Forschungsbericht 2026-11

# Ein methodischer Ansatz um heterogene und disziplinspezifische Partialmodelle für Produktvarianten und das Customizing im Flugzeugvorentwurf zu koppeln

Mara Fuchs

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Systemarchitekturen in der Luftfahrt
Hamburg

DLR Deutsches Zentrum für Luft- und Raumfahrt

**Forschungsbericht 2026-11**

# Ein methodischer Ansatz um heterogene und disziplinspezifische Partialmodelle für Produktvarianten und das Customizing im Flugzeugvorentwurf zu koppeln

Mara Fuchs

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Systemarchitekturen in der Luftfahrt
Hamburg

292 Seiten
134 Bilder
15 Tabellen
302 Literaturstellen

DLR Deutsches Zentrum für Luft- und Raumfahrt


*Herausgeber:*

Deutsches Zentrum
für Luft- und Raumfahrt e. V.
Wissenschaftliche Information
Linder Höhe
D-51147 Köln

ISSN 1434-8454
ISRN DLR-FB-2026-11
Erscheinungsjahr 2026
DOI: 10.57676/abnv-qt60

*Variantenmodellierung, Flugzeugkabine, Customizing, Model-based Systems Engineering, Kollaborativer Entwurf*

Mara FUCHS
DLR, Institut für Systemarchitekturen in der Luftfahrt, Hamburg

***Ein methodischer Ansatz um heterogene und disziplinspezifische Partialmodelle für Produktvarianten und das Customizing im Flugzeugvorentwurf zu koppeln***

Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg

In dieser Arbeit wird ein neuer methodischer, kollaborativer Ansatz für den Systementwurf vorgestellt. Mit den in dieser Arbeit entwickelten föderierten Partialmodellen wird die Auslegung von Systemvarianten über mehrere Fachdisziplinexperten im industriellen Kontext ermöglicht und somit ein partnerübergreifender Flugzeugvorentwurf realisiert.

Die Entwicklung neuartiger Flugzeugkabinen erfordert die enge Zusammenarbeit verschiedener Fachdisziplinen, die jeweils spezialisierte Modelle und Tools einsetzen. Die größte Herausforderung liegt in der Interoperabilität dieser heterogenen Modelle und Tools. Für eine ganzheitliche Betrachtung und die Bewertung von Auswirkungen über Subsystemgrenzen hinweg ist jedoch eine Kopplung der multidisziplinären Modelle notwendig, die durch fehlende Schnittstellen erschwert wird. Gleichzeitig führen Innovationen und Kundenwünsche zu einer größeren Variantenvielfalt in der Kabine, wodurch die Komplexität erhöht wird. Modellbasierte Ansätze aus dem Systems Engineering unterstützen zwar die Untersuchung von Produktvarianten, stoßen jedoch bei der kollaborativen Modellierung an Grenzen. Die häufig zum Einsatz kommenden Einzelmodelle weisen eine hohe interne Modellkomplexität auf, wodurch diese schwer erweiterbar und nur eingeschränkt flexibel nutzbar sind. Besonders im Kabinenentwurf und im Customizing, bei dem rund 80% der Bauteile von Zulieferern stammen, sind Anpassbarkeit und Erweiterbarkeit der Toolkette entscheidend, um externe Modelle und neue Technologien kontinuierlich integrieren zu können.

In dieser Arbeit wird daher eine praxisnahe, kollaborative Methode entwickelt, die eine modulare Variantenmodellierung sowohl auf Gesamtsystemebene als auch im Customizing ermöglicht und zugleich die Zusammenarbeit mit vielen multidisziplinären Partnern unterstützt. Der Forschungsansatz verfolgt dafür den Einsatz von föderierten Partialmodellen. Dabei wird das Gesamtsystem in seine Subsysteme zerlegt und diese Dekomposition auf die Modellebene übertragen. Jedes Subsystem wird durch ein Partialmodell modelliert und ausgelegt. Diese Partialmodelle werden als Black-Boxen betrachtet und verfügen daher über definierte Ein- und Ausgänge zum Austausch von Informationen bzw. Modellparametern. Die Variantenbildung erfolgt auf einer höheren Abstraktionsebene, um die Komplexität des Gesamtsystems besser zu bewältigen. Dabei wird das Gesamtsystem erst durch den Zusammenschluss spezifischer Partialmodelle vollständig definiert. Dadurch wird die Modularität gestärkt, da Varianten erst durch die Kombination mehrerer Partialmodelle generiert werden. Zeitgleich wird damit eine erhebliche Flexibilität im Systemdesign ermöglicht und der induzierten Modellkomplexität entgegengewirkt. Die Kopplung der Partialmodelle erfolgt über einen digital bereitgestellten Adapter. Durch die Black-Box-Betrachtung und die Kopplung über eine standardisierte Schnittstelle können die Partialmodelle beliebig heterogen modelliert werden. Diese neugeschaffene Interoperabilität ermöglicht hierdurch die Interaktion von deskriptiven und analytischen Modellen miteinander. Zur Überprüfung der Zielerreichung der erarbeiteten Methode als Unterstützung in der praktischen Modellierung von Produktvarianten, wurden im Rahmen der vorliegenden Arbeit in zwei Fallstudien einerseits die Bildung von Varianten auf Gesamtsystemebene im industriellen Kontext mit mehreren externen Partnern und andererseits die Anwendung der Methode für das Customizing in der Flugzeugkabine untersucht.

Zusammenfassend belegt die Arbeit, dass die Kopplung föderierter, disziplinspezifischer Partialmodelle über einen Adapter einen wirksamen Beitrag zur kollaborativen, multidisziplinären Auslegung von Kabinenvarianten leistet – insbesondere vor dem Hintergrund bisheriger Herausforderungen hinsichtlich mangelnder Interoperabilität und Modularität in der industriellen Praxis. Es wird deutlich, dass die hier erarbeite Methode den heutigen Flugzeugvorentwurf flexibler und agiler gestaltet. Durch die modulare Struktur lassen sich neue Systemkonfigurationen laufend erweitern und benötigte Analysefähigkeiten ergänzen. Somit wird

in der frühen Entwurfsphase die Untersuchung neuartiger Varianten und deren Systemarchitekturen über die Zulieferkette und auf Gesamtsystemebene umfassend ermöglicht.

*Variant Modeling, Aircraft Cabin, Customization, Model-based Systems Engineering, Collaborative Design*

*(Published in German)*

Mara FUCHS
German Aerospace Center (DLR), Institute of System Architectures in Aeronautics, Hamburg

***A methodological approach to link heterogeneous and discipline-specific partial models for product variants and customizing in preliminary aircraft design***

Helmut-Schmidt-University / University of the Federal Armed Forces Hamburg

This thesis presents a new methodological, collaborative approach to system design. The federated partial models developed herein enable the configuration of system variants across multiple disciplinary experts in an industrial context, thus realizing a cross-partner preliminary aircraft design.

The development of new aircraft cabins requires close collaboration between various disciplines, each of which uses specialized models and tools. The greatest challenge is the interoperability of these heterogeneous models and tools. However, a holistic view and the evaluation of effects across subsystem boundaries require the coupling of multidisciplinary models, which is complicated by the lack of interfaces. At the same time, innovations and customer requirements lead to a greater number of variants in the cabin, which increases complexity. Although model-based approaches from systems engineering support the investigation of product variants, they reach their limits when it comes to collaborative modeling. Since the modeling of variants is carried out within a single model, the resulting internal model complexity increases, thereby impeding extensibility and limiting overall flexibility. Particularly in cabin design and customizing, where around 80% of components come from suppliers, adaptability and expandability of the tool chain are crucial in order to be able to continuously integrate external models and new technologies.

Therefore, this thesis develops a practical, collaborative method that enables modular variant modeling both at overall system level and in customizing and at the same time supports cooperation with many multidisciplinary partners. The research approach pursues the use of federated partial models. The overall system is broken down into its subsystems and this decomposition is transferred to the model level. Each subsystem is modelled and designed by a partial model. These partial models have defined inputs and outputs for exchanging information or model parameters and are viewed as a black box. The creation of variants is carried out at a higher level of abstraction in order to better manage the complexity of the overall system. The overall system is only fully defined through the combination of specific partial models. This strengthens modularity, as variants are only generated by combining several partial models, which enables considerable flexibility in system design and counteracts the induced model complexity. The partial models are linked using a digitally implemented adapter. The partial models can be modelled heterogeneously using the black box approach, as the coupling takes place via a standardized interface. This improves interoperability, as descriptive and analytical models can interact with each other. In order to verify the effectiveness of the developed method as a support tool for the practical modeling of product variants, this thesis examined two case studies: the creation of variants at the overall system level in an industrial context with several external partners, and the application of the method for customizing aircraft cabins.

In summary, this thesis proves that the coupling of federated, discipline-specific partial models using an adapter makes an effective contribution to the collaborative, multidisciplinary design of cabin variants - especially in the context of previous challenges regarding a lack of interoperability and modularity in industrial practice. It has been shown that the developed method enables a more flexible and agile approach to preliminary aircraft design. The modular structure enables the ongoing expansion of new system configurations and the integration of required analysis capabilities. As a result, it is possible to comprehensively investigate novel variants and their system architectures across the supply chain and at the overall system level in the early design phase.

# Ein methodischer Ansatz um heterogene und disziplinspezifische Partialmodelle für Produktvarianten und das Customizing im Flugzeugvorentwurf zu koppeln

Der Fakultät für Maschinenbau und Bauingenieurwesen
der Helmut-Schmidt-Universität/ Universität der Bundeswehr Hamburg
zur Erlangung des akademischen Grades einer Doktor-Ingenieurin
genehmigte

DISSERTATION

von

Mara Fuchs

aus Göttingen

Hamburg 2026

| | |
|---|---|
| Referent: | Univ.-Prof. Dr.-Ing. Frank Mantwill |
| Korreferentin: | Prof. Dr.-Ing. Jutta Abulawi |
| | |
| Tag der mündlichen Prüfung: | 16.03.2026 |

Erst jetzt verstehe ich, dass sich selbst zu kennen nicht bedeutet,
alles zu erreichen und perfekt zu werden.
Es bedeutet zu wissen, was man kann und was man nicht kann.

Itachi Uchiha

# Vorwort

„Sometimes,“
said Pooh,
„the smallest things take up
the most room in your heart.“

Eine lange Reise geht zu Ende und ein lange verfolgtes persönliches Ziel wird Wirklichkeit. Auf diesem Weg haben mich viele mir wichtige Menschen begleitet. Entstanden ist die vorliegende Dissertation am Deutschen Zentrum für Luft- und Raumfahrt e.V. (DLR) in Kooperation mit der Professur für Maschinenelemente und Rechnergestützte Produktentwicklung (MRP) des Instituts für Konstruktions- und Fertigungstechnik an der Helmut-Schmidt-Universität, Universität der Bundeswehr Hamburg.

Mein erster Dank gilt meinem Doktorvater Univ.-Prof. Dr.-Ing. Frank Mantwill, der mir die Möglichkeit gegeben hat, meine Promotion als Externe durchzuführen und mich von Beginn an offen aufgenommen hat. Seine kontinuierliche Unterstützung, die vielen wertvollen Impulse sowie die Zeit, die er sich für meine fachliche Entwicklung genommen hat, haben diesen Weg maßgeblich geprägt. Dafür bin ich ihm sehr dankbar.
Mein weiterer Dank gilt meiner Zweitgutachterin Prof. Dr.-Ing. Jutta Abulawi. Ihre stetige Unterstützung - bereits weit vor Beginn meiner Promotion - war für mich von unschätzbarem Wert. Ihr Engagement und ihr Verständnis von gelebtem Netzwerkgedanken im Sinne von „girls support girls“ haben mich sehr beeindruckt und nachhaltig geprägt.
Univ.-Prof. Dr. Oliver Niggemann danke ich für die Übernahme des Vorsitzes der Promotionskommission.

Ans DLR bin ich damals durch Dr.-Ing. Björn Nagel gekommen, der mich ermutigt hat den Weg einer Promotion überhaupt erst in Betracht zu ziehen. Vielen Dank Björn für die Möglichkeiten, die du mir am DLR gegeben hast und dein Vertrauen in mich. Ein ganz herzlicher Dank geht auch an Prof. Dr.-Ing. Jörn Biedermann. Du warst für mich wie ein Mentor, der mich gleichzeitig gefördert und gefordert hat. Deine stetige Unterstützung und mich hin und wieder auch einfach mal machen zu lassen, hat mir unglaublich viel gegeben und mich beruflich wie menschlich stark wachsen lassen.

Ein nicht zu unterschätzender Faktor im beruflichen Alltag sind die Kolleginnen und Kollegen, die mir neben der fachlichen Zusammenarbeit auch viel Spaß im Arbeitsalltag bereitet haben. Dabei danke ich allen für die angenehme Atmosphäre und die enorme Hilfsbereitschaft. Besonders inspiriert haben mich Line Winkler, die stets genau wusste, was ich gerade brauche, Fabian Reimer für unsere Kakao- und Kaffeeaustauschrunden sowie für Kreativität und Humor seit unserer Zeit im „Europacenter-Exil“, Dr.-Ing. Christian Hesse für seine Gelassenheit, Ruhe und langjährige fachliche

Erfahrung, Dr.-Ing. Jan-Niclas Walther für wertvolle Einblicke darin, mit einer Idee im Kopf einfach anzufangen und diese umzusetzen, Denise Horn für ihren emotionalen Rückhalt sowie Prajwal Shiva Prakasha für seine Begeisterungsfähigkeit und die Faszination für die eigenen Forschungsthemen. Ein weiterer Dank gilt Yassine Ghanjaoui und Florian Beckert für ihre grandiosen studentischen Arbeiten, die diese Doktorarbeit unterstützt haben.

Ein weiterer Dank gilt den Projektpartnern, insbesondere Centerline Design, für den stetigen und offenen Austausch zu industriebezogenen Fragestellungen im Kontext meiner Forschung am DLR. Ein großes Dankeschön auch an die Doktorandenrunde beim MRP für den fachlichen Austausch - die unterschiedlichen Blickwinkel haben mich sehr inspiriert. Vielen Dank auch an ProTechnicale, die mir die Möglichkeit gegeben haben mich während der Promotion im Bereich Wissenschaftskommunikation und MINT-Nachwuchsförderung weiterzubilden.

Auf diesem Weg haben mich viele Freunde begleitet. Ich danke euch allen für eure vorsichtigen Nachfragen zum aktuellen Fortschritt und eure Aufmunterungen weiterzumachen. Manchmal sind es die kleinen Dinge, die den größten Impact haben. Daher möchte ich besonders hervorheben:
Niklas - Danke für die viele sportliche Ablenkung zwischendurch, um den Kopf frei zu bekommen, sei es unsere wöchentlichen Lauf-&-Schnack-Runden um die Alster oder gemeinsame Abendessen mit Annika und Finn.
Cathy - Spaziergänge, Matcha, Brunch... seit fast 20 Jahren begleiten wir das Leben des jeweils anderen. Es bedeutet mir viel, dich an meiner Seite zu wissen.
Jasmin - Ob gemeinsames Bachelorschauen mit Sushi oder die Umsetzung irgendwelcher Instagram-DIYs bei Kreativabenden mit indischem Essen - es war mir eine willkommene Ablenkung.
Micha - Deine ständige positive Aufmunterung und Pushs waren genau das, was ich brauche. Danke, dass du immer für einen da bist.
Heeschi - Ohne Trash-TV - ohne uns. Als kognitiver Ausgleich gibt es für mich kaum etwas Besseres als Crime-Podcasts oder die neueste Staffel Trash-TV auf RTL+.

Meinen Eltern Mariele und Michael danke ich von Herzen für die Wurzeln, die sie mir mitgegeben haben, um meinen eigenen Weg zu gehen sowie für ihre bedingungslose Unterstützung. Meine Schwester Pia hat mich dabei immer seelisch unterstützt. Für dein offenes Ohr und die vielen wertvollen Gespräche danke ich dir von Herzen!

Zu guter Letzt gebührt der größte Dank Finn. Deine Unterstützung lässt sich für mich schwer in Worte fassen. Danke, dass du mir gezeigt hast Erfolge zu feiern - egal wie klein sie sind - für meine Ziele und Bedürfnisse einzustehen und nach den Sternen zu greifen und mir nicht immer über alles den Kopf zu zerbrechen. Deinen fantastischen Humor und die Comic Reliefs habe ich zwischenzeitlich echt gebraucht.

Hamburg, April 2026
Mara Fuchs

# Inhaltsverzeichnis

# Abkürzungsverzeichnis

| Abkürzung | Bezeichnung |
|---|---|
| 2D | Zweidimensional |
| 3D | Dreidimensional |
| A4A | Airlines for America |
| act | Activity Diagram |
| ALH | Action Language Helper |
| ALM | Application Lifecycle Management |
| AP | Applikationsprotokolle |
| API | Application Programming Interface |
| APU | Auxiliary Power Unit |
| ARP | Aerospace Recommended Practice |
| ATA | Air Transport Association |
| bdd | Blockdefinitionsdiagramm |
| BWI | Betriebswissenschaftliches Zentrum der ETH Zürich |
| CAD | Computer-aided Design |
| CAx | Rechnerunterstütze Systeme |
| CDT | Conceptual Design Tool |
| CFD | Computational Fluid Dynamics |
| CPACS | Common Parametric Aircraft Configuration Schema |
| CS-25 | Certification Specification for Large Aeroplanes |
| CSV | Comma-Seperated Values |
| DEVS | Discrete Event System Specification |
| DigECAT | Digital Twin for Engine, Components and Aircraft Technologies |
| DLR | Deutsches Zentrum für Luft- und Raumfahrt e.V. |
| DMU | Digital Mock-up |
| DOORS | Dynamic Object Oriented Requirements System |
| DRM | Design Research Method |
| DS | Descriptive Study |
| DSM | Design Structure Matrix |
| DZ | Digitaler Zwilling |
| EASA | European Aviation Safety Agency |
| EIS | Entry Into Service |
| ET | ElementTree |
| EVA | Eingabe-Verarbeitung-Ausgabe |
| FAS | Functional Architecture for Systems Method |
| fbx | Filmbox |
| FEM | Finite-Elemente-Methode |

| Abkürzung | Bezeichnung |
|---|---|
| FODA | Feature Oriented Domain Analysis |
| fUML | Foundational UML |
| GATE | Green Aviation Technologies |
| GUI | Graphical User Interface |
| HoV | Head of Version |
| ibd | Internal Block Diagram |
| ID | Identifier |
| INCOSE | International Council on Systems Engineering |
| JSON | JavaScript Objekt Notation |
| KB | Kilobyte |
| KBE | Knowledge-based Engineering |
| KMU | Kleine und mittlere Unternehmen |
| LOPA | Layout of Passenger Accommodations |
| MBSE | Model-based Systems Engineering |
| MIWa | MBSE-basierte Integration & Variantenbildung von Wasserstoffkryo-drucktanksystemen zukünftiger Flugzeugkonfigurationen |
| MOSART | Multi-objective Optimization for Safety and Reliability Trade-off |
| MSN | Manufacturer Serial Numbers |
| OEM | Original Equipment Manufacturer |
| OHSC | Overhead Storage Compartment |
| OMG | Object Management Group |
| OOP | Objektorientierte Programmierung |
| OOSEM | Object-Oriented Systems Engineering |
| OSLC | Open Services for Lifecycle Collaboration |
| PAKo | Parametrisch-assoziative Konstruktion |
| PDM | Produkt Daten Management |
| pkg | Package diagram |
| PLE | Product Line Engineering |
| PLM | Product Lifecycle Management |
| PS | Prescriptive Study |
| PSU | Passenger Service Unit |
| RC | Research Clarification |
| RDF | Resource Description Framework |
| req | Requirement Diagram |
| RH | Right Hand |
| SAE | Society of Automotive Engineers |
| SE | Systems Engineering |
| SOA | Service-Oriented Architecture |
| SOI | System of Interest |
| STEP | Standard for the Exchange of Product Model Data |
| STL | Standard Triangulation Language |
| SysML | Systems Modeling Language |
| SYSMOD | Systems Modeling Toolbox |
| SysPhS | SysML Extension for Physical Interaction and Signal Flow Simulation |
| TCP | Transmission Control Protocol |
| TF | Teilforschungsfrage |

| **Abkürzung** | **Bezeichnung** |
|---|---|
| uc | Use Case Diagram |
| UML | Unified Modeling Language |
| UDP | User Data Protocol |
| URI | Uniform Resource Identifier |
| VAMOS | Variant Modeling Method for SysML |
| VDI | Verein Deutscher Ingenieure e.V. |
| VR | Virtual Reality |
| VSP | Vehicle Sketch Pad |
| W3C | World Wide Web Consortium |
| WGP | Wissenschaftliche Gesellschaft für Produktionstechnik |
| XMI | XML Metadata Interchange |
| XML | eXtensible Markup Language |

