieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Aufzeichnungen über die Aus- und Weiterbildungsmaßnahmen etablieren	keine	nein	nein	nein	nein	nein
Schnittstellen managen	mittel	ja	ja	nein	nein	nein
Korrekturmaßnahmen ergreifen	keine	nein	nein	nein	nein	nein
Projektrisiken erkennen	keine	nein	nein	nein	nein	nein
Aufzeichnungen etablieren	keine	nein	nein	nein	nein	nein
Projektleistung managen	keine	nein	nein	nein	nein	nein
Anforderungen an Produktbestandteile zuweisen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Risiken bewerten, kategorisieren und priorisieren	keine	nein	nein	nein	nein	nein
Beschaffte Produkte abnehmen	keine	nein	nein	nein	nein	nein
Technische Datenpakete etablieren	keine	nein	nein	nein	nein	nein
Validierungsergebnisse analysieren	keine	nein	nein	nein	nein	nein
Peer-Reviews durchführen	keine	nein	nein	nein	nein	nein

Tab. 34: Bewertung des CMMI-DEV – SP 2.3

SP 2.3						
Bezeichnung	**SP 2.3 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
Daten der Ursachenanalyse aufzeichnen	gering	nein	nein	nein	ja	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Koordinierungsprobleme lösen	keine	nein	nein	nein	nein	nein
Daten und Ergebnisse speichern	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Verbesserungen validieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Wirksamkeit der Aus- und Weiterbildungsmaßnahmen bewerten	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Korrekturmaßnahmen managen	keine	nein	nein	nein	nein	nein
Datenmanagement planen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Ursachenanalyse durchführen	keine	nein	nein	nein	nein	nein
Schnittstellenanforderungen identifizieren	gering	nein	ja	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Überführung von Produkten sicherstellen	keine	nein	nein	nein	nein	nein
Schnittstellen unter Verwendung von Kriterien entwerfen	gering	nein	ja	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Daten aus Peer-Reviews analysieren	keine	nein	nein	nein	nein	nein

Tab. 35: Bewertung des CMMI-DEV – SP 2.4

SP 2.4						
Bezeichnung	**SP 2.4 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Ergebnisse kommunizieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Verbesserungen zum Rollout auswählen und umsetzen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Projektressourcen planen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Analysen bezüglich Herstellung, Beschaffung oder Wiederverwendung durchführen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 36: Bewertung des CMMI-DEV – SP 2.5

SP 2.5						
Bezeichnung	**SP 2.5 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Erforderliche Kenntnisse und Fähigkeiten planen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 37: Bewertung des CMMI-DEV – SP 2.6

SP 2.6						
Bezeichnung	**SP 2.6 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Einbeziehung von Stakeholdern planen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 38: Bewertung des CMMI-DEV – SP 2.7

SP 2.7						
Bezeichnung	**SP 2.7 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Projektpläne etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 39: Bewertung des CMMI-DEV – SG 3