# Definitionen

| | |
|---|---|
| Adapter | Ein Adapter ist ein Hilfsmittel, das als Austauschformat für Daten und als Mittler zwischen Disziplinen oder Modellierungsumgebungen dient. |
| Customizing | Customizing bezeichnet die individuelle Anpassung eines Angebots (Serienprodukt, Dienstleistung, Software) an die Bedürfnisse und Wünsche eines Kunden. |
| Digitales Produktmodell | „Ein Produktmodell ist die formale Beschreibung aller Informationen zu einem Produkt über alle Phasen des Lebenszyklus hinweg." [ISO24] |
| Interoperabilität | Interoperabilität im MBSE ist die Fähigkeit verschiedener Modellierungswerkzeuge und -umgebungen, Modelle und Daten ohne Informationsverlust auszutauschen und korrekt zu interpretieren, was eine kontinuierliche Systementwicklung über Disziplinen und Lebenszyklusphasen hinweg ermöglicht. |
| Komplexität | „Komplexität [ist] eine Systemeigenschaft, deren Grad von der Anzahl der Systemelemente, von der Vielzahl der Beziehungen zwischen diesen Elementen sowie der Anzahl möglicher Systemzustände abhängt." [Sch05] |
| MBSE | „Model-based Systems Engineering (MBSE) ist die formalisierte Anwendung der Modellierung zur Unterstützung der Systemanforderungen, des Entwurfs, der Analyse, der Verifizierung und der Validierung von Aktivitäten, die in der Konzeptionsphase beginnen und sich über die gesamte Entwicklung und spätere Lebenszyklusphasen erstrecken." [INC07, S. 15] |
| Modell | Ein Modell ist die Nachbildung (Darstellung oder Wiedergabe) eines geplanten oder existierenden physikalischen Systems, bei dem die wesentlichen Eigenschaften hervorgehoben werden. Ein Modell stellt damit ein vereinfachtes Abbild der Wirklichkeit dar. [VDI10] |
| Ontologie | „Eine Ontologie ist eine formale, explizite Konzeptualisierung einer Problemdomäne, die von allen Beteiligten geteilt wird; sie stellt ein kontrolliertes Vokabular dar, das eine Reihe von vereinbarten Begriffen (semantische Domäne) und Regeln für die Verwendung und Interpretation dieser Begriffe innerhalb der Domäne umfasst." [MS18] |

| | |
|---|---|
| Partialmodell | Ein Partialmodell beschreibt und legt ein Subsystem aus. Es kann entweder aus einem einzigen Modell oder aus mehreren heterogenen Modellen bestehen, die miteinander gekoppelt sind. Zudem hat es fest definierte Ein- und Ausgänge für Parameter. |
| Technisches System | Ein System ist eine Ansammlung aus in Beziehung stehender Teile, die zusammen ein Ganzes bilden. [HWFV12] |
| Variante | Varianten sind unterschiedliche Ausführungen eines Produkts. Varianten sind Optionen, die in bestimmten Produktmerkmalen wie z.B. Form, Funktion oder Material voneinander abweichen. |
| Version | Unter einer Produktversion wird eine einzelne abgeschlossene Art beziehungsweise Form eines Produkts bezeichnet. Man unterscheidet dabei Produktversionen von vorläufigen Zwischenprodukten sowie verschiedenen Endprodukten [Pro22]. |
| Virtuelles Produkt | „Der Begriff Virtuelles Produkt fasst mehrere physikalische Eigenschaften eines Produktes zusammen und vereinigt sie interoperabel in einem Produktmodell.“ [And06] |
| Werkzeug | Werkzeuge repräsentieren technische Arbeitsmittel (Software und Hardware), die bei der Erfüllung der Engineering-Aufgabe zum Einsatz kommen [Hil21]. (Software-)Werkzeuge unterstützen die Mitarbeiter durch CAD-Software, Simulationsmodelle oder abstrakte Modelle bei der Planung, Analyse, Simulation und Konstruktion. Nachfolgend wird in dieser Arbeit die englische Bezeichnung Tool verwendet. |

# 1. Einleitung

## 1.1 Hintergrund und Motivation

Die Zukunft des Engineerings liegt in der digitalen Entwicklung von Produkten. Dabei zeigt sich ein Trend beginnend vom digitalen Produkt in Richtung virtuelles Produkt hin zum virtuellen Testen. Dafür werden simulationsfähige digitale Produktmodelle benötigt, die alle Eigenschaften und Produktinteraktionen abbilden [LS21]. Gleichzeitig steigt nicht nur die Systemkomplexität, sondern auch die Komplexität in der Produktentwicklung durch steigende Produktvielfalt aufgrund einer zunehmenden Individualisierung der Produkte [Har09]. Um Produkte effizient zu entwickeln, Anforderungen zu testen und holistisch im Gesamtsystem zu untersuchen, werden heterogene Fachdisziplinen benötigt. Zusätzlich müssen zahlreiche, unterschiedliche Experten und deren Ergebnisse ins Gesamtsystem integriert werden [GDS+13]. Lösungsansätze und Methoden zur Bewältigung dieser komplexen Aufgabe bietet das modellbasierte Systems Engineering. Dieses stellt Modelle bereit, die eine Analyse, Simulation und Vorhersage dieser Systeme ermöglichen. Die Herausforderungen, die dieser Wandel mit sich bringt, sind unter anderem eine wachsende Interdisziplinarität und eine hohe Schnittstellenvielfalt (Abbildung 1.1). Eine wichtige Rolle spielt dabei die Interoperabilität zwischen den heterogenen Modellen. Eine durchgängige Vernetzung zwischen den Teilmodellen der verschiedenen Fachdisziplinen ist essentiell für die Beherrschung der Komplexität dieser multidisziplinären Systeme.

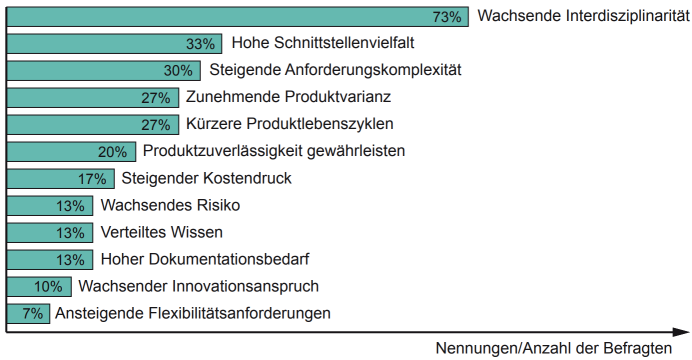



Abbildung 1.1: Umfrageergebnisse zu Herausforderungen der zukünftigen Produktentwicklung (Mehrfachnennung möglich) von [GDS+13].

Weitere Studien zum industriellen Einsatz von modellbasiertem Systems Engineering, wie etwa von Kößler [KP17], Amorim et al. [AVP+19], Albouq et al. [ASA+22], Feichtinger et al. [FMR+22], Wilke et al. [WGB+23], Berschik et al. [BSLK23] sowie die INCOSE Systems Engineering Vision 2035 [INC21], verdeutlichen, dass die genannten zentralen Herausforderungen aus der Studie von Gausemeier et al. in 2013 [GDS+13] weiterhin bestehen (Abbildung 1.2). Insbesondere Interoperabilität, Variabilität, Multidisziplinarität und die Bewältigung wachsender Komplexität bleiben ungelöst. Aus Sicht der Industrie fehlen bislang praxistaugliche Lösungsansätze, die eine durchgängige Integration unterschiedlicher Disziplinen, ein effizientes Variantenmanagement sowie eine Handhabung komplexer Systeme und Modelle ermöglichen.


Abbildung 1.2: Identifizierte, zentrale Herausforderungen im modellbasierten Systems Engineering im industriellen Kontext, basierend auf den Studien [KP17, AVP+19, ASA+22, FMR+22, WGB+23, BSLK23, INC21].

Auch in der Luftfahrtforschung werden die durchgängige Digitalisierung und Vernetzung der spezifischen Werkzeuge (im Weiteren als Tool bezeichnet) als wesentliche Ziele angestrebt. Mit Hilfe der Digitalisierung sollen ungenutzte Potenziale identifiziert und der gesamte Entwicklungsprozess neuer Flugzeuge sowie der Zulassungs- und Zertifizierungsprozess beschleunigt werden [RFG+23]. In der Luftfahrtbranche finden die modellbasierten Ansätze und Methoden aus dem Systems Engineering für die Entwicklung multidisziplinärer Systeme vermehrt Anwendung [MAK+24]. Das zu entwickelnde Produkt wird dabei durch digitale Modelle dargestellt und analysiert. Je nach Fachdisziplin sowie in den unterschiedlichen Entwicklungsphasen im Prozess werden andere Fidelitätsebenen und Detaillierungsgrade benötigt. Jede Disziplin stellt dabei ihre speziellen Modellierungsumgebungen bereit, die nicht mit denen anderer Disziplinen kompatibel sind. Beispielsweise werden Modelle für interdisziplinäre Simulationen für die Überprüfung von Sicherheitsanforderungen oder die dreidimensionale geometrische Darstellung benötigt. Abbildung 1.3 veranschaulicht verschiedene Fidelitätsgrade der spezifischen Entwicklungsaufgaben während des Entwicklungsprozesses.

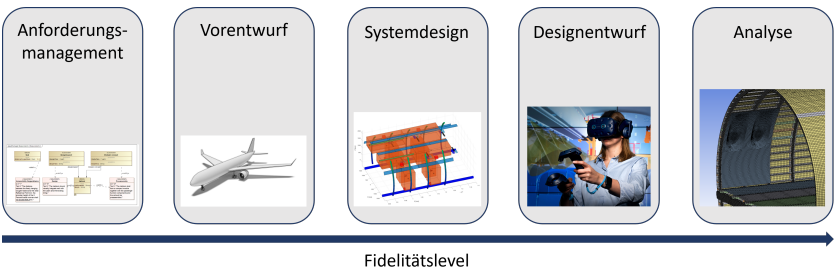



Abbildung 1.3: Verschiedene Fidelitätsgrade von Modellen während des konzeptuellen Entwicklungsprozesses von Flugzeugen von [FGBN23].

Trotz der Vielfalt an Tools und Modellen gibt es auf der Systemebene die Notwendigkeit, diverse Produktkonfigurationen in durchgängigen Modellen zu beschreiben und zu analysieren. Die Abbildung von Varianten und die Integration neuer Technologien in bestehende Architekturen ist wichtig. Besonders in der Flugzeugkabine ist der Konfigurierungsgrad sehr hoch, da unterschiedliche Wünsche des Kunden zu einer neuen Architektur bzw. Anordnung der Komponenten innerhalb des Systems führen. Gleichzeitig besteht die Kabine aus vielen Subsystemen, die wiederum miteinander wechselwirken. Die funktionalen Abhängigkeiten führen zu vernetzten physikalischen Komponenten und damit zu komplexen Systemen. Steigt die Anzahl der zu vernetzenden Modelle, erschwert dies die Modellierung von Varianten. Die Kopplung von Schnittstellen muss einerseits auf Systemarchitekturebene zwischen den Subsystemen stattfinden und andererseits auf Toolebene. Nur die Kombination aller Modelle miteinander ermöglicht eine effiziente Auslegung des Produktes und Überprüfung aller Anforderungen. Zusätzlich können neue Technologiebausteine oder Varianten integriert werden, um zu untersuchen sowie zu bewerten, welche Auswirkungen diese auf die bestehende Systemarchitektur haben. Dadurch können Unternehmen schneller und aussagekräftiger neue Luftfahrtprodukte bewerten, Kosten reduzieren und die Qualität steigern. Allerdings fehlen derzeit Ansätze für die Vernetzung von Prozessen und Methoden, um eine durchgängige Nutzung der Daten in allen disziplinspezifischen Modellen zu ermöglichen.

## 1.2 Zielsetzung und Abgrenzung der Arbeit

In dieser Arbeit wird die digitale Durchgängigkeit und die Vernetzung von Modellen für eine effiziente Auslegung von Produktvarianten betrachtet. Nur durch einen frühen Markteintritt neuer Technologien und Produkte ist es Unternehmen möglich, den neuen und sich stetig ändernden Anforderungen der Kunden, den Industriestandards und den staatlichen Richtlinien zu genügen [Bra23].

> „Die dynamische technologische Entwicklung und der globale Wettbewerb führen zu immer kürzeren Produktlebenszyklen. Dies bedeutet aber auch kürzere Entwicklungszeiten trotz steigender Anforderungen an das Produkt. Das Spannungsfeld zwischen Qualität, Kosten und Zeit determiniert den gesamten Produktentstehungsprozess und kann nur durch eine systematischere sowie effizientere Vorgehensweise gelöst werden.“ [GDS+13, S. 17]

Wie bereits Gausemeier et al. aufzeigen, steigen die Anforderungen an ein Produkt durch den vermehrten Bedarf an intelligenten, vernetzten Systemen [GDS+13]. Bei der Produktentwicklung einer Flugzeugkabine wird dies durch das Customizing zusätzlich verstärkt. Gründe hierfür sind einerseits der hohe Individualisierungsbedarf der Airlines als Kunde, anderseits der kurze Lebenszyklus einer Kabine, da die Innenausstattung in der Kabine nach sieben bis acht Jahren aus- oder umgerüstet wird [LHK21]. Die digitalen Modelle für den Entwurf von Kabinenvarianten müssen daher flexibel und erweiterbar sein, damit Rekonfigurationen oder neue Kabinenkonzepte untersucht werden können. Gleichzeitig müssen die fachspezifischen Modelle für eine ganzheitliche Betrachtung miteinander gekoppelt werden.

Der Fokus dieser Arbeit liegt auf der Modellierung von Systemen in deskriptiven Modellen sowie auf der Interoperabilität zwischen deskriptiven und analytischen Modellen. In der Luftfahrtindustrie wird für die Architekturmodellierung eines Systems vermehrt die deskriptive Modellierungssprache SysML eingesetzt. Das Gesamtsystem wird zudem in einem einzigen SysML-Modell modelliert und daraus an die Analyseumgebungen angeschlossen. Dies hat zur Folge, dass bei der Modellierung von Produktvarianten riesige Modelle entstehen, die aufgrund der Größe und internen Komplexität bei Änderungen oder bei der Integration neuer Variationen unflexibel sind. Diese Herausforderungen führen zu einem zusätzlichen Zeit- und Arbeitsaufwand in der Produktentwicklung.

Ziel dieser Arbeit ist die Vorstellung einer methodischen Unterstützung, um in der praktischen Anwendung die Modellierung und Auslegung von Produktvarianten in der frühen Phase der Produktentwicklung zu verbessern. Der entwickelte Ansatz zielt auf eine Beschleunigung in der Produktentwicklung ab, indem Produktvarianten modular und flexibel ausgelegt werden können. Dabei werden von Anfang an alle Fachdisziplinen bei der Produktentwicklung eingebunden und der Entscheidungsprozess zur Auswahl der bestmöglichen Lösung verbessert. Gleichzeitig wird das Customizing unterstützt, indem schneller und zuverlässiger auf spezifische Wünsche und Anforderungen des Kunden eingegangen werden kann. Dadurch kann der Kunde neue Features in einem frühen Stadium visualisiert bekommen, bevor ein physischer Prototyp entwickelt wird [Bra23]. Zusammenhänge können aufgezeigt und die Daten als Entscheidungshilfe aufbereitet werden, um durch frühzeitiges Wissen bessere Entscheidungen im Entwicklungsprozess treffen zu können.

Das Hauptziel dieser Arbeit ist die Reduzierung der induzierten Komplexität der Modelle bei gleichzeitig bereitgestellter Modularität durch einen erweiterten methodischen Ansatz. Die folgende Forschungsfrage liegt der Arbeit zugrunde:

> *Wie kann methodisch eine Modularität des Gesamtsystems ermöglicht werden, damit Varianten eines komplexen technischen Systems abgebildet und für die multidisziplinäre Analyse des Systemverhaltens kollaborativ modelliert werden können?*

Ausgehend von der Forschungsfrage zur kollaborativen Entwicklung von Varianten ergibt sich als zentrales Subziel, eine Methodik zu entwickeln, die im industriellen Kontext praktisch anwendbar ist. Die praktische Einsetzbarkeit stellt dabei eine wesentliche Anforderung dar, da eine rein theoretische Methodik den Anforderungen der industriellen Realität nicht gerecht würde. Darüber hinaus muss die Methodik ein angemessenes Aufwand-Nutzen-Verhältnis aufweisen, um in der Praxis akzeptiert und umgesetzt zu werden. Die Berücksichtigung der häufig hohen Komplexität variantenreicher Produktmodelle erfordert zudem eine Lösung, die alternative Wege zur Komplexitätsreduktion ermöglicht. Damit leitet sich aus dem Kontext der Forschungsaufgabe die Notwendigkeit ab, eine praxistaugliche, anwendungsorientierte Methodik zu schaffen, die sowohl den kollaborativen Entwicklungsprozess unterstützt als auch auf die realen Herausforderungen industrieller Unternehmen eingeht. Daraus folgt die ergänzende Forschungsfrage dieser Arbeit:

> *Wie kann eine praxisorientierte Methodik zur kollaborativen Entwicklung von Varianten gestaltet werden, die im industriellen Kontext anwendbar ist*

*und Komplexität sowie Aufwand-Nutzen-Verhältnis angemessen berücksichtigt?*

## 1.3 Aufbau der Arbeit

Im Folgenden wird ein Einblick in die Vorgehensweise dieser Arbeit und deren Aufbau gegeben. Als zugrundeliegende Methodik wird das methodische Vorgehen der Design Research Methodology (DRM) von Blessing und Chakrabarti verwendet [BC09]. Die DRM wurde mit der Absicht entwickelt, eine standardisierte Versuchsmethodik bereitzustellen, die einen Überblick über die Designforschung schafft, die Bewertung der Forschungsleistung verbessert und den Transfer der Forschungsergebnisse in die Praxis erleichtert [BC09]. Dabei kann die erforschte Unterstützung eine Methode, eine Richtlinie oder ein Tool sein [BC09]. In dieser Arbeit wird eine Methode als Unterstützung zur Verbesserung der Variantenmodellierung entwickelt. Daher werden nachfolgend in dieser Arbeit die Begriffe *entwickelte Methode* und *entwickelte Unterstützung* als Synonym verwendet.

> „The overall aim of engineering design research is to support industry by improving our understanding of engineering design and, based on this, developing knowledge, in the form of guidelines, methods and tools that can improve the chances of producing a successful product [...]. “ [BC02, S. 2]

Die DRM unterteilt sich in vier Phasen: Research Clarification (RC), Descriptive Study I (DS I), Prescriptive Study (PS) und Descriptive Study II (DS II). Die Strukturierung der Arbeit erfolgt nach diesen vier Phasen, die den Rahmen für die wissenschaftliche Arbeit darstellen und deren englische Bezeichnungen verwendet werden. Abbildung 1.4 veranschaulicht den Aufbau der Arbeit und die Zuordnung der Kapitel zu den Phasen. Die gewählte Vorgehensweise der DRM bietet sich an, um die Forschung für den Entwurf eines methodischen Ansatzes für die Variantenmodellierung zu unterstützen und wurde daher als zugrundeliegende Forschungsmethodik für diese Arbeit gewählt.

Research Clarification
In der ersten Phase werden die Ausgangssituation und die Zielsetzung der Arbeit vorgestellt sowie die zugrundeliegende Forschungsfrage definiert (Kapitel 1). Zusätzlich werden in Kapitel 2 die Problemstellung hergeleitet und die allgemeinen Grundlagen für die Bestimmung des Forschungsziels erarbeitet.

Descriptive Study I
Das Ziel der zweiten Phase ist die Erarbeitung eines besseren Verständnis der Problemstellung und die Identifikation von Schlüsselfaktoren für eine Verbesserung durch die zu entwickelnde Unterstützung [BC09]. Damit bietet sie ebenfalls die Grundlage für die Entwicklung dieser Unterstützung. Zusätzlich werden Anforderungen erarbeitet, anhand derer die entwickelte Unterstützung abschließend bewertet wird. Innerhalb der DS I erfolgt daher eine Analyse des Stands der Technik und der Forschung (Kapitel 3). Der Fokus liegt dabei zum einen auf der deskriptiven Studie zur digitalen Entwicklung von Flugzeugkabinensystemen und der Vorstellung des industriellen

Forschungskontextes. Zusätzlich wird vorgestellt, welche Modelltypen und Tools im Flugzeugentwurf zur Anwendung kommen (Kapitelabschnitt 3.1). Zum anderen wird der allgemeine Einsatz von modellbasierten Entwicklungsansätzen im Systementwurf und begriffliche Grundlagen in einer erklärenden Studie vorgestellt (Kapitelabschnitt 3.2). Aus beiden Studien leiten sich aktuelle Herausforderungen und die Forschungslücke bei der Modellierung von Produkten im Hinblick auf Varianten ab. Anschließend wird eine Analyse sowohl bestehender Methoden für die Variantenbildung (Kapitel 4) als auch bestehende Ansätze für die Kopplung von (multidisziplinären) Modellen für eine kollaborative Entwicklung (Kapitel 5) durchgeführt. Dabei werden die bereits vorhandenen Lösungsvorschläge aus der Literatur zusammengetragen und Herausforderungen identifiziert.