Akro-nym	Kategorie	Reife-grad	Prozessgebiet	Relevanz Prozess-gebiet	SG 3	Rele-vanz SG 3
CAR	Unter-stützung	5	Ursachenanalyse und -beseitigung	hoch	n/a	n/a
CM	Unter-stützung	2	Konfigurations-management	keine	Integrität etablieren	keine
DAR	Unter-stützung	3	Entscheidungs-findung	keine	-	n/a
IPM	Projekt-management	3	Fortgeschrittenes Projektmanagement	keine	-	n/a
MA	Unter-stützung	2	Messung und Analyse	keine	-	n/a
OPD	Prozess-management	3	Organisationsweite Prozessentwicklung	keine	-	n/a
OPF	Prozess-management	3	Organisationsweite Prozessausrichtung	keine	Prozess-Assets der Organisation ausrollen und Lessons Learned	keine
OPM	Prozess-management	5	Organisationsweites Leistungs-management	keine	Verbesserungen ausrollen	keine
OPP	Prozess-management	4	Organisationsweite Prozessleistung	keine	-	n/a
OT	Prozess-management	3	Organisationsweite Aus- und Weiterbildung	keine	-	n/a
PI	Entwicklung	3	Produktintegration	hoch	Produktbestandteile zusammenbauen und das Produkt ausliefern	gering
PMC	Projekt-management	2	Projektverfolgung und -steuerung	gering	-	n/a
PP	Projekt-management	2	Projektplanung	hoch	Zusagen zu Plänen einholen	keine
PPQA	Unter-stützung	2	Prozess- und Produkt-Qualitätssicherung	keine	-	n/a
QPM	Projekt-management	4	Quantitatives Projektmanagement	gering	-	n/a
RD	Entwicklung	3	Anforderungs-entwicklung	hoch	Anforderungen analysieren und validieren	gering
REQM	Projekt-management	2	Anforderungs-management	keine	-	n/a
RSKM	Projekt-management	3	Risikomanagement	gering	Risiken abschwächen	keine
SAM	Projekt-management	2	Zulieferungs-management	gering	-	n/a
TS	Entwicklung	3	Technische Umsetzung	hoch	Produktentwürfe umsetzen	keine
VAL	Entwicklung	3	Validierung	keine	-	n/a
VER	Entwicklung	3	Verifizierung	keine	Ausgewählte Arbeitsergebnisse verifizieren	keine

Tab. 40: Bewertung des CMMI-DEV – SP 3.1

SP 3.1						
Bezeichnung	**SP 3.1 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
Aufzeichnungen zum Konfigurationsmanagement etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Prozess-Assets der Organisation ausrollen	keine	nein	nein	nein	nein	nein
Rollout planen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Bereitschaft der Produktbestandteile zur Integration bestätigen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Pläne mit Einfluss auf das Projekt überprüfen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Betriebskonzepte und Anwendungsszenarien etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Pläne zur Risikoabschwächung entwickeln	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Entwürfe umsetzen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Verifizierung durchführen	keine	nein	nein	nein	nein	nein

Tab. 41: Bewertung des CMMI-DEV – SP 3.2

SP 3.2						
Bezeichnung	**SP 3.2 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
Konfigurations-Audits durchführen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Standardprozesse ausrollen	keine	nein	nein	nein	nein	nein
Rollout managen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Produktbestandteile zusammenbauen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Aufgaben und verfügbare Ressourcen abgleichen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Definition erforderlicher Funktionalität und Qualitätsattribute etablieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Pläne zur Risikoabschwächung umsetzen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Produktbegleitende Dokumentation erstellen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Verifizierungsergebnisse analysieren	keine	nein	nein	nein	nein	nein

Tab. 42: Bewertung des CMMI-DEV – SP 3.3

SP 3.3						
Bezeichnung	**SP 3.3 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Umsetzung überwachen	keine	nein	nein	nein	nein	nein
Auswirkungen von Verbesserungen bewerten	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Zusammengebaute Produktbestandteile bewerten	gering	nein	ja	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
Zusagen zu Plänen einholen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Anforderungen analysieren	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 43: Bewertung des CMMI-DEV – SP 3.4

SP 3.4						
Bezeichnung	**SP 3.4 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Erfahrungen in die Prozess-Assets der Organisation aufnehmen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Produkte oder Produktbestandteile verpacken und ausliefern	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Anforderungen analysieren und abgleichen	keine	nein	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Tab. 44: Bewertung des CMMI-DEV – SP 3.5

SP 3.5						
Bezeichnung	**SP 3.5 Relevanz**	**Frühe Entwicklungs-phasen**	**Schnitt-stellen**	**Komplexität**	**Zuverlässig-keit**	**Reifegrad-absicherung**
n/a	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
Anforderungen validieren	gering	ja	nein	nein	nein	nein
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a
-	n/a	n/a	n/a	n/a	n/a	n/a

Anhang 2: Gegenüberstellung der abgeleiteten Anforderungen mit dem CMMI-DEV

Tab. 45: Gegenüberstellung der abgeleiteten Anforderungen mit dem CMMI-DEV (i)