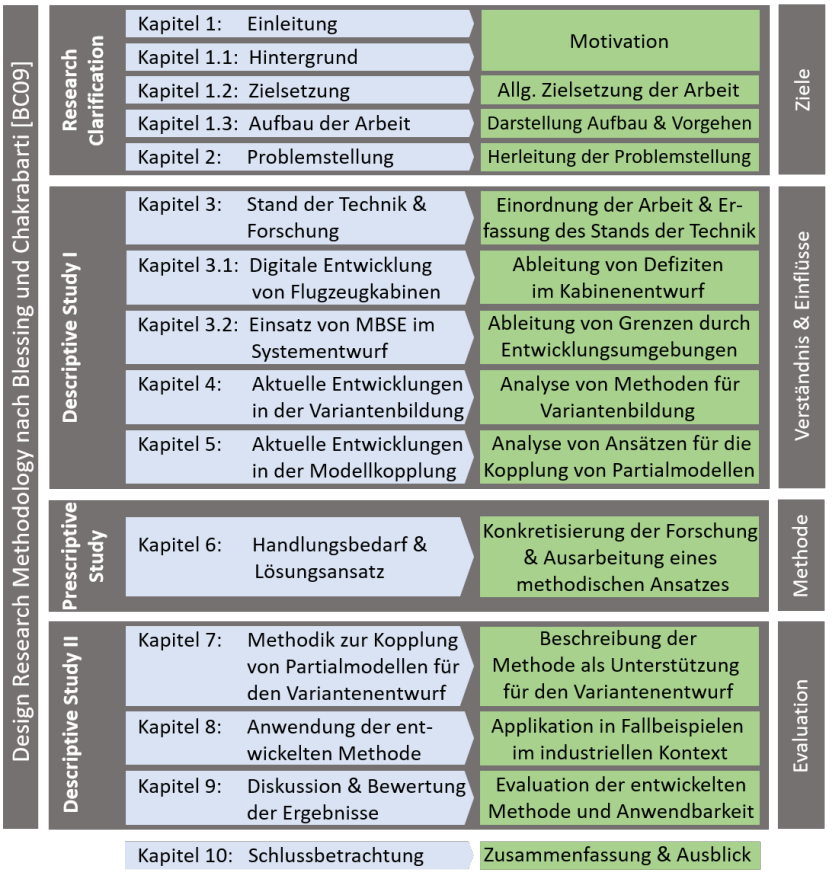



Abbildung 1.4: Struktureller Aufbau dieser Arbeit und Zuordnung zur DRM.

Prescriptive Study

Die dritte Phase beinhaltet die Ausarbeitung und Beschreibung einer Unterstützung

zur Lösung der erarbeiteten Problemstellung. Kapitel 6 konkretisiert den Forschungsbedarf aufbauend auf den Erkenntnissen der vorangegangenen Phase, indem die Herausforderungen und Grenzen aus der Literaturrecherche der vorherigen Kapitel zusammengefasst und der Handlungsbedarf abgeleitet werden. In Kapitelabschnitt 6.2 wird der entwickelte Lösungsansatz beschrieben und vorgestellt, der wiederum als Unterstützung in dieser Arbeit zur Beantwortung der Forschungsfrage und zur Erfüllung der in Kapitelabschnitt 1.2 vorgestellten Zielsetzung erarbeitet wurde.

Descriptive Study II

In der vierten Phase wird die Anwendung der entwickelten Unterstützung im industriellen Kontext in einem realistischen Anwendungsszenario erprobt und anschließend bewertet, inwieweit die erarbeiteten Anforderungen erfüllt werden. Die praktische Umsetzung der entwickelten Methode wird am Beispiel der Flugzeugkabine in Kapitel 7 erläutert. Die durchgeführten industriellen Fallstudien und deren Ergebnisse sind in Kapitel 8 beschrieben. Die Evaluation der Ergebnisse erfolgt in Kapitel 9. In diesem werden die Ergebnisse der zwei Studien im industriellen Kontext ausgewertet und diskutiert. Zudem werden in diesem Kapitel die Ergebnisse mit der Forschungsfrage reflektiert. In Kapitel 10 wird die Arbeit zusammengefasst und die wichtigsten Erkenntnisse resümiert. Abschließend wird ein Ausblick auf weiterführende Forschungsarbeiten basierend auf der hier vorgestellten Unterstützung gegeben.

# 2. Herleitung der Problemstellung

In diesem Kapitel wird der Forschungsschwerpunkt der Arbeit erläutert und ist damit ein Teil der Research Clarification der DRM. Dazu werden die aktuellen Herausforderungen in der Produktentwicklung und das zugrundeliegende Problemverständnis der Forschungsarbeit vorgestellt. Im Anschluss folgen Ansätze, wie diese Herausforderungen gelöst werden können. In einem Zwischenfazit wird der daraus resultierende konkretisierte Handlungsbedarf herausgestellt.

## 2.1 Herausforderungen in der Produktentwicklung

Für die Auslegung eines Systems stellt das Systems Engineering modellbasierte Methoden und Ansätze bereit. Bei der Anwendung dieser modellbasierten Methoden für die Bildung von Produktvarianten ergeben sich zwei Herausforderungen:

**1. Vorliegende breite, heterogene Toollandschaft mit einer fehlenden Interoperabilität**
Eine disziplinübergreifende Zusammenarbeit verbessert das Systemverständnis, identifiziert Fehler frühzeitig und ermöglicht die Gestaltung innovativer Kundenlösungen [DAG+21]. Bei der Auslegung eines Systems stellt jede Fachdisziplin eigene Modelle bereit, um einen bestimmten Systemaspekt zu modellieren. Denn die im Produktentwicklungsprozess aufgetretenen Problemschwerpunkte erfordern unterschiedliche Hilfsmittel oder Methoden zur Bearbeitung [SK97]. Dabei zeigen die Modelle nur die für die Vorhersage der Fachdisziplin notwendige Abstraktion der Realität, wodurch diese in ihrer Detailtiefe variieren. Wesentlicher Vorteil dieser breiten Informationsverfügbarkeit ist die Verbesserung der Produktqualität [DAG+21]. Hieraus ergibt sich allerdings eine breite Toollandschaft. Zusätzlich entstehen durch die Vielzahl an Softwarelösungen und Modellen ohne eine geeignete Kopplung der Schnittstellen Datensilos. Die Notwendigkeit einer Verknüpfung verschiedener Disziplinen und deren Modellen für die umfassende Betrachtung eines Systems und der Überprüfung aller Anforderungen ist dabei bereits bekannt [BBH+21]. Gausemeier et al. betonen in ihrer Übersicht zum Einsatz von Systems Engineering im industriellen Kontext die Wichtigkeit einer durchgängigen Werkzeugkette [GDS+13].

> „Zur Beherrschung der Komplexität multidisziplinärer Erzeugnisse ist eine durchgängige Virtualisierung des Produktentstehungsprozesses unabdingbar. Hier sind Konsistenzbeziehungen zwischen den Teilmodellen

> der Produkt- und Produktionssystementwicklung (insbesondere mit unterschiedlichem Detaillierungsgrad und verschiedener Fachdisziplinen) herzustellen und so die spezifischen Werkzeuge zu kohärenten Werkzeugketten zu kombinieren.“ [GDS+13, S. 18]

Schumacher und Inkermann stellen dennoch heraus, dass die Verknüpfung von Daten aus disziplinspezifischen (z.B. Analyse- oder Geometriemodelle) und disziplinübergreifenden Modellen (z.B. UML/SysML-Modelle) oft nicht ausreichend betrachtet wird [SI21]. Die fehlende Interoperabilität der Modelle hat jeweils eine Einschränkung der Datendurchgängigkeit und der Datenkonsistenz zur Folge [SI21]. Darüber hinaus werden in der Industrie überwiegend kommerzielle Softwarelösungen eingesetzt, die im Vergleich zu open-source Lösungen in ihrer Anbindung an weitere Toolumgebungen limitiert sind. Daher werden Schnittstellentechnologien gesucht, die eine Kopplung und damit einen Austausch von Parametern zwischen den verschiedenen Entwicklungsumgebungen ermöglichen.

**2. Eingeschränkte Flexibilität durch hohe induzierte Komplexität in den Systemmodellen**

Eine weitere Herausforderung liegt in der Abbildung der Varianten eines Produkts. Wie bereits Du et al. in ihrer Arbeit herausstellen, bieten Produktanpassungen (Customizing) die Möglichkeit, die individuellen Bedürfnisse des Kunden besser zu erfüllen [DJT06]. Allerdings funktioniert dies nur dann, solange die Anforderungen bereits zu Beginn der Produktentwicklung vorliegen. Häufig entwickeln sich zusätzliche Anforderungen an das Produkt erst im weiteren Verlauf der Entwicklung.

> „In most cases, customer requirements are impossible to be completely satisfied by a base product.“ [DJT06, S. 400]

Neben der Möglichkeit zur Generierung von Varianten im Customizing müssen ebenfalls durch Technologie- und Marktentwicklungen getriebene Varianten untersucht werden können, die durch die Integration neuer Systemgruppen (z.B. Brennstoffzelle) oder Technologien (Flüssigwasserstoff) in bereits bestehende Systemarchitekturen entstehen [SBA+21]. Die dabei auftretende Vielzahl an möglichen Produktkonfigurationen muss abgebildet werden können, damit frühzeitig die bestmögliche Lösung vertieft weiterverfolgt werden kann. Für die Beherrschung der hohen Variantenvielfalt und die Möglichkeit auf unterschiedliche Kundenanforderungen zu reagieren, findet unter anderem das Baukastenprinzip Anwendung [SSK19]. Mit Hilfe der Modularisierung wird dabei die Produktstruktur in Komponenten zerlegt und nach technisch-funktionalen oder produktstrategischen Aspekten neu zu Modulen gruppiert [KG18]. Durch den Zusammenschluss verschiedener Module und Subsysteme werden dann technische Systeme mit unterschiedlichen Funktionalitäten entsprechend der Kundenwünsche konfiguriert [Bur16]. Zwei zentrale Herausforderungen dabei sind die Gewährleistung der Konsistenz und die Wahrung der Übersichtlichkeit [SSK19].

> „Even though mass customization is a very attractive strategy with many advantages, it is not easy to implement, and many issues should be considered while planning for such a strategy. One of these issues is the balance between variety and induced complexity.“ [DDBL11, S. 173]

Daaboul et al. sehen die Herausforderung bei der Massenkonfiguration darin, die Balance zwischen Varietät und induzierter Komplexität zu finden [DDBL11]. Die frühzeitige Betrachtung aller potentiell auftretender Konfigurationen erhöht nicht nur den Rechenaufwand, sondern auch die Komplexität der Modelle, die für die Abbildung der Systeme benötigt werden. Langfristig führt dies zu einer erhöhten Entwicklungszeit. Begründet ist dies durch die zunehmende interne Komplexität der Modelle, wodurch sowohl die Anzahl an Änderungen im Systemmodell durch die Anpassung vorhandener Varianten als auch der Entwicklungsaufwand durch Erweiterungen bei der Modellierung neuer Varianten steigt [Roh20, MHS+21]. Myrodia et al. bezeichnen die Komplexität, die durch die Anzahl an Endproduktvarianten entsteht als induzierte Komplexität [MHS+21]. Der Begriff induzierte Komplexität wird in dieser Arbeit weiter verwendet.

Auf Basis der vorgestellten Herausforderungen wurde ein Reference Model nach Blessing und Chakrabarti [BC09] angefertigt. Das Reference Model wird im Rahmen der DRM für die Darstellung der bestehenden Situation verwendet und illustriert die ermittelten Zusammenhänge [BC09]. Abbildung 2.1 zeigt das Reference Model dieser Forschungsarbeit im Kontext der Variantenmodellierung im konzeptionellen Entwicklungsprozess mit den zugehörigen Quellenangaben, aus denen die kausalen Zusammenhänge abgeleitet wurden. Zudem sind die beiden Herausforderungen als Einordnung zu den ermittelten Faktoren dargestellt. Es veranschaulicht die zuvor aufgeführten Zusammenhänge.

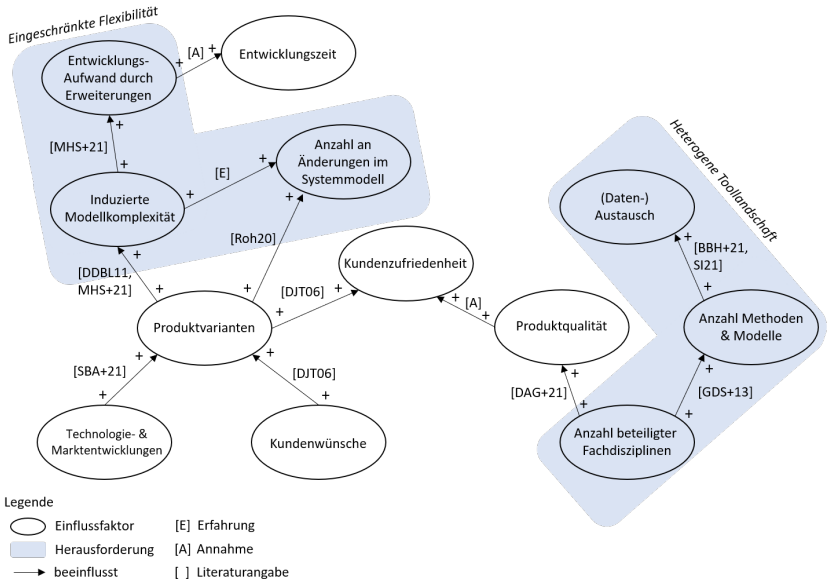



Abbildung 2.1: Reference Model für den derzeitigen Entwicklungsprozess von Varianten in Anlehnung an die DRM von Blessing und Chakrabarti.

## 2.2 Problemverständnis

In der vorliegenden Arbeit wird eine Unterstützung entwickelt, um die genannten Herausforderungen zu adressieren und in der praktischen Anwendung der Variantenmodellierung Abhilfe zu schaffen. Dafür müssen zuerst zwei grundlegende Aspekte der Systemmodellierung näher erläutert werden. Diese sind die Theorie zum Aufbau eines technischen Systems und die Philosophien zur Verlinkung von Modellen. Im folgenden Abschnitt wird ein näherer Einblick gegeben.

#### Technisches System und Allgemeine Systemtheorie

Der Begriff technisches System findet seinen Ursprung in der Definition von Hubka [Hub84] und basiert auf der von Bertalanffy erarbeiteten Allgemeinen Systemtheorie [vB72]. Die Systemtheorie betrachtet zusammengesetzte Gebilde [vB72]. Unter dem Begriff System wird eine Menge an Elementen verstanden, zwischen denen Wechselbeziehungen bestehen [vB72, Hub84]. Dabei kann ein technisches System eine Anlage, ein Apparat, eine Maschine, ein Gerät, eine Baugruppe, ein Maschinenelement oder ein Einzelteil sein, welches eine technische Aufgabe erfüllt [FG21]. Das System ist durch eine Systemgrenze nach außen zu seiner Umgebung oder zu anderen Systemen hin abgegrenzt. Über die Eingangsgrößen $E_1$ (Input) und Ausgangsgrößen $A_1$ (Output) als Schnittstelle steht das System mit der Umgebung in Beziehung [FG21]. Grundsätzlich können drei Eingangs- und Ausgangsgrößen unterschieden werden. Dies sind die Energie (z.B. mechanisch, elektrisch), der Stoff (z.B. Gas, Flüssigkeit) und das Signal (z.B. Messgröße, Informationen) [FG21]. Zusätzlich besitzt jedes System Eigenschaften. Abbildung 2.2 veranschaulicht die allgemeine Prinzipdarstellung eines technischen Systems. Im Folgenden dieser Arbeit wird nur noch der Begriff System verwendet.

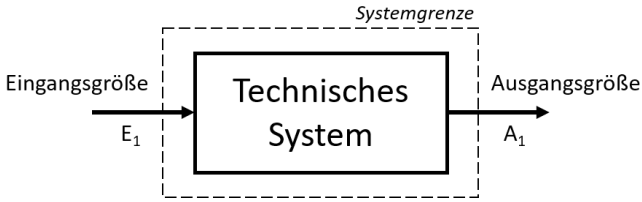



Abbildung 2.2: Allgemeine Prinzipdarstellung eines technischen Systems.

Ein System dient einem bestimmten Zweck (Funktion) und erfüllt eine Gesamtaufgabe bzw. Gesamtfunktion [HWFV12, FG21, BFK+18]. Hierfür werden die Eingaben verarbeitet und Ausgaben erzeugt [BFK+18]. Ein System kann dabei aus beliebig vielen Teilsystemen bestehen [HWFV12]. Bei komplexen technischen Systemen wie dem Flugzeug oder dem Auto wirken verschiedene Teilsysteme zusammen, um eine Gesamtfunktion zu erfüllen. Diese Gesamtfunktion wiederum kann in Teilfunktionen aufgegliedert werden [FG21]. Abbildung 2.3 veranschaulicht eine beispielhafte Darstellung einer Funktionsstruktur abgeleitet aus einer Gesamtfunktion.

Bei der Modellierung dieser Systeme erfolgt eine schrittweise Einengung des Betrachtungsfeldes nach dem Vorgehen „*vom Groben zum Detail*“ [HWFV12]. Zu Beginn der Entwicklung eines Systems erfolgt eine Übersetzung der Anforderungen in Funktionen [VDI21]. Anschließend wird eine Funktionsstruktur (funktionale Beschreibung) mit Teilfunktionen aufgebaut, die wiederum in eine Produktstruktur (physische Beschreibung) transformiert werden kann [FG21]. Ausgehend von der Gesamtfunktion werden je nach gewünschter Detaillierungstiefe Teilfunktionen bzw. Teilsysteme aufgebrochen [BFK$^+$18]. Die Teilsysteme besitzen wieder eigene Eingangs- und Ausgangsgrößen, erfüllen eine Funktion, haben Eigenschaften und stehen mit den anderen Teilsystemen in Beziehung. Sobald die Ausgangsgröße eines Teilsystems die Eingangsgröße eines anderen Teilsystems bildet, liegt eine Beziehung zwischen diesen vor.

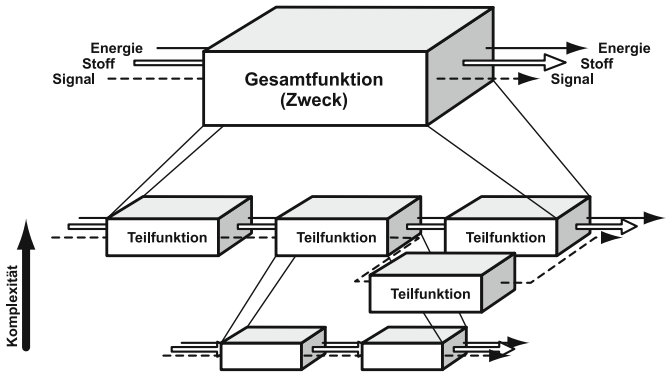



Abbildung 2.3: Beispielhafte Darstellung der Entwicklung einer Funktionsstruktur aus einer Gesamtfunktion von [FG21].

Hierbei kann die Unterteilung in Teilsysteme nach unterschiedlichen Kriterien erfolgen [BFK$^+$18]. Das Zerlegen der Gesamtaufgabe in Teilaufgaben ermöglicht eine parallele oder sequentielle Bearbeitung [VDI21, BFK$^+$18]. Für die Gesamtlösung werden die Teillösungen wieder zusammengefügt. Dies geschieht in der Entwicklung von disziplinübergreifenden Modellen überwiegend in einem einzigen Modell. Besonders bei der Abbildung von Varianten werden in einem Modell eine Vielzahl an alternativer Teilsysteme für die Bereitstellung der gleichen Teilfunktion oder variable Features modelliert, wodurch die induzierte Komplexität steigt. Eine Adaption der Aufteilung auf Modellebene ist daher vielversprechend. Der Ansatz verfolgt eine Aufteilung eines Systems in Teilsysteme, die jeweils in separaten Modellen (Partialmodell) modelliert werden. Die Partialmodelle werden wieder als ein System betrachtet, dass über Ein- und Ausgänge verfügt. Über diese werden die Partialmodelle zu einer Gesamtlösung auf Modellebene zusammengefügt und die Gesamtaufgabe des Gesamtsystems umgesetzt. Diese Vorgehensweise ermöglicht eine schrittweise Reduzierung der induzierten Komplexität und erhöht dadurch die Flexibilität und Modularität in der Modellierung bei Varianten. Die Komplexität und Herausforderungen in der Modellierung, die durch die Abhängigkeiten, die Funktionalitäten oder das Verhalten des (Teil-)Systems selbst entstehen, werden nicht adressiert.

### Philosophien zur Modellverlinkung

Für die Abbildung oder Auslegung eines Systems werden Modelle verschiedener technischer Anwendungsgebiete verwendet [ERZ14]. Disziplinübergreifende Modelle werden als wesentliches Hilfsmittel dazu eingesetzt, komplexe Zusammenhänge besser darzustellen und zu verstehen [HWFV12]. Darüber hinaus werden für die Abbildung und Auslegung von interdisziplinären Systemen spezifische Modelle weiterer Fachdisziplinen benötigt [SI21]. Zusätzlich findet die virtuelle Produktentwicklung in den „initialen Phasen des Produktentstehungsprozesses“ meist in einem Verbund aus Erstausrüster (Original Equipment Manufacturer, OEM), Zulieferern (Supplier) und Partnern statt, wodurch weitere heterogene Partialmodelle unterschiedlicher Toolumgebungen vorliegen [LS21]. Dabei können einige Modelle in Abhängigkeit zueinander stehen und Ausgaben bereitstellen, die als Eingabe für andere Modelle benötigt werden [SI21]. Für eine durchgängige Kopplung dieser Modelle und die Dokumentation der Entwicklungsergebnisse gibt es zwei verschiedene Philosophien (Abbildung 2.4).

Der integrierte Ansatz verfolgt eine Modellierung „innerhalb einer durchgängigen Toolumgebung“ [EGZ12]. Alle Partialmodelle werden in einer gemeinsamen Datenbank verwaltet [GAM+14]. Allerdings sind die Integration neuer Tools oder grundlegende Änderungen an der Datenstruktur mit einem erheblichen Aufwand verbunden [Pow23]. Eine Zusammenführung von inhomogenen Partialmodellen ist beim integrierten Ansatz erschwert und resultiert in einer unzureichenden Berücksichtigung der Individualität des Entwicklungsprozess [Pow23, ABS+19].

Bei dem föderierten Ansatz sind die Modelle nicht direkt miteinander gekoppelt. Stattdessen werden Inhalte zwischen den Modellen nach Schumacher und Inkermann über einen Adapter ausgetauscht [SI21]. Nach Grundel et al. sind die Modelle über eine Föderierungsplattform verbunden. Die Partialmodelle verfügen jeweils über individuelle Schnittstellen und sind darüber mit der Föderierungsplattform verbunden [GAM+14]. Vorteile des föderierten Ansatzes sind der weitere Einsatz bereits existierender Tools sowie die hohe Anbindungsfähigkeit und Flexibilität in einer bereits etablierten heterogenen Toollandschaft [GAM+14, Pow23]. Nachteilig bei diesem Ansatz sind die Versionierung und die hohe Anzahl an vielfältigen Datentypen, die in Einklang gebracht werden müssen [Pow23].

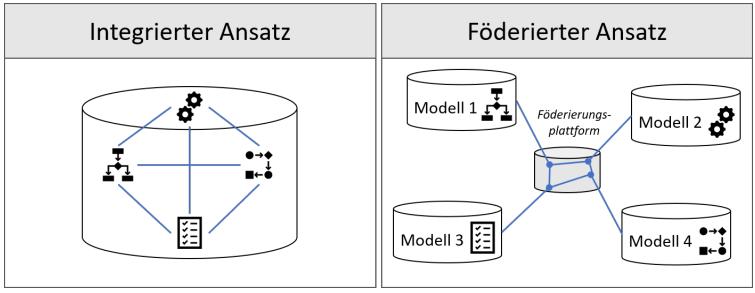



Abbildung 2.4: Ansätze zur Modellverknüpfung, angelehnt an [GAM+14, Pow23].

Der föderierte Ansatz verspricht im Gegensatz zum integrierten Ansatz eine hohe Flexibilität für die Integration neuer Methoden oder Tools und wird in dieser Arbeit für

die Entwicklung einer Unterstützung in der praktischen Anwendung zur Modellierung von Produktvarianten näher betrachtet. Eine große Herausforderung bleibt dennoch die Kopplung der Modelle unter Gewährleistung konsistenter Daten und konsistenter Schnittstellen [Pow23].

## 2.3 Zwischenfazit

Die genannten Herausforderungen sind aus Sicht der Industrie nur teilweise gelöst und bestehende Lösungsansätze wurden nur bei der Kopplung von zwei Fachdisziplinen und an kleineren Systemen getestet. Bei komplexeren Systemen wie dem Flugzeug oder der Flugzeugkabine mit einer Vielzahl an stark vernetzten Subkomponenten stoßen die bestehenden Ansätze an ihre Grenze. Besonders bei dem Einsatz von disziplinübergreifenden Modellen sind neue Ansätze für die Modellierung von Produktvarianten gesucht. Die Überführung von der Theorie in die Praxis stockt. Das Resultat ist, dass die Abbildung aller potentiellen Varianten in einem einzigen Modell die Modellgröße erhöht, die induzierte Komplexität steigert, Modellladezeiten verlangsamt und die Flexibilität bei Anpassungen verringert. Dies führt bei der Integration neuer Systemgruppen oder Technologien im Rahmen der Variantenuntersuchung zu einem erhöhten Entwicklungsaufwand durch viele nachträgliche Anpassungen im Systemmodell und letztendlich zu einer längeren Entwicklungszeit.

# 3. Stand der Technik und Forschung

Im Folgenden werden der Stand der Technik und Forschung inklusive wichtiger Begriffe und grundlegender Zusammenhänge vorgestellt. Ziel des Kapitel ist die Gewährleistung eines gemeinsamen Verständnis und die Einordnung in den industriellen Kontext. Zuerst erfolgt ein Überblick über den industriellen Kontext, indem die methodische Unterstützung für die praktische Anwendung entwickelt wird. Der Fokus liegt auf den Themen virtuelles Produkt, Modellarten sowie Potentiale und Herausforderungen für den Kabinenentwurf (Abschnitt 3.1). Anschließend wird das Themengebiet Systems Engineering mit einem besonderen Fokus auf modellbasiertes Systems Engineering, Modelle und SysML vorgestellt (Abschnitt 3.2). Das Kapitel bildet damit den ersten Teil der Descriptive Study I der DRM.

## 3.1 Digitale Entwicklung von Flugzeugkabinen

> *„[...] plants rely on thousands of suppliers worldwide, who produce roughly 80% of the aircraft, before it enters our premises." [Air23c]*

Eine Herausforderung, die in dieser Arbeit adressiert wird, ist die vorherrschende heterogene Toollandschaft bei der Entwicklung von Systemen. Ohne eine Kopplung der Modelle dieser Tools und ohne einen Datenaustausch der Modelle entstehen zudem sogenannte "Datensilos". Klöpper und Schlake verstehen unter dem Begriff Datensilo die Speicherung und Verarbeitung gesammelter Daten in isolierten Informationssystemen [KS14]. In einem derartigen Zustand ist die Datenanalyse erschwert. Ein weiterer Einfluss für die Entstehung von Datensilos ist die Vielzahl an Experten, die am Entwicklungsprozess beteiligt sind. Systeme wie Flugzeuge und deren Kabinen bestehen zu 80% aus Zukaufteilen. Im Entwicklungsprozess müssen diese unterschiedlichsten Experten miteinander kommunizieren und dabei über Unternehmensgrenzen hinaus Daten konsistent austauschen. Hierbei kann eine Unterbrechung im Wissenstransfer entstehen [RWKGVV19].

Dennoch liegt in der heterogenen Toollandschaft ein großes Potential. Jede Fachdisziplin modelliert einen bestimmten Systemaspekt. Erst durch einen Zusammenschluss dieser Tools können das Gesamtsystem gänzlich verstanden und systemübergreifend Auswirkungen durch Änderungen nachvollzogen werden. Dafür müssen die Modelle digital Daten austauschen und alle am Prozess beteiligten Experten miteinander kommunizieren. Diese digitale Durchgängigkeit wird auch als Digitaler Faden (Digital Thread) bezeichnet [SW18]. Besonders bei der Entwicklung von Flugzeugkabinen, die durch ihre hohe interne Systemkomplexität und die vielen am Prozess beteiligten

Zulieferer geprägt ist, kann die durchgängige Kopplung der Modelle den Entwicklungsprozess beschleunigen und zur Beherrschung der Komplexität beitragen.

Ein Einblick in die Entwicklung von Flugzeugkabinen und deren Systeme wird in dem folgenden Kapitel gegeben. Begonnen wird mit einer Einführung in die virtuelle Produktentwicklung. Es wird das Potential aufgezeigt, das eine durchgängige Kopplung der Modelle bewirken kann und die Nutzung einer virtuellen Produktentwicklung für den Kabinenentwurf motiviert. Darauffolgend wird der Stand der Technik zur Nutzung von Tooltypen im Kabinenentwurf vorgestellt. Abschließend erfolgt eine Zusammenstellung der auftretenden Herausforderungen und Grenzen der heterogenen Toollandschaft, mit Fokus auf die Flugzeugkabine.

### 3.1.1 Einführung in das virtuelle Produkt

In den letzten Jahren haben sich verschiedene Trends im Bereich der Produktentwicklung aufgetan, hervorgerufen durch die steigende Komplexität der Produkte und sich stetig ändernden Kundenbedürfnisse. Einer davon ist die stetige Digitalisierung in der Produktentwicklung. Mit Hilfe der Digitalisierung können komplexere Produkte in kürzerer Zeit realisiert, Informationen entlang des gesamten Produktlebenszyklus gemanagt und Informationen von der Produktion als Feedback in die Entwicklung zurückgesendet werden. Dies führt zunehmend in allen Produktentwicklungsphasen dazu, dass rechnerunterstützt gearbeitet wird. Diese Weiterentwicklung wurde vor allem durch die steigenden Anforderungen und verbesserten Möglichkeiten in der Informationsverarbeitung getrieben. Mit dem Aufkommen des Begriffs Produktmodell Mitte der 80er Jahre wurden nicht nur die geometrischen Sachverhalte, sondern auch weitere Informationen definiert, die für eine integrierte Arbeit notwendig sind [SK97]. Eine Modellart, wie z.B. 3D CAD (Computer-aided Design), kann nicht alle Informationen bereitstellen und speichern. Daraus resultiert die Anforderung nach Übertragbarkeit von Daten aus einem Modelltyp in ein oder mehrere Systemmodelle und somit die Möglichkeit zum Austausch von Produktmodelldaten. Der Begriff Virtuelles Produkt wird in dieser Arbeit nach Anderl definiert:

**Definition 3.1.1 Virtuelles Produkt**

„Der Begriff Virtuelles Produkt fasst mehrere physikalische Eigenschaften eines Produktes zusammen und vereinigt sie interoperabel in einem Produktmodell.“ [And06]

Ein weiterer Aspekt ist die Langzeitarchivierung von Daten. Für die Wartung, Produkthaftung oder Weiterentwicklung, basierend auf bereits entwickelten Produkten, müssen diese Informationen in einer standardisierten und einheitlichen Weise gespeichert werden. Mit der fortschreitenden Möglichkeit der Rechnerverarbeitung wurden physische Versuchsmodelle schrittweise durch Berechnungen und Simulationen ersetzt. Dazu gehören z.B. die rechnerbasierten Methoden zur Produktentwicklung und -konstruktion (CAD), zur Simulation (Finite-Elemente-Methode, FEM) und zur Validierung und Verifikation über digitale Versuchsmodelle (Digital Mock-up, DMU) [And06]. Zielsetzung ist die aktuelle, konsistente Verfügbarkeit multipler Sichten auf Produktgestaltung, -funktion und technologische Zusammenhänge, wodurch mit der Modellierung und Simulation eine verbesserte Gestaltung der Auslegung erfolgen kann

[SK97]. Das üblicherweise aus geometrischen CAD-Modellen bestehende DMU wird mit weiteren Metadaten ergänzt, sodass sich der resultierende Datensatz für computergestützte physikalische Simulationen eignet [MB 23].

Dieser ganzheitliche Lösungsansatz verspricht kürzere Entwicklungszeiten auch bei gleichzeitig steigender Variantenvielfalt [MB 23]. Im Gegensatz zu physischen Versuchsmodellen zeigen sich deutliche Kosteneinsparungen bei deren Realisierung, Aktualisierung und Pflege [AB98]. Zudem entsteht ein besserer Informationsaustausch. Kleinste Details können aus verschiedenen Blickpunkten betrachtet, Bilder digital überall auf der Welt geteilt und Animationen für die Validierung sowie Visualisierung von Abläufen erstellt werden [AB98, S.135]. Spur und Krause [SK97] definieren daher das virtuelle Produkt als das primäre, digitale Auslegungsmodell mit dem Ziel der Simulation aller Phasen eines Produktes im rechnergestützten Produktentwicklungsprozess. Dabei werden alle notwendigen Daten des Produktes für sämtliche Phasen abgebildet [SK97]. Der Modellierungsablauf eines Produktes ist in 3.1 dargestellt.

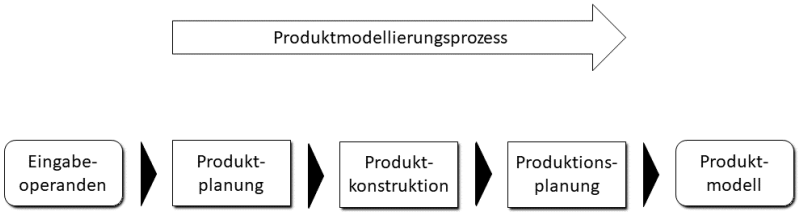



Abbildung 3.1: Ablauf der virtuellen Produktmodellierung nach [SK97, S. 3].

Die digitale Transformation, abgeleitet von der Digitalisierung, ist ein fortschreitender Prozess, der Unternehmen die Möglichkeit bietet, Maschinen und Geräte durch die Integration von Informationssystemen zu steuern und zu verwalten [PV22]. Dabei könnte die DMU Plattform eine Basis für die Einbeziehung weiterer Aspekte im Entwicklungsablauf wie Produktkonzeption, Design und Konstruktion darstellen. Langfristig können dadurch Prozesse automatisiert und neue Technologiebausteine in bestehende Produktstrukturen integriert werden. Die Möglichkeit der Datenerfassung, -speicherung, -übertragung und -auswertung erhöhen die Erwartungen an zukünftige Entwicklungskonzepte [WSW21].

Zwei Konzeptbegriffe haben sich dabei besonders etabliert: der Digitale Schatten (Digital Shadow) und der Digitale Zwilling (Digital Twin, DZ) [KKT+18]. Da die Begriffe Digitaler Schatten und Digitaler Zwilling häufig synonym verwendet werden, ist der Zusammenhang zusammen mit dem Digitalen Master/Prototyp in Abbildung 3.2 nach dem Positionspapier der Wissenschaftlichen Gesellschaft für Produktentwicklung (WiGeP) dargestellt. In der digitalen Fabrik findet der Digitale Zwilling Anwendung, aber auch in der Produktentstehung ermöglichen Digitale Zwillinge neue Geschäfts- und Wertschöpfungsmodelle [SATW20]. Die drei Begriffe unterscheiden sich im Grad der Datenintegration (Abbildung 3.3) zwischen dem physischen Produkt und dem digitalen Gegenstück [KKT+18]. Die drei Begriffe Digitales Produktmodell, Digitaler Schatten und Digitaler Zwilling werden im folgenden Abschnitt näher erläutert. Vor dem Hintergrund der zunehmenden Verknüpfung heterogener Modelle im digitalen Produktentwicklungsprozess und der Forschungsfrage dieser Arbeit bildet die Klärung

und begriffliche Abgrenzung eine wesentliche Grundlage für das Verständnis datenbasierter Zusammenhänge, der Modellinteroperabilität und einer digitalen Durchgängigkeit für die Beschleunigung der Produktentwicklung.

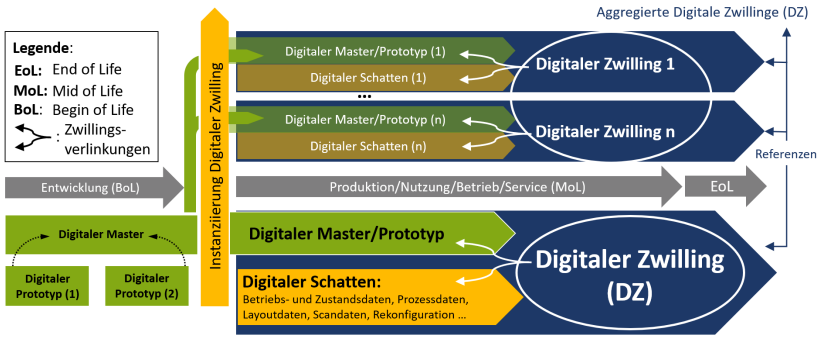



Abbildung 3.2: Definitionen des Digitalen Zwillings, Digitalen Masters/Prototyps und Digitalen Schattens in Abhängigkeit von Produkt (unten) und Produktinstanz (oben) von [SATW20].

### Digitales Produktmodell

In dieser Arbeit wird der Begriff Digitales Produktmodell in Anlehnung an ISO 10303 folgendermaßen definiert:

**Definition 3.1.2 Digitales Produktmodell**

„Ein Produktmodell ist die formale Beschreibung aller Informationen zu einem Produkt über alle Phasen des Lebenszyklus hinweg.“ [ISO24]

Ein digitales Produktmodell wird manuell erzeugt. Bei einem digitalen Produktmodell, analog zu dem Digitalen Master (Abbildung 3.2), existiert kein automatisierter Datenaustausch zwischen dem physischen und dem digitalen Objekt (Abbildung 3.3). Es stellt eine digitale Darstellung eines Objektes dar, bestehend aus mehreren Modellen. Dies können unter anderem Simulationsmodelle, verhaltenbeschreibende Modelle oder Geometriemodelle sein, die keine automatische Datenintegration verwenden. Dabei können digitale Daten bestehender physischer Systeme verwendet werden, die allerdings manuell eingepflegt werden müssen. Diese zentralen digitalen Produktdatenmodelle werden unter anderem als Digitaler Master bezeichnet [SATW20, Abu11]. Eine Änderung des physischen Objekts hat keine direkte Auswirkung auf das digitale Objekt und umgekehrt [KKT+18]. Somit ist ein Digitales Modell überwiegend statisch und existiert isoliert. Mit dem fortschreitenden Entwicklungsstand wächst die Anzahl verknüpfter Datensätze und der Informationsgehalt in dem Produktmodell [Abu11]. Aus den digitalen Produktmodellen können graphische oder textuelle Darstellungen technischer Sachverhalte (z.B. Technische Zeichnungen oder Stücklisten) für die Dokumentation oder Weitergabe ausgegeben werden [And06]. Das Ziel der digitalen Produktmodelle ist die Erstellung Digitaler Prototypen und die rechnerische Absicherung der Funktionalität [SATW20].

### Digitaler Schatten

Der Digitale Schatten ist eine Teilmenge des Digitalen Zwillings [SATW20]. Nach Stark et al. bildet der Digitale Schatten die Betriebs-, Zustands- und Prozessdaten realer Produktinstanzen ab [SATW20]. Zusätzlich enthält der Digitale Schatten Informationen über seine Herstellung. Von der Wissenschaftlichen Gesellschaft für Produktionstechnik (WGP) wird der Digitale Schatten in der Industrie 4.0 als das „hinreichend genaue Abbild der Prozesse in der Produktion, der Entwicklung und [der] angrenzenden Bereiche[] mit dem Zweck, eine echtzeitfähige Auswertungsbasis aller relevanten Daten zu schaffen“ beschrieben [BKRS16]. Dies ermöglicht eine kontinuierlich aktualisierte und dynamische Abbildung der Betriebsdaten. Für die Erstellung sind eine Beschreibung der erforderlichen Datenformate, die Auswahl der Daten und die Granularität wichtig [SDT18]. Der Digitale Schatten wird in Anlehnung an [SATW20] wie folgt definiert:

**Definition 3.1.3 Digitaler Schatten**

„Der Digitale Schatten [ist ein] Abbild der Betriebs-, Zustands- und Prozessdaten der realen Produktinstanz, auch [in Bezug auf seine] Herstellung [].“ [SATW20]

Im Gegensatz zum Digitalen Produktmodell findet ein automatisierter und einseitiger Datenfluss vom physischen zum digitalen Datenmodell statt (Abbildung 3.3). Der Digitale Zwilling kann produktspezifische Daten der zugehörigen Produktinstanz wie Mess-, Fertigungs- oder Rekonfigurationsdaten entweder selbst erfassen, verwalten und verarbeiten oder bilateral mit der physischen Instanz des Produkts kommunizieren [SATW20]. Ein Zugriff auf diese produktinstanzspezifischen Daten kann aber ebenfalls durch den Zugriff über dessen Teilmenge, dem Digitalen Schatten, erfolgen [SATW20]. Auf Basis dieser Daten kann aufbauend mit Simulationen und Prozessmodellen der DZ ein nahezu „identisches Abbild der Realität liefern“ [BKRS16].