Prozessgebiet, Praktik, Seite	Wortlaut der spezifischen Praktik [SEI 2011]	Abgeleitete Anforderung
PI, SP 2.2, S. 274	„Das Management der Schnittstellen von Produkten und Produktbestandteilen beginnt in einem frühen Stadium der Produktentwicklung“	Ein Schnittstellenmanagement muss frühzeitig implementiert werden
RD, SP 3.5, S. 348	„Die Anforderungsvalidierung erfolgt in einer frühen Phase der Entwicklung zusammen mit den Endanwendern [...]“	Anforderungen müssen frühzeitig validiert werden
RSKM, SP 1.1, S. 360	„Die frühe Identifizierung interner und externer Risikoquellen kann zur frühen Erkennung von Risiken führen. Risikoabschwächungspläne können dann früh im Projekt umgesetzt werden, um dem Auftreten von Risiken vorzubeugen oder die Konsequenzen ihres Auftretens zu reduzieren“	Risiken müssen frühzeitig identifiziert werden
RSKM, SP 1.3, S. 363	„Eine Risikomanagementstrategie sollte früh im Projekt entwickelt werden, damit relevante Risiken vorausschauend identifiziert und behandelt werden können“	Eine Risikomanagementstrategie muss frühzeitig festgelegt werden
PI, SP 2.1, S. 273	„Schnittstellendaten auf Vollständigkeit überprüfen und die vollständige Abdeckung aller Schnittstellen sicherstellen“	Schnittstellen müssen vollständig erfasst werden
TS, SP 1.2, S. 387	„Schnittstellen zwischen Produktbestandteilen werden beschrieben“	Schnittstellen zwischen Produktbestandteilen müssen beschrieben werden
TS, SP 2.1, S. 389	„Identifizierung wichtiger interner sowie aller externen Schnittstellen“	Es müssen alle externen Schnittstellen und die wichtigen internen Schnittstellen identifiziert werden
TS, SP 2.3, S. 393	„Schnittstellen von Produktbestandteilen anhand von etablierten Kriterien entwerfen“	Schnittstellenkategorien müssen anhand definierter Kriterien entwickelt werden
RD, SP 2.3, S. 343	„Anforderungen an die ermittelten Schnittstellen entwickeln“	Identifizierte Schnittstellen müssen genutzt werden, um hieraus Anforderungen an das System abzuleiten
PI, SP 3.3, S. 277	„Zusammengebaute Produktbestandteile auf Schnittstellenkompatibilität bewerten“	Die Schnittstellenkompatibilität muss am Gesamtsystem bzw. den Subsystemen nachgewiesen werden
TS, SP 1.1, S. 385	„Überlegungen zu alternativen Lösungen umfassen: [...] Komplexität des Produktbestandteils und der produktbezogenen Lebenszyklusprozesse“	Die Komplexität von Subsystemen muss bei der Auswahl von Lösungen berücksichtigt werden

Tab. 46: Gegenüberstellung der abgeleiteten Anforderungen mit dem CMMI-DEV (ii)