### Digitaler Zwilling

Das Konzept des Digitalen Zwillings findet seinen Ursprung im Jahr 2002 in der Gründung eines Produktlebenszyklusmanagement (Product Lifecycle Management, PLM) Zentrums an der Universität von Michigan [GV16]. Das PLM Zentrum enthält alle Elemente eines DZ [GV16]. Historisch entwickelt hat sich das Konzept des Digitalen Zwillings aus der Luft- und Raumfahrtindustrie und breitete sich von dort auf viele weitere Bereiche aus, wodurch unter anderem bislang keine einheitliche wissenschaftliche Definition des Begriffs vorliegt [SD19]. Die NASA definiert den Digitalen Zwilling als eine integrierte, multiphysikalische, deterministische und probabilistische Simulation eines Systems, die den Zustand des physischen Objektes zeitnah unter Verwendung physikalischer Modelle und historischer Daten widerspiegelt [SCD+10]. In der Literatur wird der Digitale Zwilling überwiegend als ein dynamisches Abbild eines realen Produktes bezeichnet, dass eigenständig beobachtet, bewertet, agiert und nicht nur eine reine digitale Kopie darstellt [BKRS16, MLKS22, SDT18]. Dieser Ansatz beruht auf der Erstellung hochauflösender digitaler Modelle, um das Leben eines Produktes zu spiegeln und in einer virtuellen Umgebung zu testen [JSH+20, SW18]. Ein Digitaler Zwilling basiert dabei auf dem Wissen und den Informationen, die entweder in der

frühen Lebenszyklusphase des Produktes aus dem Digitalen Prototyp/Master[1] oder direkt aus der realen Produktinstanz abgeleitet wurden [SATW20]. In dieser Arbeit wird der Begriff Digitaler Zwilling nach CIRP[2] Encyclopedia of Production Engineering definiert als:

**Definition 3.1.4 Digitaler Zwilling**

„Ein Digitaler Zwilling ist eine digitale Darstellung einer aktiven, einzigartigen Produkt[instanz] [...], das seine ausgewählten Merkmale, Eigenschaften, Bedingungen und Verhaltensweisen mittels Modellen, Informationen und Daten innerhalb einer einzigen oder sogar über mehrere Lebenszyklusphasen hinweg umfasst.“ [SD19]

Demnach ist der Digitale Zwilling eine digitale Repräsentation einer Produktinstanz (z.B. physisches Gerät) oder einer Instanz eines Produkt-Service-Systems [SATW20]. Zudem enthält dieser Merkmale, Verhalten sowie Zustände und verbindet je nach Lebenszyklusphase unterschiedliche Modelle, Informationen oder Daten miteinander, die aus dem Digitalen Schatten oder Digitalen Master bereitgestellt werden [SATW20]. Instanziiert wird der DZ entweder auf Basis des Digitalen Masters oder durch die Datenerfassung einer realen Produktinstanz (Abbildung 3.2) [SATW20].

Das Ziel des Digitalen Zwillings ist die kontinuierliche Bereitstellung von Informationen und Daten eines Produktes während der Produktion, als auch darüber hinaus. Mit einem DZ kann der gesamte Lebenszyklus eines Produktes überwacht und Vorhersagen über das zukünftige Verhalten gemacht werden. Zudem sind Auswirkungen am Produkt durch die Integration neuer Funktionen oder Technologien im Entwicklungsprozess durch den Digitalen Zwilling frühzeitig analysierbar. Aufgrund der Abhängigkeit zwischen den erfassten Informationen und der jeweiligen Anwendung erzeugt jedes einzelne physische Produkt seine eigene digitale Repräsentation [LHK21].

Für die konkrete Umsetzung Digitaler Zwillinge müssen die bisherigen digitalen Modelle erweitert werden [SATW20]. Einerseits müssen Schnittstellentechniken sowie Datenübertragungsfunktionalitäten ergänzt werden, um den DZ für Verhaltensvorhersagen oder Entscheidungen laufend mit aktuellen Daten von der Produktinstanz zu aktualisieren. Die Daten vom Ist-Zustand werden dabei häufig durch Sensoren erfasst [MLKS22]. Andererseits muss ein aktiver Datenrückfluss aus dem DZ zurück erfolgen, um mit den daraus gewonnenen Informationen und dem abgeleitetem Wissen bestehende Produkte zu optimieren oder für die Entwicklung zukünftiger Produktgenerationen zu nutzen [SATW20]. Für Vorhersagen existiert eine Integration für den bilateralen Datenaustausch und die Kommunikation mit der physischen Instanz durch den Zugriff auf z.B. Fertigungs-, Montage- oder Instandhaltungsdaten (Abbildung 3.3) [SAMW17]. Der Digitale Zwilling kann dadurch als Kontrollinstanz für das physische Objekt agieren, sodass Zustandsänderungen in beide Richtungen zu Änderungen führen [KKT+18].

[1] Der Digitale Master oder Prototyp beinhaltet die Produktgeometrie und die verhaltensbeschreibenden Modelle des abgebildeten Produkts oder Systems [SATW20].

[2] College International pour la Recherche en Productique: weltweit führende Organisation in der produktionstechnischen Forschung.

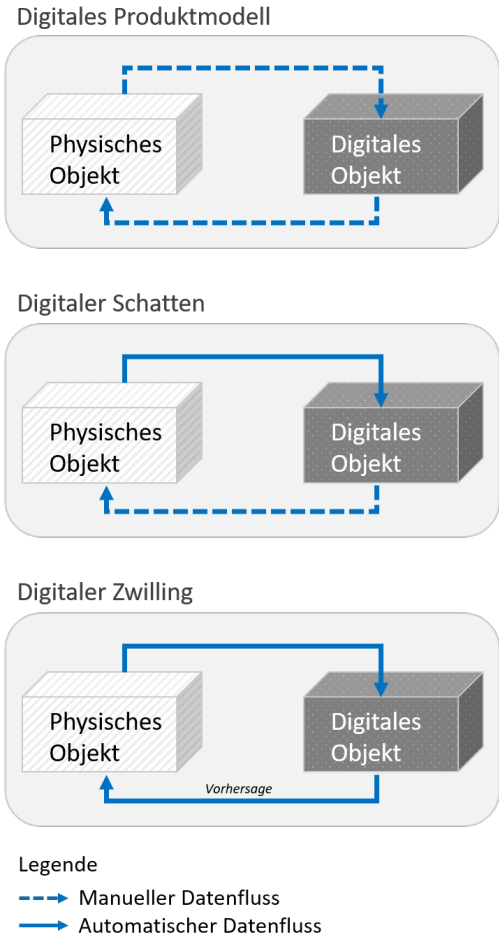



Abbildung 3.3: Unterscheidungsmerkmal Datenintegration für das Digitale Produktmodell, den Digitalen Schatten und den Digitalen Zwilling, modifiziert von [KKT+18].

Der Digitale Faden ist eine „datengesteuerte Architektur mit gemeinsam genutzten Ressourcen (z.B. Sensoroutput und Methoden)“ , die Informationen aus dem gesamten Produktlebenszyklus miteinander verknüpft [SW18]. Zudem enthält dieser alle Informationen, die für die Erzeugung und Aktualisierung des digitalen Zwillings notwendig sind [SW18]. Damit ist der Digitale Faden kein direkter Datenlieferant, sondern eine Dateninfrastruktur. McGuinnes et al. sehen den Digitalen Faden als Digitalisierung der Verbindung eines Wissenspunkts mit jedem Ort, an dem dieser referenziert, genutzt, beobachtet oder beeinflusst wird [MLO23]. Eine Grundvoraussetzung hierfür ist die Entwicklung und Implementierung von verknüpften Datenmodellen, die nicht über einen dokumentenbasierten Informationsaustausch kommunizieren [SW18]. Ein Digitaler Zwilling kann in Form von hochauflösenden Rechenmodellen oder in einer

Kombination verschiedener Modelle und Tools mit hinreichender Genauigkeit repräsentiert werden [SW18].

In der Luftfahrt wird weitestgehend dokumentenbasiert entwickelt. Dabei werden digitale Datensätze verwendet. Digitale Modelle der Produkte sowie eine ganzheitliche und vernetzte Datenkette für Digitale Zwillinge existieren noch nicht oder werden erst erforscht [Deu23a]. Ein Grund hierfür ist, dass jeder Fachbereich wie Fertigung, Wartung, Betrieb oder Entwurf andere Anforderungen an den Digitalen Zwilling stellt. Dabei werden unterschiedliche Aspekte und Modelle mit verschiedenen Abstraktionsniveaus gebraucht, um Aussagen und Vorhersagen über das Produkt tätigen zu können.

Das Konzept des Digitalen Zwillings für die Abbildung und Vorhersage des Zustands und des Verhaltens des physischen Produkts aus unterschiedlichen Sichtweisen ist für die Luftfahrtindustrie vielversprechend. Das Flugzeug als Serienprodukt stellt ein komplexes Produkt, aufgrund seiner speziellen Produktstruktur, des langen Lebenszyklus und der vielen Retrofitprozesse dar. Eine digitale Unterstützung im gesamten Entwicklungsprozess für Entscheidungen, Informationsanalysen oder Verhaltenssimulationen ist daher angestrebt. Dafür wird allerdings eine durchgängig vernetzte und interoperable Modellkette benötigt [SATW20]. Der Digitale Master und die modellbasierte Systementwicklung zu Beginn der Produktentstehungsphase liefern die Vorbereitung des Digitalen Zwillings und müssen daher Modelle in einer Qualität bereitstellen, die – im Kontext des jeweiligen Modells und dessen Abstraktionsebene – eine hinreichend realitätsnahe Abbildung der Produktinstanz ermöglicht [SATW20]. Der folgende Abschnitt gibt einen Überblick über das Potential der virtuellen Produktentwicklung für den Produktentstehungsprozess von Flugzeugkabinen. Zudem werden die dabei auftretenden Herausforderungen herausgearbeitet. Damit wird der zugrundeliegende industrielle Kontext dieser Arbeit für den Bedarf der Entwicklung einer Unterstützung in der praktischen Anwendung zur Modellierung von Produktvarianten näher vorgestellt.

### 3.1.2 Potential der virtuellen Produktentwicklung für die Flugzeugkabine

Bei der Entwicklung von Flugzeugen und besonders der Kabine werden multidisziplinäre Teams zusammengebracht, die unterschiedliche Fähigkeiten vereinen und verschiedene technische Disziplinen bedienen. Dabei sind die Teams einem hohen kooperativen Umfeld ausgesetzt mit dem Ziel alle relevanten Kompetenzen zusammenzuführen, den Zeit- und Arbeitsaufwand bei der Flugzeugentwicklung zu reduzieren und die Qualität zu erhöhen. Durch die zunehmende Digitalisierung und Automatisierung verschiebt sich die Entwicklung weiter in eine virtuelle Auslegung. Den Mittelpunkt bei der Auslegung von Flugzeugkabinen soll beim Hersteller Airbus ein digitales Mock-up liefern (Abbildung 3.4). Mit diesem definieren und bestimmen die beteiligten Entwickler gemeinsam beispielsweise die Hauptgeometrie des Flugzeugs oder die Position von Systemen und Ausrüstung [Air23b]. Darüber hinaus soll das digitale Mock-up bei der Beurteilung von Konstruktionen im Hinblick auf Wartbarkeit während des Betriebs des Flugzeugs unterstützen. Weitere Analyse- und Optimierungsfunktionen und modellbasierte Ansätze sollen den Entwurfsprozess bei der Erkundung

des Entwurfsraums und Finden der bestmöglichen Lösung unterstützen. Diese digitale Durchgängigkeit über die Fachbereiche und Abteilungen hinweg funktioniert allerdings noch nicht. Während in einigen Bereichen bereits vermehrt digitale Modelle zum Einsatz kommen und miteinander kommunizieren, ist die Kopplung auf übergeordneter Ebene oder mit mehr als einer weiteren Fachdisziplin noch nicht möglich. Voraussetzung dafür ist sicherzustellen, dass den Modellen das gleiche Gesamtsystemverständnis zugrunde liegt und alle Experten über dieselbe Flugzeugdefinition sprechen.

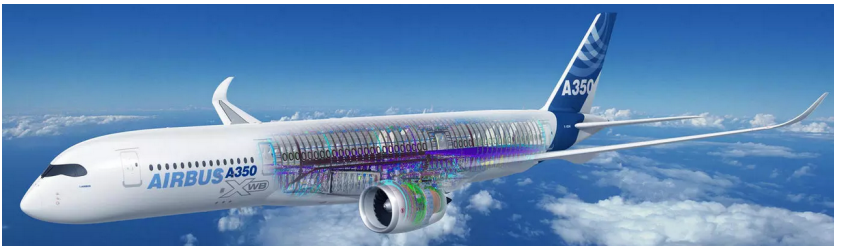



Abbildung 3.4: Digitales Modell eines Airbus A350 XWB von [Air23b].

Wie am Beispiel Airbus aufgezeigt, bringen digitale Mock-ups als zentrales Element ein großes Potential mit sich. Darauf aufbauend ist das Ziel, durch die Anreicherung mit Real-Daten vom physischen Endprodukt und Simulationsdaten, das digitale Mock-up zu einem Digitalen Zwilling hin zu erweitern. Ein übergeordnetes Metamodell verwaltet dann die Basisdaten und erzeugt für jeden n-ten Konfigurationsdatensatz n passende Digitale Zwillinge. Der abgeleitete Digitale Zwilling wird je nach Implementierung dazu genutzt, beispielsweise eine vorausschauende Wartung zu planen oder durch die bereitstehenden historischen Datensätze früherer Produktgenerationen den modularen Leichtbau von Flugzeugkabinen voranzutreiben [LHK21]. Besonders bei der Aus- und Umrüstung der Innenausstattung in der Kabine nach sieben bis acht Jahren kann von dem Digitalen Zwilling profitiert werden [LHK21]. Enthält dieser alle verfügbaren Entwicklungsdaten zusammen mit den tatsächlichen Geometrien sowie den Betriebs- und Zustandsdaten, können Umbauprozesse präziser geplant und Bodenstandzeiten reduziert werden. Rauscher et al. zeigten bereits den Vorteil einer vorzeitigen Geometrieanalyse zwischen den Referenzdaten der entworfenen Kabine und der Ist-Geometrie des sich im Betrieb befindenden Flugzeugs [RFG+23]. Durch die umfassende Digitalisierung des Produktlebenszyklus verspricht man sich ein voll digitales komplexes System, das automatisiert und flexibel auf neue Anpassungen reagieren kann. Die wichtigsten Aspekte sind die Verbesserung von Kommunikation und Konnektivität, das Erhöhen der Sichtbarkeit, das Schaffen von Transparenz bei der Datenverarbeitung und Wissen sowie das Erhöhen der Vorhersagekapazität durch Simulationen und Optimierungsfunktionen [JSH+20, S. 3].

Neben der Industrie befasst sich die Forschung ebenfalls mit der virtuellen Auslegung von Flugzeugen. Das Deutsche Zentrum für Luft- und Raumfahrt stellte in seiner neuen Luftfahrtstrategie vor, zukünftig als virtueller Hersteller (Virtual OEM) seine ganzheitlichen Forschungskompetenzen unter anderem für die Energiewende in der Luftfahrt bereitzustellen [Deu21a]. Die fortschreitende Digitalisierung und Virtualisierung ermöglichen eine beschleunigte Entwicklung und Integration technologischer In-

novationen in bestehende Prozessstrukturen, wodurch sich die Zeit bis zur Marktreife potenziell verkürzen lässt. Ziel des DLRs ist als virtueller Hersteller mit Hilfe der digitalen Durchgängigkeit von Entwurf, Produktion und Betrieb Entwicklungskosten und -risiken zu senken [Deu21a]. Neben dem Entwurf neuer Konzepte werden somit ebenfalls optimierte Produktionsprozesse ermittelt, die die Bauteile und Fertigungsprozesse in Echtzeit überwachen und begleiten [Deu23b]. Srinivasan et al. zeigten in ihrer Arbeit bereits einen ersten Ansatz für die Kopplung zwischen virtueller Entwicklung von Flugzeugkabinen mit Roboteroperationen für den Kabinenmontageprozess in einer Vormontagezelle [Sri22, SMW+21]. Weiteres Potential wird sich durch den Einsatz von immersiven Technologien versprochen. Durch die Nutzung immersiver Technologien verspricht man sich eine Erweiterung der digitalen Versuchsmodelle in ihrer Funktion und eine interaktive Auseinandersetzung mit dem virtuellen Produkt. Erste Untersuchungen für den Einsatz von immersiver Technologie in Kombination mit DMUs sind Airbus (Visualisierung der Ausrüstung in der Produktionsumgebung für Montagearbeiter) [CDB+18], Lockhead Martin Tactical Aircraft Systems (Bewertung und Visualisierung der Wartungsfreundlichkeit eines Designs) [AB98] und Forschungen an der Beihang University (immersives System zur Überprüfung und Bewertung der Wartbarkeit) [GZC+18].

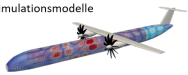
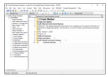
Insgesamt verspricht der Einsatz von digitalen Produktmodellen viele Vorteile bei der Betriebsdatennutzung, der Bewertung neuer Anwendungsfälle und der Vorhersage vom Produktverhalten. Erste Ergebnisse aus der Literatur zeigen das Potential der virtuellen Eigenschaftsabsicherung vernetzter Modelle. Dennoch bleiben die zwei größten Herausforderungen die Modellbildung und deren Vernetzung [DAG+21].

### 3.1.3 Angewandte Toolarten im Kabinenentwurf

Der Entwurfsprozess für Flugzeugkabinen hat sich in den letzten Jahren nur langsam weiterentwickelt. Eine Vorstellung des heutigen Produktentstehungsprozess von Flugzeugen ist in Anhang A.1.1 gegeben. Eine Einführung in die Vorgehensweise des Kabinen- und Systementwurfs erfolgt in Anhang A.1.2. Im Entwurfsprozess der Kabine und deren Systeme werden verschiedene Aspekte untersucht und berechnet. Einerseits werden Simulationen für die Bestimmung der Akustik in der Kabine für Lärmschutzmaßnahmen (Ansys) [HB21] und numerische Crashsimulationen (FEM) im Falle einer Notlandung auf starrer Oberfläche und auf dem Wasser (Ditching) [SH15] benötigt. Andererseits werden mit parametrischen Geometriemodellen (CAD) die Kabinenkomponenten im vorhandenen Rumpfbauraum des Flugzeugs platziert und modifiziert. Darüber hinaus müssen die generierten Kabinenkonzepte hinsichtlich des Passagierkomforts und der Akzeptanz mit Probandentests untersucht [RMME+22], Passagiersimulationen für den (De-)Boarding-Prozess [EH19] durchgeführt und Strömungsanalysen (CFD) für die Aerosoldispersion kalkuliert [SSSW23] werden. Die Einsatzfähigkeit von abstrakten Systemmodellen und im Speziellen die Nutzung des Cameo Systems Modelers (CSM) für die funktionale Auslegung der Kabinensystemkomponenten mit Darstellung der Datenflüsse wird bei Airbus noch erprobt [MdSPF23]. Eine vielfältige Toollandschaft mit unterschiedlichen Fidelitätsgraden für die Auslegung und Berechnung von Kabinen im Entwurfsprozess ist die Folge. Tabelle 3.1 zeigt eine Kategorisierung der Toolarten, die im Flugzeug- und Kabinenentwurf zur Anwendung kommen.

Einige Software-Tools, die bereits von Universitäten, der Industrie oder Forschungsinstituten für unterschiedliche Bereiche der Kabinenauslegung entwickelt wurden, sind im Folgenden aufgezählt. Dazu gehören die wissensbasierte Entwurfsmethodik Fuselage Geometry Assembler (FUGA) von Walther [Wal24], das Konzeptentwurfswerkzeug PADlab der TU Berlin [PAD23], das Open Vehicle Sketch Pad (OpenVSP) der NASA [MG22] für parametrische Geometriemodellierung von Flugzeugentwürfen oder das Flugzeugentwurfsberechnungstool Aircraft Preliminary Sizing Tool (PreSTo) von Scholz an der Hochschule für Angewandte Wissenschaften Hamburg [Sch24], das Kabinen- und Rumpfentwurfstool CAFE [EH24] mit der open-source Passagierflusssimulation PAXelerate von Bauhaus Luftfahrt [EDH20]. Weiterhin zeigten Prokic et al. [PEM24] mit ihrer wissensbasierten Flugzeugentwurfsanwendung Robust Aircraft Parametric Interactive Design (RAPID) eine Kombination eines analytischen Tools mit einer 3D-Umgebung und Virtueller Realität. Der Schwerpunkt liegt auf der Berechnung von Massen und der Platzierung von Monumenten (Bordküche, Toilette, Sitze) im dreidimensionalen Kabinenbereich unter Berücksichtigung von einfachen Anforderungen. Systeme wie die elektrische Versorgung oder die Klimatisierung werden hingegen in den genannten Tools nur rudimentär oder gar nicht berücksichtigt.

Tabelle 3.1: Übersicht über Tools im Flugzeugkabinenentwurf

| Technologien | Beschreibung | Verwendung/Anwendung | Beispiele (Software & Materialien) |
|---|---|---|---|
| CAD (Computer-Aided Design) Software  | Software zur Erstellung von detaillierten 3D-Modellen von Flugzeugkabinen, einschließlich Sitzanordnungen, Innenraumkomponenten und Einrichtungen. | Erstellung virtueller Darstellungen des Kabinendesigns und zur Visualisierung des Gesamtlayouts und der Konfiguration. | AutoCAD, CATIA, SolidWorks, Rhino |
| Mockups und physische Modelle  | Physische Darstellungen des Kabinendesigns, einschließlich Mockups und maßstabsgetreuen Modellen, die für visuelle und funktionale Bewertungen verwendet werden. | Für physische Tests, Ergonomiebewertungen und Validierung des Designs, indem sie eine greifbare Darstellung des Kabinenlayouts und der Gestaltungselemente bietet. | Schaumstoff, Karton, Holz, Kunststoff, 3D-Druck |
| Virtuelle Realität (VR)  | Immersive digitale Umgebung, die es Anwendern ermöglicht, das Kabinendesign virtuell zu erleben und zu bewerten. | Für virtuelle Rundgänge und interaktive Simulationen, um das Aussehen und das Gefühl des Kabinendesigns aus der Perspektive des Benutzers zu bewerten. | Unity, Unreal Engine, Autodesk VRED |
| Berechnungs- und Simulationsmodelle  | Werkzeug, um visuellen Darstellungen, Prototypen und Simulationen des Systems oder der Komponente zu erstellen, um Anforderungen zu validieren und zu überprüfen. | Ermöglichen virtuelle Tests, Analysen und Optimierungen des Systemverhaltens bevor physische Prototypen gebaut werden. | Matlab/ Simulink, Ansys (Fluent), OpenFOAM, Abaqus, Nastran, Patran |
| Anforderungsmanagement Software  | Zentrale Plattform für Erfassung, Dokumentation, Verfolgung und Analyse von Anforderungen während des gesamten Produktentwicklungszyklus. | Anwendung für die System- und Softwarespezifikation im Rahmen des Anforderungsmanagement. | IBM DOORS, PTC Integrity |
| Abstrakte Systemmodelle  | Plattform zur grafischen Modellierung von Systemen, Hardware- und Softwarearchitekturen. | Funktionalität von Anwendungen früh validieren, Anforderungen überprüfen und Durchführung technischer Analysen zur Bewertung. | Cameo Systems Modeler, IBM Rhapsody, Capella, Papyrus, PTC Integrity |

Fuchte et al. [FNG12] untersuchten eine Methodik für das Layout großer Rumpfsysteme im Flugzeugvorentwurf. Diese Methode liefert eine geometrische Analyse und einen Pfadfindungsalgorithmus für das Kabelrouting. Mit diesem vorläufigen digitalen Mock-up können dann geeignete Architekturen in der frühen Phase des Entwurfs identifiziert werden.

Die Auslegung von Systemen wurde ebenfalls in der Arbeit von Motzer [Mot16] berücksichtigt. Er entwickelte mit Hilfe einer graphenbasierten Entwurfssprache den regelbasierten Aufbau von digitalen Geometriemodellen für den Rumpf und die Kabine inklusive einiger Systeme und deren vollautomatisierter Verkabelung. Vorteile dieses Ansatzes sieht Motzer in dem automatisierten Abrufen und der Anwendung des regelbasierten Wissens für die vielfältige Variantenbildung und -untersuchung.

Wie man für ein System Ziele modellieren, dann für eine Optimierung nutzen und anschließend die Ergebnisse visualisieren kann, zeigten Johannsson et al. mit ihrer Methode MOSART (Multi-objective Optimization for Safety and Reliability Trade-off) [JDÖ17, S. 406]. Als Anwendungsfall untersuchten sie die Auslegung eines Flugzeugtreibstoffsystems. Dabei zeigten sie, wie man durch die Methode Designschleifen reduzieren und die Nachverfolgbarkeit von absolvierten Schritten im Konzeptdesign verbessern kann.

Einige Tools sind primär für eine erste visuelle Darstellung gedacht. Verwendete Anforderungen oder generierte Daten während des Prozesses können im fortschreitenden Entwicklungsverlauf nicht mehr bis zum Ursprung zurückverfolgt werden, sodass sich systemübergreifende Einflüsse durch Parameteränderungen nur schwer nachvollziehen lassen. Zudem fehlt die Berücksichtigung von Funktionalitäten, sodass eine Erweiterung des Entwurfsprozesses notwendig ist. Der nächste Schritt ist daher neben einer geometrischen Auslegung auch eine eigenschaftsbezogene Auslegung der Systeme zu schaffen, sodass auch beispielsweise Spannungen und Ströme im Rahmen der Auslegung untersucht werden können. Ein konsistentes Modell, das all die Informationen miteinander verknüpft, fördert die multidisziplinäre Auslegung und verbessert zudem die Zusammenarbeit zwischen den Teams einzelner Disziplinen.

Anstelle von open-source Lösungen testet Airbus die kommerzielle Innovationsplattform 3DEXPERIENCE von der Firma Dassault Systemes für die digitale Entwicklung und Fertigung von Produktlinien sowie die abstrakte Systemmodellierung [Das19]. Diese Plattform dient als zentrale Informationsquelle und vereint Software wie unter anderem den Cameo Systems Modeler für die Modellierung von Produkten. Die Plattform ermöglicht die 3D-CAD-Modellierung und Anforderungsmodellierung in einem Systemaufruf. Darüber hinaus bietet kommerzielle Software Vorteile durch vertraglich festgehaltene Zuverlässigkeit und Wartung der Software sowie Betreuung bei Aktualisierungen der Betriebssysteme. Da die Industrie vermehrt kommerzielle Software in der industriellen Flugzeugentwicklung einsetzt und vor allem der Cameo Systems Modeler als Tool weit verbreitet ist, wird in dieser Arbeit der CSM für die Modellierung genauer untersucht [FW22, Das19].

### 3.1.4 Herausforderungen und Grenzen des Kabinenentwurfs

> *„[...] every approximately 7 to 8 years an aircraft is retrofitted with a complete or partly new interior.“ [LHK21, S. 687]*

Allgemein beträgt der Produktentwicklungsprozess eines Flugzeugs bis zu 20 Jahre. Der Prozess vereinigt viele Fachdisziplinen und Experten. Indes ist eine abnehmende Fertigungstiefe bei den Flugzeugherstellern zu beobachten [Hin19]. Systeme, Komponenten und Teile werden durch darauf spezialisierte Zulieferer entwickelt [Hin19].

Rund 80% des Produkts werden durch Unterauftragsnehmer geliefert [Air23c]. Die rechnerische Auslegung und Dimensionierung der Systeme durch die Experten sowohl beim Supplier als auch beim OEM erfolgen teilweise durch selbst programmierte Codes und teilweise mit kommerzieller Software wie Matlab/Simulink. Jedes Produktdatenmodell besitzt andere Detailtiefen, um die fachspezifischen Fragestellungen zu beantworten. Das Resultat ist eine breite und heterogene Toollandschaft. Während teilweise in Fachbereichen einige Tools und deren Modelle bereits miteinander kommunizieren, erfolgt beim Datenaustausch über die Fachdisziplin und darüber hinaus auch über die Unternehmensgrenzen hinweg ein Bruch in der Kommunikation. Hier findet die Kommunikation überwiegend in Form von textbasierten Dokumenten statt. Die überlieferten Informationen müssen manuell in die Modelle bei den Zulieferern oder des anderen Fachbereichs eingepflegt werden. Dadurch sind systemübergreifende Einflüsse durch Änderungen in einem Fachbereich auf weitere Subsysteme der Kabine nicht unmittelbar erkennbar.

Besonders für die Kabine sind die Rückverfolgung von Änderungen und Darstellung systemübergreifender Einflüsse relevant. Der Grund sind Modifikationen und Nachrüstungen (Retrofit). Die Lebensspanne eines Flugzeugs beträgt bis zu 30 Jahre. Im Vergleich dazu wird eine Kabine nach sieben bis acht Jahren ausgetauscht [LHK21]. Gründe hierfür sind die starke Abnutzung durch die Passagiere, die beispielsweise durch mechanischen Abrieb, Stöße mit harten Gegenständen oder verschüttete Flüssigkeiten entsteht, sowie die individuelle Anpassung durch die Airline je nach Marketingkonzept und Auslastung der Flugstrecke[3]. Zudem variieren je nach Art der Innenausstattung die Nachrüstzyklen [Int19]. Passagiersitze werden häufiger erneuert als Gepäckfächer oder Bordtoiletten. Zeitgleich entwickeln sich Normen und Designs der Kabinenausstattung, durch Innovationen getrieben, stetig weiter. Je nach Umfang der Änderung ist die Forschung zu Wechselwirkungen innerhalb der Kabine aufwendig. Eine effiziente Nachrüstung von Kabinen ist durch die genannten Herausforderungen zeitaufwendig und fehleranfällig, da häufig Daten veraltet oder nicht vorhanden sind [LHK21].

Zudem besteht die Flugzeugkabine aus verschiedenen interoperablen Elementen, die auf engem Raum nebeneinander bestehen müssen [Alt16]. Dabei müssen unter anderem strenge Einbaubedingungen z.B. in Bezug auf Vibrationen oder Brandgefahr berücksichtigt werden. In stark beanspruchten Bereichen wie der Kabinendecke, in der viele verschiedene Systeme nebeneinander angeordnet sind, müssen bei der Installationsplanung die Anforderungen an jedes Subsystem berücksichtigt werden [Alt16]. Die Modifikation an einer Subsystemkomponente kann durch geänderte Abmaße oder Anforderungen z.B. Auswirkungen auf die Art der Installation in die Rumpfstruktur haben, wodurch wiederum weitere Kabinensystemgruppen beeinflusst werden können. Die Komplexität nimmt zu und in Folge dessen müssen Entwickler verschiedenster Systemgruppen mehr miteinander kommunizieren [PS92].

> *„In the [Boeing] 707, the systems were relatively simple in function, and thus in complexity, and had little interaction with each other. [...] This allowed each system group to design part of the airplane with only a small degree of systems coordination. By the time the Boeing 757 and 767 (circa 1980) were being developed, system functionality*

[3] Die Airlines bauen auf Strecken, die im Jahr unterschiedlich frequentiert sind, sogenannte Sommer- und Winterbestuhlungskonzepte ein. Dabei variiert innerhalb der Kabine der Sitzabstand, um eine maximale Auslastung der Flüge zu erreichen.

> *had grown dramatically in complexity and interdependency. No longer could these airplane systems be developed independently. System designers were required to achieve a higher degree of coordination and communication with designers of other systems. This system complexity and interdependency design evolution is continuing and has required a more explicit application of system engineering on the Boeing 777 than ever before.“ [PS92]*

Durch die zunehmenden Modifikationen steigt die Komplexität, ebenso der Aufwand für das Daten- und Wissensmanagement [LRK22]. Für die Umsetzung und Lösung der auftretenden Herausforderungen wird eine digitale Durchgängigkeit zwischen allen heterogenen Modellen benötigt, um alle am Prozess beteiligten Disziplinen frühzeitig in einen Austausch zu bringen und miteinander zu vernetzen. Dadurch können bei Änderungen im System, die durch Variantenvielfalt oder hinzukommende Wünsche im Rahmen des Customizings entstehen, die Auswirkungen frühzeitig erkannt, Effekte berechnet und Bewertungen durchgeführt werden. Eine (frühzeitige) Vernetzung aller Disziplinen im Entwurf ermöglicht durch eine digitale Durchgängigkeit einen Austausch zwischen den Disziplinen und verbessert letztendlich die Entwicklungszeiten und die Qualität des Produktes.

### 3.1.5 Schlussfolgerungen

Der vorangegangene Abschnitt gibt einen Einblick in den Flugzeug- und Kabinenentwurf. Dabei hat jeder Fachbereich seine eigenen Modelle mit unterschiedlichen Abstraktionsgraden sowie ein unterschiedliches Systemverständnis. Dazu gehören CAD Software, Virtuelle Realität, Berechnungs- und Simulationstools, Anforderungsmanagement Software, abstrakte Systemmodelle, aber auch physische Modelle. Grund hierfür sind die vielen Disziplinen, die nach Entwurfsstadien andere Detaillierungsgrade benötigen und ihre eigenen Toolwelten bereitstellen. Dadurch sind diese Modelle bereits entsprechend ihrer Fragestellung optimiert. Einfluss auf die heterogene Toollandschaft hat neben den Fachdisziplinen auch die hohe Beteiligung an externen Experten durch die Vielzahl am Entwicklungsprozess beteiligten Zulieferer. Folglich entstehen verschiedene Modelle desselben Systems – bedingt durch unterschiedliche Fachperspektiven, Modellierungsziele und eingesetzte Tools [Moh12].

Durch die zunehmende Digitalisierung erweitert sich diese Toollandschaft ständig. Zudem bestehen Unterschiede zwischen Industrie und Forschung. Eine bestehende Herausforderung bleibt der überwiegende Einsatz kommerzieller Tools in der Industrie, während in der Forschung open-source Umgebungen bevorzugt werden [Cam22, DDD+21]. Gründe für die Nutzung kommerzieller Software sind eine fortlaufende Unterstützung durch den Anbieter, regelmäßige Updates und Sicherheitsaspekte. Gleichzeitig führt dies zu eingeschränkten Anpassungsmöglichkeiten der Tools und deren Funktionalität. Zusätzlich ermöglichen kommerzielle Software häufig einen Informationstransfer nur über einige wenige Schnittstellen und erschweren damit den Export von Daten zu anderen Modellierungsumgebungen.

Darüber hinaus ist die Kabine geprägt durch Modifikationen und Anpassungen. Durchschnittlich wird die Innenausstattung der Kabine alle sieben bis acht Jahre ausgetauscht. Die Untersuchung zur Integration neuer Kabinenmodule in bestehende Flugzeugstrukturen ist zeitaufwendig. Wechselwirkungen zu anderen Systemen innerhalb

der Kabine können aufgrund der unzureichenden Datenlage oder der fehlenden Kopplung zwischen den Fachbereichen nur bedingt dargestellt werden. Nur durch eine durchgängige Kopplung der Modelle können das Gesamtsystem ganzheitlich modelliert und systemübergreifende Wechselwirkungen erfasst werden. Zudem wird sich durch die fortschreitende Digitalisierung sowohl die Anzahl der Modelle als auch die Modellkomplexität erhöhen. Dementsprechend sind zukünftig Lösungsansätze gesucht, die den steigenden Anforderungen an Interoperabilität, Komplexitätsbeherrschung und Datenmanagement begegnen. Daher muss die in dieser Arbeit entwickelte Unterstützung einerseits die Modellierung von Teilaspekten des Systems mit unterschiedlichen Modellierungstools berücksichtigen und andererseits die Kopplung dieser heterogenen Modelle miteinander ermöglichen. Ziel ist es, dadurch die Entwicklung neuer Flugzeuge zu beschleunigen und einen Beitrag in Richtung Digitaler Zwillinge zu leisten.

## 3.2 Einsatz von modellbasierten Ansätzen im Systemvorentwurf

*„To handle increasing complexity in product development, model-based systems engineering (MBSE) approaches are well suited, in which the technical system is represented in a system model." [BSJ+22, S. 1]*

Wie bereits im vorangegangenen Abschnitt vorgestellt, kommen bei der Entwicklung der Flugzeugkabine verschiedene Modelle und Entwicklungsumgebungen zum Einsatz. Vor allem Modelle zur geometrischen Beschreibung und zur mechanischen Analyse von Komponenten und Baugruppen werden verwendet. Beispiele für diese Art der Modelle sind CAD, FEM oder CFD. Sie besitzen einen hohen Detaillierungsgrad, um möglichst genaue Aussagen über das Verhalten des abzubildenden Systems treffen zu können. Um ein komplexes und technisches System wie die Kabine zu entwickeln, ist die alleinige Nutzung dieser hochauflösenden Modelle allerdings nicht ausreichend. Zudem würde die Abbildung des Gesamtsystems in nur einem Modell die induzierte Komplexität stark erhöhen. Besonders im Anfangsstadium der Entwicklung werden Modelle mit einer geringeren Detailtiefe benötigt. Gleichzeitig müssen komplexe Systemstrukturen aufgebrochen und besser verstanden werden, bevor diese mit detailreichen Modellen erfasst werden können. Das Systems Engineering (SE) stellt hierfür Ansätze bereit, um ein System mit einem höheren Abstraktionsgrad abzubilden. Im Fokus steht die Betrachtung des Systems im Ganzen und auf einer höheren Ebene (top-level). Durch diesen Ansatz wird die Komplexität des Systems abgefangen und handhabbar gemacht. Besonders das modellbasierte Systems Engineering (MBSE) stellt für diese Herausforderung Ansätze bereit.

In diesem Abschnitt wird der Frage nachgegangen, inwieweit mit Hilfe von MBSE die Komplexität beherrschbar gemacht werden kann und welche Herausforderungen dieser Ansatz für die Anwendung in der Kabinenauslegung mit sich bringt. Zuerst folgt ein Einblick in die Grundlagen des Systems Engineerings und Model-based Systems Engineerings sowie eine Abgrenzung der Themen zueinander. Zudem werden die drei Bestandteile Methoden, Sprachen und Tools der Modellbasierten Systementwicklung näher beschrieben. Im Anschluss werden einige Konzepte und Projekte vorgestellt,

die bereits den modellbasierten Ansatz im luftfahrttechnischen Bereich oder im Kabinendesign angewandt haben. Abschließend wird ein Einblick in die Herausforderungen und Grenzen modellbasierter Ansätze gegeben.

### 3.2.1 Einführung ins Systems Engineering (SE)

In den letzten fünfzig Jahren hat sich der Begriff des Systems Engineering im Zusammenhang mit der Bearbeitung von vielfältigen Produkten etabliert und gefestigt [oos19]. Unter dem Begriff des SE wird ein interdisziplinärer und integrativer Ansatz für die erfolgreiche Entwicklung und Umsetzung von technischen Systemen in großen Projekten verstanden. Ein Fokus liegt hierbei auf der Definition und anschließenden Dokumentation der Anforderungen in einer frühen Entwicklungsphase und Überprüfung des zu entwickelnden Systems auf dessen Einhaltung [oos19]. Dieser Entwicklungsprozess vom Konzept bis zur Betriebsphase integriert dabei verschiedene Disziplinen, wie z.B. die Bereiche Projektmanagement, Qualität, Tests, Planung und Entsorgung. Anwendung finden hierbei unterschiedliche Systemprinzipien und -konzepte sowie wissenschaftliche Managementmethoden. Das Zusammenspiel der sowohl technischen als auch wirtschaftlichen Aspekte ergibt dann eine ganzheitliche Betrachtung und schafft ein gemeinsames Systemverständnis [oos19]. Über die letzten Jahre hinweg hat sich der SE-Ansatz weiterentwickelt und neue Konzepte und Methoden hervorgebracht. Die drei wichtigsten Definitionen und Sichtweisen des Systems Engineering sowie die zugrunde liegenden Systemdefinitionen werden im Folgenden vorgestellt.