Prozessgebiet, Praktik, Seite	Wortlaut der spezifischen Praktik [SEI 2011]	Abgeleitete Anforderung
CAR, SP 1.2, S. 145f	„Ausgewählte Ergebnisse analysieren, um ihre Ursachen zu ermitteln“	Für relevante Ergebnisse müssen die Ursachen ermittelt werden und geeignete Maßnahmen entwickelt und umgesetzt werden
CAR, SP 2.1, S. 147	„Aus gewählte vorgeschlagene Maßnahmen umsetzen, die in der Ursachenanalyse entwickelt wurden“	
CAR, SP 2.2, S. 148	„Änderungen der Prozessleistung der betroffenen Prozesse oder Teilprozesse des Projekts messen und analysieren“	Die Effektivität getroffener Maßnahmen muss gemessen und analysiert werden
CAR, SP 2.3, S. 149	„Daten der Ursachenanalyse aufzeichnen und verfügbar machen, sodass andere Projekte angemessene Prozessänderungen durchführen und ähnliche Ergebnisse erzielen können“	Erkenntnisse der Ursachenanalyse müssen dokumentiert und als Lessons Learned für andere Projekte aufbereitet werden
QPM, SP 1.1, S. 320f	„Die messbaren Qualitäts- und Prozessleistungsziele für das Projekt festlegen und dokumentieren“ / „Zwischenziele ableiten, um den Fortschritt beim Erreichen der Projektziele zu beobachten“	Es müssen geeignete Qualitätsziele definiert und auf einzelne Projektphasen heruntergebrochen werden, um die Zielerreichung projektbegleitend zu überwachen
PMC, SP 1.6, S. 285	„Fortschritt, Arbeitsleistung und Probleme des Projekts regelmäßig überprüfen“	
PP, SP 1.3, S. 295	„Die Festlegung der Phasen eines Projekts sorgt für geplante Perioden der Bewertung und Entscheidungsfindung. Diese Perioden werden normalerweise so definiert, dass sie logische Entscheidungspunkte unterstützen [...]“	Es müssen Meilensteine im Projekt definiert werden, an denen die Produktreife bewertet wird, um hieraus Entscheidungen über den weiteren Projektablauf abzuleiten
PP, SP 2.1, S. 298	„Wichtige Meilensteine identifizieren“	
PMC, SP 1.7, S. 285	„Bisherige Umsetzung und Ergebnisse des Projekts an ausgewählten Projektmeilensteinen überprüfen“	
SAM, SP 2.1, S. 378f	„Den Fortschritt und die Leistung des Lieferanten (z.B. Terminplan, Aufwand, Kosten und technische Leistung) so überwachen, wie es in der Lieferantenvereinbarung festgelegt ist“ / „Die in der Lieferantenvereinbarung festgelegten technischen Prüfungen mit dem Lieferanten durchführen“	Der Reifegrad zugekaufter Subsysteme muss projektbegleitend überwacht werden
TS, SP 2.1, S. 390	„Kriterien zur Bewertung des Designs etablieren und beibehalten“	Es müssen geeignete Kriterien definiert werden, mit denen eine Reifegradbewertung durchführbar ist

Tab. 47: Gegenüberstellung der abgeleiteten Anforderungen mit dem CMMI-DEV (iii)

Prozessgebiet, Praktik, Seite	Wortlaut der spezifischen Praktik [SEI 2011]	Abgeleitete Anforderung
RD, SP 3.2, S. 345	„Eine Vorgehensweise zur Definition des erforderlichen Funktionsumfangs und der Qualitätsattribute besteht darin, Szenarien in einer sogenannten „funktionalen Analyse“ zu untersuchen, um zu beschreiben, was das Produkt tun soll. Diese Funktionsbeschreibung kann Aktionen, Abläufe, Eingangsgrößen, Ausgangsgrößen und andere Informationen, die die Verwendungsart des Produkts beschreiben, umfassen. Die daraus resultierenden Beschreibungen von Funktionen, ihrer logischen Gruppierung und ihrer Verknüpfung mit Anforderungen wird als „funktionale Architektur“ bezeichnet“	Es muss eine funktionale Architektur gebildet werden

Anhang 3: Bewertung des RGA-Modells

Tab. 48: Bewertung des RGA-Modells – RG0

		RG0				
Indikator	**Relevanz Indikator**	**Relevanz RG0**	**Schnitt-stellen**	**Komple-xität**	**Zuver-lässigkeit**	**Funkti-ons-nachweis**
Innovation / Konzept / Zuverlässigkeit	hoch	gering	ja	nein	nein	nein
Projektmanagement	gering	keine	nein	nein	nein	nein
Risikomanagement	gering	keine	nein	nein	nein	nein
Beschaffungsprozess	gering	keine	nein	nein	nein	nein
Lieferkette / Teilever-sorgung	keine	n/a	n/a	n/a	n/a	n/a
Änderungs-manage-ment	keine	n/a	n/a	n/a	n/a	n/a
Produktentwicklung	gering	n/a	n/a	n/a	n/a	n/a
Produktabsicherung	hoch	n/a	n/a	n/a	n/a	n/a
Prozessentwicklung	keine	n/a	n/a	n/a	n/a	n/a
Prozessabsicherung	keine	n/a	n/a	n/a	n/a	n/a
Produkt- und Pro-zessfreigabe (PPF)	keine	n/a	n/a	n/a	n/a	n/a
Absicherung Serie	keine	n/a	n/a	n/a	n/a	n/a