#### SE-Ansatz der ETH Zürich

Für die Bearbeitung von komplexen Sachverhalten hat sich in den vergangenen Jahren das Systems Engineering des Betriebswissenschaftlichen Zentrums (BWI) der ETH Zürich als Standardmethode in allen Ingenieursbereichen durchgesetzt [HWFV12]. Mit dieser Problemlösungsmethodik, bestehend aus der Modellvorstellung und dem Systemansatz, können komplexe Sachverhalte strukturiert werden. Dies lässt sich ebenfalls für die problemgerechte Lösung komplexer betriebswissenschaftlicher Problemstellungen anwenden und geht damit über den rein technischen Aspekt hinaus [HWFV12]. Ein System wird nach Haberfellner et al. als eine Ansammlung aus in Beziehung stehender Teile definiert, die zusammen ein Ganzes bilden [HWFV12]. Diese Teile oder auch Elemente haben Eingenschaften oder Funktionen und ihrerseits wieder selbst als ein System betrachtet werden [HWFV12].

**Definition 3.2.1 Technisches System**

Ein technisches System ist eine Ansammlung aus in Beziehung stehender Teile, die zusammen ein Ganzes bilden. [HWFV12]

Das Systems Engineering ist ein Zusammenspiel aus dem Systemdenken und dem Vorgehensmodell. Ersteres dient als Hilfsmittel für die Strukturierung von Situationen, um diese zusammenhängend zu verstehen und voneinander abzugrenzen [HWFV12, S. 27 ff.]. Das Vorgehensmodell wiederum beschreibt unterschiedliche Komponenten

oder Prinzipien, die zur Unterteilung der einzelnen Schritte bei der Lösungsfindung angewandt werden [HWFV12]. Dabei wird der Entwicklungsprozess in Phasen unterteilt, Systemvarianten berücksichtigt und vom Groben ins Detail (top-down) vorgegangen. Weitere Einblicke zu Vorgehensmodellen, wie dem V-Modell, werden in 3.2.1.3 gegeben. Zusammen unterstützen das Systemdenken und das Vorgehensmodell ein ganzheitliches Denken und das Verständnis komplexer Systeme, um die bestmögliche Lösung zu finden. Eine vereinfachte Darstellung des SE-Konzepts nach Haberfellner et al. ist in der folgenden Abbildung 3.5 dargestellt.

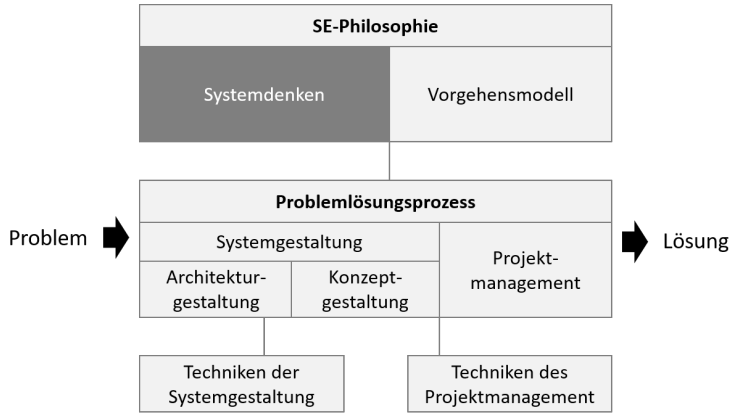



Abbildung 3.5: Bestandteile der SE-Philosophie nach [HWFV12].

Haberfellner et al. heben in ihrem Ansatz hervor, dass eine Beschreibung des Systems durch Modelle wichtig ist und sich das Systemdenken durch unterschiedliche Perspektiven auf das System auszeichnet [HWFV12]. Die Modelle zeigen ein spezifisches Problem der Realität in abstrakter und vereinfachter Weise [EDA17]. Dabei wird ein Modell für einen bestimmten Zweck erstellt. Die von Haberfellner et al. gezeigten Ansätze dienen als Basis für die Systemmodelldarstellung und wie das System betrachtet werden soll [HWFV12].

#### SE-Ansatz der NASA

Die NASA versteht unter dem Systems Engineering einen methodischen und multidisziplinären Ansatz für das Design, die Umsetzung, das technische Management sowie für den Betrieb und die Stilllegung eines Systems [Nat07]. Der Ansatz zielt darauf ab, ein funktionsfähiges System zu entwickeln, welches die gestellten Anforderungen trotz häufig widersprüchlicher Einschränkungen erfüllt [Nat07]. Ziel des SE ist es, die Beiträge unterschiedlicher Disziplinen zu bewerten, gegeneinander abzuwägen und zu integrieren, um ein einheitliches, holistisches Ganzes zu schaffen [Nat07]. SE ist demnach eine logische Denkweise und die Art, das Gesamtbild zu betrachten, während man technische Entscheidungen trifft.

Analog zur Begriffsdefinition eines Systems von Haberfellner et al., wird ein System bei der NASA als eine Kombination von Elementen beschrieben, die miteinander

wechselwirken, um eine Fähigkeit zu ermöglichen und damit einen bestimmten Bedarf zu erfüllen [Nat07]. Elemente können dabei sowohl die Hardware oder Software sein, als auch Verfahren, Personal oder Prozesse, die für das Erzielen von Ergebnissen auf Systemebene erforderlich sind. Ergebnisse können Eigenschaften, Verhalten oder Qualitäten sein. Die Beziehungen zwischen den einzelnen Elementen beschreiben deren Verbindung miteinander und ermöglichen erst einen übergeordneten Mehrwert, der über den individuellen Beitrag hinausgeht.

#### SE-Ansatz von INCOSE

Für die Bearbeitung von komplexen Produkten in der Industrie, wie z.B. in der Luftfahrtbranche, hat sich in den vergangenen Jahren das Systems Engineering Konzept vom International Council on Systems Engineering (INCOSE) ebenfalls etabliert. Das von zahlreichen Mitgliedern verschiedenster INCOSE Arbeitsgruppen entwickelte Konzept fokussiert einen interdisziplinären Ansatz für die Realisierung erfolgreicher Systeme durch die Betrachtung der Gesamtproblematik [WRF+15, INC19]. Im Gegensatz zu den beiden vorherigen Ansätzen verfolgt dieser Ansatz eine standardisierte, international normierte und modellbasierte Herangehensweise und schafft damit eine formalisierte Grundlage für die disziplinübergreifende Zusammenarbeit. Das SE berücksichtigt geschäftliche und technische Anforderungen der Kunden. Ziel ist, ein qualitativ hochwertiges Produkt abzuliefern und die Anforderungen des Benutzers zu erfüllen. Der Fokus liegt darauf, frühzeitig im Entwicklungsprozess die Kundenanforderungen und notwendige Funktionalitäten zu definieren. Dabei werden die Anforderungen dokumentiert und anschließend die Systementwürfe und -validierungen durchgeführt, wobei das gesamte Problem berücksichtigt wird.

Die System Begriffsdefinition von INCOSE leitet sich von Ludwig von Bertalanffy (1968) ab, der ein System als ein Ganzes betrachtet, das aus interagierenden Teilen besteht [WRF+15]. INCOSE ergänzt, dass ein technisches System menschengemacht ist und geschaffen wurde, um für den Nutzer oder andere Stakeholder ein Produkt oder eine Dienstleistung in einer definierten Umgebung bereitzustellen [WRF+15]. Es besteht aus einer Menge von Elementen, Subsystemen oder Baugruppen, die in Kombination eine oder mehrere festgelegte Zielsetzungen erreichen [WRF+15]. Dabei bezieht sich die Definition auf ein System aus der realen Welt, während ein Systemkonzept die mentale Repräsentation des tatsächlichen Systems darstellt.

Trotz ihrer methodischen Unterschiede eint die SE-Ansätze von der ETH Zürich, INCOSE und NASA ein gemeinsames Verständnis: Systems Engineering dient als strukturierendes, systematisches Denk- und Kommunikationsmittel, um in komplexen, interdisziplinären Kontexten ein gemeinsames Systemverständnis zu schaffen. SE wird als Mittel gesehen, die bestehende Komplexität im Engineering zu managen, die allerdings vorwiegend in den frühen Phasen der Produktentwicklung zum Einsatz kommt [DAG+21]. Im Zentrum steht nicht die Methode selbst, sondern der Mensch als Entscheidungs- und Verständnisträger, dem durch Modelle, Prozesse und klare Strukturen die Zusammenarbeit erleichtert und ein zielgerichteter Austausch ermöglicht wird. In Zukunft ist ein Anstieg der Entwicklungskomplexität zu erwarten [DAG+21]. Der Fokus in der Entwicklung rückt dann auf die Interaktion zwischen technischem System, Anwendern und Nutzern, sodass eine menschenzentrierte Gestaltung und die Mensch-Maschine-Interaktion mehr Bedeutung erhält.

#### 3.2.1.1 Komplexität im Systementwurf

Die stetig voranschreitende Entwicklung von Technologien und der Ausbau von Funktionen führt zu mehr Komplexität bei Produkten und Systemen [EETM12]. Während man in den 1990er mit einem Mobiltelefon nur telefonieren konnte, sind heutzutage weitaus mehr Funktionen, wie im Internet surfen oder fotografieren möglich [She18]. Zu komplexen Systemen gehören zum Beispiel Flugzeuge und Autos. Gleichzeitig werden kürzere Entwicklungszeiten dieser Produkte bei gleichbleibender Qualität gefordert, um neue Technologien einzubauen oder den Marktanforderungen gerecht zu werden.

Die kontinuierliche Weiterentwicklung durch Hinzufügen oder Herausnehmen von Systemelementen und -fähigkeiten stellt damit neben der kontinuierlichen Verbesserung der Leistungssteigerung eine der zentralen Herausforderungen dar. Nur dadurch können komplett neue Konfigurationen entwickelt und bessere Lösungen im Vergleich zu bereits vorhandenen Konzepten erreicht werden. Wie jüngst anhand der Covid-19-Pandemie zu sehen, müssen sich Unternehmen der neuen Situation mit ihren Produkten und Strukturen anpassen [Sin20]. Dies verlangt einen flexiblen und leicht anzupassenden Produktentwicklungsprozess, ohne dabei die geltenden Anforderungen oder Richtlinien zu verletzten. Ein Treiber hierbei ist die Digitalisierung, die globale Wertschöpfungsketten, komplexe Netzwerke und Kollaborationen zwischen crossfunktionalen Fachbereichen fördert.

In dieser Arbeit nimmt der Begriff Komplexität mit seinen Eigenschaften eine zentrale Rolle ein. Daher wird in diesem Abschnitt der Begriff definiert und eingeordnet. Im Duden wird Komplexität mit „Vielschichtigkeit“ erläutert [Cor23]. Im Bezug auf Systeme wird für diese Arbeit allerdings der Begriff Komplexität in Anlehnung an Schuh folgendermaßen definiert [Sch05]:

**Definition 3.2.2 Komplexität**

„Komplexität [ist] eine Systemeigenschaft, deren Grad von der Anzahl der Systemelemente, von der Vielzahl der Beziehungen zwischen diesen Elementen sowie der Anzahl möglicher Systemzustände abhängt.“ [Sch05]

Die Komplexität ist die Eigenschaft eines Systems, viele Zustände oder Verhaltensweisen annehmen zu können. Komplexe Systeme werden als die Gesamtheit einer Reihe charakteristischer Eigenschaften gesehen, die ein System oder Modell mitbringen und die wiederum miteinander interagieren. Durch die steigende Anzahl an Elementen oder Verknüpfungen zwischen den Elementen steigt auch die Komplexität. Weitere Aspekte wie Anzahl an Funktionen und Unüberschaubarkeit erhöhen ebenfalls die Komplexität.

Demnach können vier grundsätzliche Systemtypen unterschieden werden. Diese sind einfache Systeme (wenig Elemente, Beziehungen und Verhaltensmöglichkeiten), komplizierte Systeme (viele Elemente und Beziehungen, Verhalten ist deterministisch), relativ komplizierte Systeme (wenig Elemente und Beziehungen, hohe Vielfalt an Verhaltensmöglichkeiten) und äußerst komplexe Systeme (Vielzahl von Elementen mit vielfältigsten Beziehungen, große Vielfalt an Verhaltensmöglichkeiten) [Sch05]. Bei beiden komplexen Systemen ist eine vollständige Beherrschbarkeit nicht möglich und

bei äußerst komplexen Systemen treten zusätzlich veränderliche Wirkungsverläufe zwischen den Elementen auf. Abbildung 3.6 zeigt schematisch die vier Systemtypen.

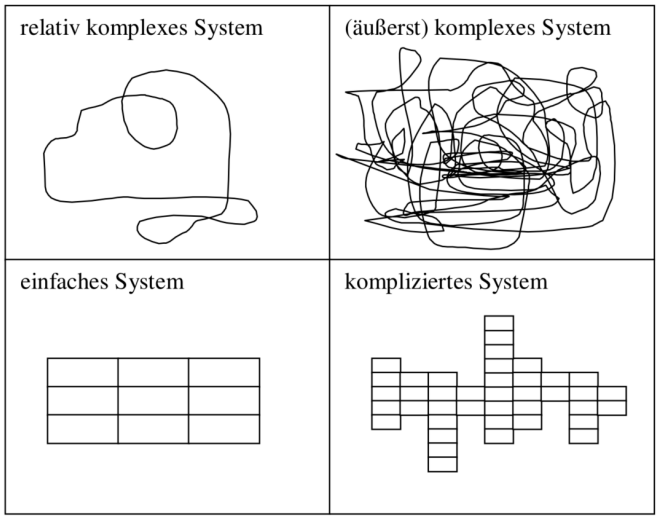



Abbildung 3.6: Schematische Darstellung von einfachen, komplizierten, relativ und äußerst komplexen Systemen von [Dob01].

Ein komplexes System wird von Haberfellner et al. beschrieben als ein System, bei dem systemweite Wechselwirkungen zwischen den einzelnen Elementen auftreten und welches eine große Vielzahl unterschiedlicher Verbindungen und Elemente aufweist [HWFV12]. Ein Fokus liegt hierbei auf der Systembetrachtung. Haberfellner et al. bevorzugen ein systemhierarchisches Denken. Dadurch wird einer Unübersichtlichkeit der vielen Elemente und deren Beziehungen entgegengewirkt. Ein betrachtetes System wird in seine Untersysteme aufgegliedert und in den jeweiligen Ebenen nur die jeweils wesentlich erscheinenden Elemente samt Beziehungen dargestellt, gemäß dem Black-Box-Prinzip. Bei einer Black-Box-Betrachtung wird der innere Aufbau nicht betrachtet [HWFV12]. Stattdessen werden nur der Eingang und der Ausgang des Systems näher ausgeführt. Die nach hierarchischen Ebenen ausgeführte Auftrennung eines Systems ist in der Abbildung 3.7 zu sehen.

Die Herausforderungen, die solche komplexen Systeme mit sich bringen, sind Nachverfolgbarkeit von Anforderungen, wechselseitige Abhängigkeiten und Unübersichtlichkeit. Um dieser Komplexität gerecht zu werden und die Fragestellungen der Fachdisziplinen zu beantworten, wurden viele heterogene, isolierte Softwarelösungen geschaffen und damit die Unübersichtlichkeit verstärkt [SBWF10]. Beispielsweise sind dadurch Änderungen innerhalb eines Subsystems und deren Einfluss auf andere Systeme nicht direkt ersichtlich [SBWF10]. Darüber hinaus verändern und entwickeln sich die Systeme zu unterschiedlichen Zeitpunkten im Auslegungsprozess [TE20].

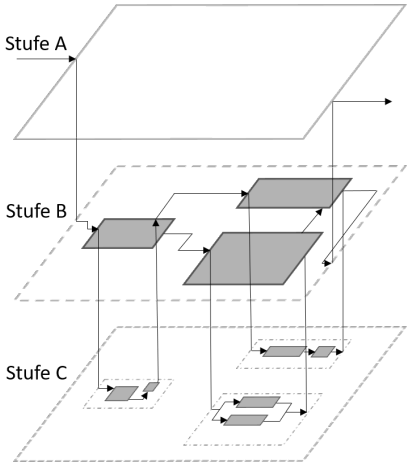



Abbildung 3.7: Stufenweise Auftrennung eines Systems in seine Untersysteme nach [HWFV12].

Die Konzeptauswahl beruht in der frühen Konzeptphase auf langjährigen Erfahrungen von Experten, die durch praktische Anwendung erworben wurde. Dieses Wissen ist oft intuitiv, implizit und kontextabhängig, wodurch es nicht vollständig extrahiert und in ein digitales Format überführt werden kann. Zudem können aufgrund des hohen Aufwands für zeitintensive Analysen (z.B. Finite-Elemente-Methode) nur einige Konzepte ausgewählt werden, sodass eventuell vielversprechende Alternativkonzepte vorschnell verworfen werden [NZW+12]. Gleichzeitig sind die virtuellen Modelle und Analysen in ihrer Detailtiefe wesentlich genauer, als es für eine solche frühe Bewertung erforderlich ist.

#### 3.2.1.2 Allgemeine Beschreibung des RFLP-Ansatzes

Bei der Entwicklung und dem Entwurf komplexer und multidisziplinärer Produkte müssen die vielfältigen Kundenanforderungen, Systemfunktionen und Funktionsprinzipien im Rahmen eines gemeinsamem Produktmodells beschrieben werden. Hierbei treffen unterschiedliche Disziplinen aufeinander, die jeweils eigene Methoden oder rechnerunterstützte Systeme (CAx) nutzen [KK13]. Nur eine integrierte Entwicklungsumgebung mit einer durchgängigen Datenanbindung ermöglicht eine holistische Betrachtung des Gesamtsystems. In früheren Entwicklungen fehlten integrierte Informationsmodelle, die alle Produktentwicklungsphasen und Disziplinen umfassen [KK13]. Eine Lösung bietet der RFLP-Ansatz. Mit diesem ist eine ganzheitliche Unterstützung für die Entwicklung mechatronischer Systeme, basierend auf der Philosophie des Systems Engineering, geschaffen worden.

Der Begriff RFLP setzt sich aus dem Anforderungsmanagement (Requirement Engineering, R), dem funktionalen Design (Functional Design, F), dem logischen Design (Logical Design, L) und dem physikalischen Design zusammen (Physical Design, P).

In der ersten Abstraktionsebene, dem R, werden die Anforderungen des Kunden gesammelt und zur Verwaltung in einem Modell abgelegt. Die Anforderungen sind während des gesamten Auslegungsprozess global abrufbar und werden automatisch synchronisiert, damit diese immer auf dem neuesten Stand sind. In der zweiten Ebene, dem F, werden Funktionen von den Anforderungen abgeleitet und eine funktionale Architektur aufgebaut. Die Funktionen unterteilen sich wiederum in Teilfunktionen und werden miteinander zu Gruppen verlinkt. Anschließend erfolgt auf der dritten Abstraktionsebene, dem L, die Zuordnung der Funktionen zu logischen Elementen und beantwortet die Frage, woraus das System besteht. Logische Architekturelemente werden später durch physikalische Elemente als finale Lösung realisiert. Abschließend wird auf Ebene 4, dem P, die technische Perspektive dargestellt. Im Gegensatz zur logischen Perspektive werden nicht nur Wirkkonzepte beschrieben, sondern konkrete Technologien und physikalische Elemente genannt. Mit den verschiedenen Abstraktionsebenen wird eine strukturierte Überleitung bereitgestellt, um Anforderungen besser in physische Produktdaten zu transformieren [LLL20]. Dadurch wird sichergestellt, dass zuerst definiert wird, *was* das System tun soll (Funktionen), bevor dann definiert wird, *wie* das System es tut (Form) [BC23, S. 5]. Zusammengefasst beschreibt der RFLP Prozess die systematische Produktentwicklung von der Systemanalyse bis hin zu Systementwicklung und ergänzt die Systembildung im V-Modell (siehe 3.2.1.3) [KK13].

#### 3.2.1.3 Das V-Modell

Im Bereich der mechatronischen Systementwicklung sind in den letzten Jahrzehnten diverse methodische Ansätze entstanden, die zusammengefasst durch die Richtlinie VDI 2206 repräsentiert werden. In einem mechatronischen System verschmelzen die Fachdisziplinen Mechanik, Elektronik und Informatik miteinander und es entsteht ein synergetisches Zusammenwirken dieser. Die neueste Ausgabe der VDI 2206 mit einem erweiterten V-Modell wurde im November 2021 vom Verein Deutscher Ingenieure e.V. (VDI) herausgegeben[4]. Zusammengefasst stellt diese SE-Methode einen Support für eine disziplinübergreifende Entwicklung mechatronischer Systeme sowie eine end-to-end Entwicklungsumgebung für diese Systeme bereit [KK13]. Mit dieser allgemeinen Richtlinie ist einerseits eine methodische Unterstützung für den Ingenieur zur disziplinübergreifenden Entwicklung mit Fokus auf den Systementwurf für die frühe Phase der Entwicklung geschaffen worden [EDA17]. Andererseits betrachtet die VDI den gesamten Entwicklungsprozess ausgehend von den Anforderungen bis zum fertigen Produkt. Insgesamt setzt sich die Richtlinie aus drei Elementen zusammen, einem allg. Problemlösungszyklus (Mikrozyklus), dem V-Modell (Makrozyklus) und vordefinierten Prozessmodulen für wiederkehrende Arbeitsschritte [EDA17]. Ziel ist die Umsetzung einer disziplinübergreifenden Systemarchitektur.