Tab. 49: Bewertung des RGA-Modells – RG1

		RG1				
Indikator	**Relevanz Indikator**	**Relevanz RG1**	**Schnitt-stellen**	**Komple-xität**	**Zuver-lässigkeit**	**Funkti-ons-nachweis**
Innovation / Konzept / Zuverlässigkeit	hoch	gering	nein	nein	ja	nein
Projektmanagement	gering	keine	nein	nein	nein	nein
Risikomanagement	gering	gering	ja	nein	nein	nein
Beschaffungsprozess	gering	gering	ja	nein	nein	nein
Lieferkette / Teilever-sorgung	keine	keine	nein	nein	nein	nein
Änderungs-manage-ment	keine	n/a	n/a	n/a	n/a	n/a
Produktentwicklung	gering	keine	nein	nein	nein	nein
Produktabsicherung	hoch	gering	nein	nein	nein	ja
Prozessentwicklung	keine	n/a	n/a	n/a	n/a	n/a
Prozessabsicherung	keine	n/a	n/a	n/a	n/a	n/a
Produkt- und Pro-zessfreigabe (PPF)	keine	n/a	n/a	n/a	n/a	n/a
Absicherung Serie	keine	n/a	n/a	n/a	n/a	n/a

Tab. 50: Bewertung des RGA-Modells – RG2

		RG2				
Indikator	**Relevanz Indikator**	**Relevanz RG2**	**Schnitt-stellen**	**Komple-xität**	**Zuver-lässigkeit**	**Funkti-ons-nachweis**
Innovation / Konzept / Zuverlässigkeit	hoch	n/a	n/a	n/a	n/a	n/a
Projektmanagement	gering	gering	ja	nein	nein	nein
Risikomanagement	gering	keine	nein	nein	nein	nein
Beschaffungsprozess	gering	keine	nein	nein	nein	nein
Lieferkette / Teilever-sorgung	keine	keine	nein	nein	nein	nein
Änderungs-manage-ment	keine	keine	nein	nein	nein	nein
Produktentwicklung	gering	keine	nein	nein	nein	nein
Produktabsicherung	hoch	hoch	ja	nein	nein	ja
Prozessentwicklung	keine	keine	nein	nein	nein	nein
Prozessabsicherung	keine	keine	nein	nein	nein	nein
Produkt- und Pro-zessfreigabe (PPF)	keine	keine	nein	nein	nein	nein
Absicherung Serie	keine	n/a	n/a	n/a	n/a	n/a

Tab. 51: Bewertung des RGA-Modells – RG3

		RG3				
Indikator	**Relevanz Indikator**	**Relevanz RG3**	**Schnitt-stellen**	**Komple-xität**	**Zuver-lässigkeit**	**Funkti-ons-nachweis**
Innovation / Konzept / Zuverlässigkeit	hoch	n/a	n/a	n/a	n/a	n/a
Projektmanagement	gering	n/a	n/a	n/a	n/a	n/a
Risikomanagement	gering	keine	nein	nein	nein	nein
Beschaffungsprozess	gering	n/a	n/a	n/a	n/a	n/a
Lieferkette / Teilever-sorgung	keine	keine	nein	nein	nein	nein
Änderungs-manage-ment	keine	n/a	n/a	n/a	n/a	n/a
Produktentwicklung	gering	gering	nein	nein	nein	ja
Produktabsicherung	hoch	gering	nein	nein	nein	ja
Prozessentwicklung	keine	keine	nein	nein	nein	nein
Prozessabsicherung	keine	keine	nein	nein	nein	nein
Produkt- und Pro-zessfreigabe (PPF)	keine	n/a	n/a	n/a	n/a	n/a
Absicherung Serie	keine	keine	nein	nein	nein	nein

Anhang 4: Gegenüberstellung der abgeleiteten Anforderungen mit dem RGA-Modell

Tab. 52: Gegenüberstellung der abgeleiteten Anforderungen mit dem RGA-Modell (i)

Reifegrad, Indikator, Seite	Wortlaut des Messkriteriums [VDA 2009]	Abgeleitete Anforderung
RG0, Innovation/Konzept/Zuverlässigkeit, S. 41	„Die Systemgrenzen der Innovation im Gesamtsystem sind definiert. Wechselwirkungen mit Schnittstellen im Gesamtfahrzeug sind bekannt“	Eine Systemgrenze muss definiert und Schnittstellen und Wechselwirkungen identifiziert werden
RG1, Beschaffungsprozess, S. 44	„Freigegebenes Lastenheft, inkl. Relevanter technischer Wechselwirkungen im System, liegt dem Lieferanten vor“	Schnittstellen und Wechselwirkungen müssen mit Lieferanten von Subsystemen besprochen werden
RG1, Beschaffungsprozess, S. 45	„Leistungsumfänge und Verantwortlichkeiten sind festgelegt; Schnittstellen der zu vergebenden Leistungsumfänge sind eindeutig definiert“	
RG1, Innovation/Konzept/Zuverlässigkeit, S. 47	„Garantiekosten-, Zuverlässigkeit- und Feldprognose liegen vor“	Zuverlässigkeitsprognosen müssen erstellt werden
RG1, Produktabsicherung, S. 48	„Die Funktionalität der Innovation im Gesamtsystem (Gesamtfahrzeug) ist nachgewiesen“	Ein Funktionsnachweis muss auf Ebene des Gesamtsystems erbracht werden
RG1, Produktabsicherung, S. 48	„Die Simulations- und Erprobungsplanung zum Nachweis der Produktqualität ist festgelegt“	Eine Planung der erforderlichen Simulationen und Tests für den Nachweis der Anforderungserfüllung muss erstellt werden
RG1, Risikomanagement, S. 49	„Schnittstellen zu anderen Projekten und Bauteilverwendungen sind geklärt“	Bei modularen Anwendungen müssen Schnittstellen mit anderen Projekten identifiziert und bewertet werden
RG2, Projektmanagement, S. 54	„Schnittstellen zu anderen Projekten und Bauteilverwendungen beim Lieferanten und Kunden sind geklärt“	
RG2, Produktabsicherung, S. 56	„Eine Schnittstellen-Analyse (z.B. System-FMEA) für priorisierte Leistungs- und Lieferumfänge ist durchgeführt“	Schnittstellen müssen analysiert werden
RG2, Produktabsicherung, S. 57	„Die Erprobungsplanung berücksichtigt alle im Lastenheft festgelegten Merkmale“	Tests zum Nachweis der Anforderungserfüllung müssen alle vereinbarten Merkmale umfassen
RG3, Produktentwicklung, S. 62	„Der Entwicklungsstand des Produkts erfüllt die Lastenheft- / Pflichtheftvorgaben, technische Spezifikationen sind freigegeben.“	Der Systementwurf muss hinsichtlich Anforderungserfüllung überprüft werden

Tab. 53: Gegenüberstellung der abgeleiteten Anforderungen mit dem RGA-Modell (ii)

Reifegrad, Indikator, Seite	Wortlaut des Messkriteriums [VDA 2009]	Abgeleitete Anforderung
RG3, Produktabsicherung, S. 64	„Funktionalität des Produkts (inkl. Innovationen) ist durch Lieferant nachgewiesen“ / Funktionalität des Produkts (inkl. Innovationen) ist im Gesamtfahrzeug (Gesamtsystem) durch den Kunden nachgewiesen“	Für das System und seine Subsysteme müssen Funktionsnachweise auf ihrer jeweiligen Ebene erbracht werden

PRODUKT
SICHERHEIT
QUALITÄT

BERGISCHE
UNIVERSITÄT
WUPPERTAL