Ursprünglich entstand die Idee eines V-Modells für die technische Entwicklung im Anwendungsbereich der Softwareentwicklung für die interdisziplinäre Produktentwicklung im Jahr 1995 durch Bröhl und Dröschel [GH20]. Das V stellt dabei auf der linken Achsenseite die Zerlegung des zu untersuchenden Systems in seine Elemente dar, während auf der rechten Achsenseite die nacheinander stattfindende Integration von Teilsystemen zum technischen Gesamtsystem erfolgt [GH20]. Grundlage bildet die

[4] www.vdi.de/2206

Erhebung und Analyse von Anforderungen. Anschließend wird die Gesamtfunktion eines Systems in seine wesentlichen Teilfunktionen zerlegt und die geeigneten Bedienprinzipien oder Lösungselemente zugeordnet [EDA17]. Die Bewertung des Abschnittes Systemarchitektur und Design sollte idealerweise mit den Nutzern durchgeführt werden [GH20]. Im untersten Abschnitt des V-Modells findet die Implementierung der Systemelemente statt, wobei die mechanischen Komponenten mit Hilfe von CAD oder FEM dimensioniert werden. Zwischen den beiden Achsenseiten des V-Modells findet eine kontinuierliche Validierung und Verifizierung statt, symbolisiert durch Pfeile. Damit wird sichergestellt, dass das richtige System (Validierung) auf die richtige Weise entwickelt wird (Verifikation) [GH20]. Die Durchführung der Tests auf der rechten Achsenseite ermöglicht einen Vergleich zwischen dem Modellergebnissen und den Ergebnissen des physikalischen Systems. Die Verifikation findet auf gleicher Ebene statt, während die Validierung nach oben zum Beginn des V-Modells zeigt und das System hinsichtlich der Stakeholderanforderungen überprüft. Die Implementierung der Systemelemente findet im unteren Teil des V-Modells statt. Erst wenn eine Designiteration abgeschlossen ist, wird mit der nächsten Iteration begonnen. Als Ergebnis des V-Modell entsteht ein Produkt. In Abbildung 3.8 ist das V-Modell dargestellt.

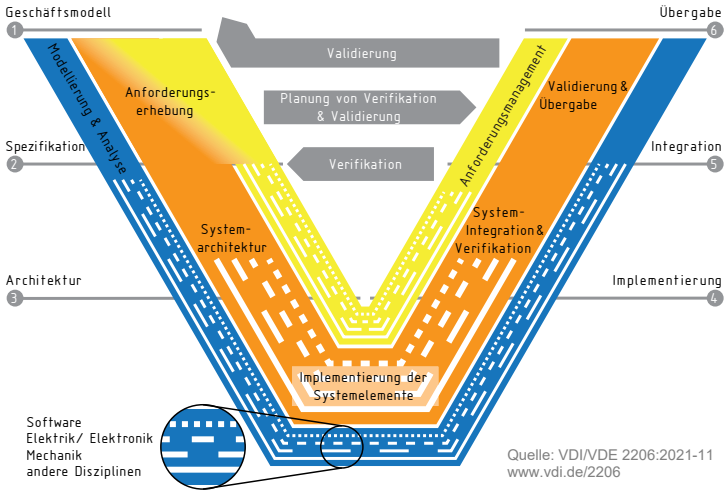



Abbildung 3.8: Makrozyklus V-Modell nach VDI 2206 [VDI21].

Maßgeblich entscheidend für den Erfolg eines Entwicklungsprojektes ist der Umgang mit Anforderungen [GH20]. Hierbei darf nicht der Eindruck entstehen, dass die Eingangsgrößen fest vorgegeben sind, sondern sich diese aus Anforderungen an das System und seine Funktionalität ableiten lassen. Darüber hinaus können sich Anforderungen im Laufe des Entwicklungsprozesses verändern. Eine durchgängige Anforderungserhebung und -verwaltung ist daher notwendig (siehe Abbildung 3.8 gelber Strang) [GH20]. Die Anforderungsvalidierung ist ein dynamischer Prozess mit vielen Iterationen [WZ20]. Dies unterstützt zusätzlich bei der Designentwicklung von Lösungen basierend auf den Nutzerbedürfnissen. Der Designprozess wird häufig als isoliert

betrachtet [LHV15]. Die Anbindung an eine durchgängige Anforderungserhebung inkludiert den Designprozess und ermöglicht eine umfassendere Sichtweise auf das zu entwickelte Systemprodukt.

### 3.2.2 Modellbasierte Systementwicklung (MBSE)

Digitale Modelle sind in der Technik seit den 1960er Jahren weit verbreitet [FHBF19]. Für eine methodischere Entwicklung von Systemen und die Einbindung eines ganzheitlichen, zusammenwirkenden Verständnisses, wurde das traditionelle Systems Engineering um visuelle Modellierungstechniken erweitert. Im Gegensatz zu den vorgestellten analytischen Modellen (z.B. CFD) aus Kapitelabschnitt 3.1.3, bei dem ein System nachgebildet wird, um Erkenntnisse des Verhaltens in der Wirklichkeit zu erlangen, wird hier ein geringerer Detailgrad verwendet und das System abstrakter dargestellt. Die visuellen Techniken ermöglichen in den Modellen unterschiedliche Ansichten auf ein System und dadurch den schnellen Zugriff auf Informationen durch reduzierte Darstellung. Durch die Weiterführung in einen modellbasierten Ansatz wurde der Fokus auf die Verwendung unterschiedlicher Modelle und deren Integration bzw. Verknüpfung untereinander gelegt [FHBF19]. Dies ermöglichte das Management von großen Datenmengen und das Verständnis von systemübergreifenden Einflüssen durch Designänderungen [oos19, HWFV12]. Damit hat die Modellbasierte Systementwicklung das Potential, schnellere und umfangreiche Einflussanalysen von Anforderungen und Designänderungen durchzuführen [FMS14].

#### Grundlagen der Modellbildung und Modellierung

Mit Hilfe von komplexitätsreduzierten Modellen können Produkte und Systeme untersucht und optimiert werden [Har09]. Modellhafte Abbildungen dienen als Hilfsmittel, um das Verständnis komplexer Zusammenhänge zu verbessern [Abu12]. Abbildung 3.9 zeigt links als Original das System der Passagierfunktionen in einer Flugzeugkabine und rechts ein Modell von diesem. Das Modell veranschaulicht die funktionalen Eigenschaften und Datenflüsse. Dabei stellt das Modell ein Abbild des zu modellierenden Originals dar. Es ist eine Abstraktion des Originals und stellt nur notwendige Eigenschaften, abhängig von Zweck des Modells, dar [Dan03]. Beschrieben wird jedes Modell nach Glinz „als Menge von Individuen und Attributen [...]“ [Gli05]. Individuen beschreiben eindeutig abgegrenzte Gegenstände. Die Attribute hingegen sind die Eigenschaften der Individuen, die Beziehungen zwischen und Operationen auf andere Individuen oder Attribute [Gli05].

Hauptmerkmale eines Modells sind Abbildung, Verkürzung und Pragmatismus. Geprägt wurden diese Begriffe durch die Grundlagen der Modellierung von Stachowiak mit seiner Allgemeinen Modelltheorie, veröffentlicht 1973 [Gli05, Har09]. Das Abbildungsmerkmal beschreibt, dass ein Modell das Abbild eines Originals ist. Je nach Ziel der Modellbildung kann diese auf verschiedene Aspekte begrenzt sein, darunter die Struktur, die Gestalt, bestimmte Eigenschaften, Verhaltensweisen oder Funktionen [Abu12]. Das Verkürzungsmerkmal definiert, dass jedes Modell abstrahiert dargestellt wird und dabei nur die aus Sicht des Modellschaffenden erforderlichen Attribute

der Realität abbildet. Zusätzlich kann das Modell auch Attribute und Individuen enthalten, die nicht im Original existieren. Das pragmatische Merkmal beschreibt, dass jedes Modell zu einem spezifischen Zeitraum geschaffen und damit zu einem bestimmten Verwendungszweck modelliert wurde [Gli05]. Ziel ist die Erzeugung eines gemeinsamen Verständnisses über ein reales Objekt. Dadurch ist laut Stachowiak jedes Modell das Ergebnis eines selektiven Bewusstseinsprozesses, da sich dasselbe Original je nach Betrachtung und Fokussierung auf verschiedene Weisen darstellen lässt [Sta73].

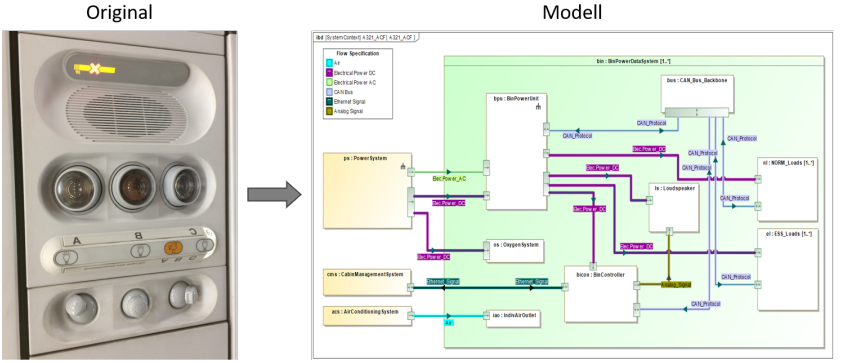



Abbildung 3.9: Modell als abstraktes Abbild eines Originals am Beispiel der Passagierservicefunktionen in der Flugzeugkabine. Das Modell zeigt die funktionalen Eigenschaften und Datenflüsse des Originalsystems.

Ein weiteres Merkmal von Modellen ist der Modellierungszweck. Hierbei wird zwischen deskriptiven, präskriptiven und analytischen Modellen unterschieden. Deskriptive Modelle beschreiben und erklären Systeme, basierend auf vorhandenen Daten oder Beobachtungen. Präskriptive Modelle hingegen helfen bei der Vorhersage, indem sie noch nicht existierende Originale abbilden [Sta73, Lud89]. Analytische Modelle sind im Vergleich zu den beiden anderen Modellen ausführbar und helfen mit mathematischen Techniken Simulationen für die Untersuchung von Verhaltensweisen zu erzeugen. Ein digitales Produktmodell ist in der Regel eine Kombination aus deskriptiven, präskriptiven und analytischen Elementen, je nachdem in welcher Phase des Entwicklungsprozesses es verwendet wird.

### Einführung ins Model-based Systems Engineering

Im Jahr 2007 wurde der Begriff des Model-based Systems Engineerings (MBSE) durch INCOSE geprägt und bekannt gemacht [oos19]. INCOSE definiert MBSE als:

**Definition 3.2.3 MBSE**

„Model-based Systems Engineering (MBSE) ist die formalisierte Anwendung der Modellierung zur Unterstützung der Systemanforderungen, des Entwurfs, der Analyse, der Verifizierung und der Validierung von Aktivitäten, die in der Konzeptionsphase beginnen und sich über die gesamte Entwicklung und spätere Lebenszyklusphasen erstrecken." [INC07, S. 15]

MBSE ist eine Methodik des interdisziplinären Ansatzes zur Entwicklung und Realisierung komplexer technischer Systeme, dem Systems Engineering [MW19]. Ziel dieses Ansatzes war eine gesteigerte Produktivität durch die Minimierung von handschriftlichen Dokumenten während der Zusammenarbeit in großen interdisziplinären Teams [Nat07]. Dabei basieren die Systeminformationen auf digitalen Modellen und nicht ausschließlich auf papier- und dokumentenbasierten Informationen und stellen damit ein disziplinübergreifendes Systemmodell in den Entwicklungsmittelpunkt [MW19]. Dazu werden Modelle zur Unterstützung der Systemanforderungen, für das Design, die Analyse sowie Verifikation und Validierungsaktivitäten, beginnend während der Konzeptphase bis hin zu den späteren Lebenszyklusphasen, entwickelt [HWFV12]. Ein Modell stellt dabei eine vereinfachte Version eines Konzepts, einer Struktur, einer Beziehung oder eines Systems dar [Bad19]. Die Modelle bilden eine Vereinfachung der Realität ab und eliminieren dabei unnötige Komponenten [Fuc18]. Dies führt zu einem besseren Systemverständnis aufgrund der unterschiedlichen Systemansichten für den Menschen, während gleichzeitig die Informationen maschineninterpretierbar gemacht werden. Im Gegensatz zu den dokumentenbasiert festgehaltenen Anforderungen bietet der Einsatz von MBSE verschiedene Ansichten auf ein System, die das Systemverständnis fördern und es den beteiligten Disziplinexperten ermöglichen, je nach Modellansicht die für sie relevanten Informationen leichter zu erfassen [Fuc18]. Allerdings werden bei der Modellierung unterstützende Elemente genutzt, die nicht im physischen, abzubildenden System existieren. Dadurch kann sich die Modellfunktionalität und auch die interne Modellkomplexität erhöhen. Darüber hinaus lassen sich mit Hilfe eines Modells Aktivitäten in der Systemumgebung wiederholen und konsistent darstellen. Die datenzentrierte Spezifikation im MBSE ermöglicht somit eine Automatisierung und Optimierung [Bad19].

Sowohl der modellbasierte als auch der dokumentenbasierte Ansatz erfüllen die Funktion der Informationsverarbeitung und -weitergabe. Der Wandel, den die Produktentwicklung erlebt, bezieht sich somit auf die Art der Informationsübermittlung [Fri18]. Im Produktentwicklungsprozess werden Dokumente vor allem am Anfang in der Planungsphase eingesetzt, um den Informationsfluss über die verschiedenen Schnittstellen zu sichern [Fri18]. Die Kernaufgaben des dokumentenbasierten Ansatzes liegen in der Sicherung, Verbreitung, Strukturierung und Verwaltung von Informationen. Zudem erfüllen die Dokumente eine Nachweisfunktion. Dabei werden allerdings Systeminformationen und Parameter über mehrere Entwicklungsumgebungen nur lose miteinander gekoppelt oder in mehreren Dokumenten wiederholt aufgeführt [Bad19]. Im Vergleich dazu sind die Kernfunktionen, die ein Modell hat, die Beschreibung, Erklärung und Prognose im Rahmen der Wissensgenerierung[5]. Beim Einsatz von Modellen werden Systeminformationen und Parameter häufig referenziert, wodurch Änderun-

[5]Nach Friedrich lassen sich Dokumente und Modelle in ihrer Kernfunktion unterscheiden [Fri18]. Dennoch können Modelle und Dokumente auch die Funktionen des jeweils anderen Konzeptes erfüllen.

gen am System nicht manuell in mehreren Entwicklungsumgebungen nachgepflegt werden müssen.

Für die Verwaltung von Produktdaten und Dokumenten während des gesamten Lebenszyklus eines Produkts wird ein Produktdatenmanagementsystem (PDM) eingesetzt. Dabei werden alle relevanten Produktinformationen zentral gespeichert und verwaltet [Sta22]. Die Grundlage hierfür bildet ein integriertes, digitales Produktmodell, das die verschiedenen Datenquellen miteinander verbindet. MBSE und PDM sind komplementäre und integrierte Ansätze. Während PDM sich auf die Verwaltung von Produktdaten, wie CAD-Dateien und Stücklisten, konzentriert, ermöglicht MBSE eine modulare und systematische Modellierung von Systemen und deren Verhalten. Eine Kombination der beiden Ansätze kann eine effiziente Verwaltung sowohl der Produktdaten als auch der Systemmodelle sicherstellen und ermöglicht eine ganzheitliche Betrachtung des Produktentwicklungsprozesses [Nat21].

Vorteile, die das MBSE im Vergleich zu dokumentenbasierten Ansätzen mitbringt, sind unter anderem mehrere Perspektiven auf das Systemmodell, verbesserte Analyse- und Auswertungsmöglichkeiten, Wiederverwendbarkeit von Wissen, einfachere Pflege und Standardisierung. Produktivitäts- und Qualitätsverbesserungen zeigen sich vor allem in der Kommunikation zwischen Entwickler und Kunde und im Management der Komplexität [Har15]. Weiteres Potenzial liegt in der Wiederverwendbarkeit, besseren Dokumentation und übersichtlicheren Produktstruktur durch verknüpfte Modelle [NFSH99]. Komplexe Zusammenhänge können analysierbar und navigierbar gemacht werden und damit ein besseres Systemverständnis schaffen [NFSH99]. Informationsverbreitung ist wichtig und der Schlüssel, damit Teams sowohl unabhängig in ihrer Disziplin (multidisziplinär) als auch im größeren Verbund (interdisziplinär) agieren können [JSM+20].

Für die Umsetzung des modellbasierten Ansatzes werden drei Elemente benötigt. Diese sind die Methode, die Sprache und das Tool (Abbildung 3.10). Die Methode enthält die Arbeitstechniken zur Umsetzung der Arbeitsschritte im angewandten Prozess [Fuc18]. Sie beschreibt den Weg, eine Aufgabe auszuführen oder ein Ergebnis zu erzielen [Mor23]. Das Tool ist das technische Instrument, um die Methode oder ihren Bediener mit seiner Arbeitstechnik zu unterstützen [Est08, Fuc18]. Es kann eine virtuelle, physische oder cyber-physische Ressource sein [Mor23]. Das Tool muss dabei die gewählte Methode und Sprache sowohl syntaktisch als auch semantisch unterstützen [BBJ+20]. Verschiedene Tools können dieselbe Methode umsetzen [Mor23]. Die Sprache wird zur Herleitung, Beschreibung und Dokumentation der Modelle genutzt, damit der Mensch oder die Maschine (Computer) diese verarbeiten kann. Wichtige Begriffe sind in dem Kontext die Notation und die Semantik. Die Notation legt die gemeinsamen Schrift- und Symbolzeichen fest und definiert Regeln (Syntax) für die Formation von Zeichenstrukturen (verbale oder graphische Elemente) [Gli05]. In der Semantik ist die Bedeutung der Zeichen und Zeichenfolgen festgelegt. Einen detaillierten Einblick und Beispiele geben die folgenden Abschnitte 3.2.3 und 3.2.4.

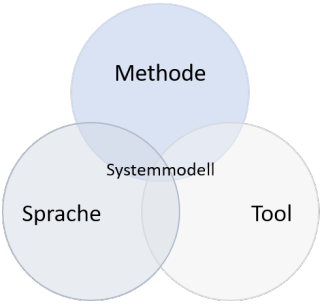



Abbildung 3.10: Die drei Bestandteile für die Erstellung eines Systemmodells nach MBSE.

### 3.2.3 Sprachen und Tools des Model-based Systems Engineering

Für die kommenden Entwicklungen verschiedener komplexer Systeme etablierten sich seit den letzten Jahren die Modellierungssprachen SysML (Systems Modeling Language) und UML (Unified Modeling Language). Auf Basis dieser Spezifikationen werden Modelle im MBSE erstellt. Im folgenden Abschnitt werden die beiden Modellierungssprachen und deren Diagramme vorgestellt.

#### 3.2.3.1 Die Unified Modeling Language (UML)

Die in den 1990er Jahren von der Object Management Group (OMG) entwickelte Modellierungssprache UML wird für die objektorientierte Softwareentwicklung eingesetzt. Die UML ist durch die ISO/IEC 19505 genormt. Mit Hilfe der grafischen Notation können komplexe Systemzusammenhänge für fachfremde Personen visuell verständlich dargestellt werden. Der abstrahierte Blick auf das Gesamtsystem ermöglicht unterschiedliche Ansichten auf das System, je nach gewählter Systemgrenze, sodass nur ein Teilausschnitt des Modells gezeigt wird. Die unterschiedlichen Ansichten eines Systems verringern die Komplexität und fördern das Systemverständnis. Hierbei gibt die UML vor, wie ein Modell spezifiziert wird, indem definiert ist, welche Begriffe verwendet werden und wie diese in Beziehung zueinander stehen. Ursprünglich war die UML vorwiegend zur Dokumentation von Softwaresystemen gedacht. Abbildung 3.11 zeigt am Beispiel der Klassen für zwei Flugzeugtypen mit spezifischen Attributen die Vererbung zur generellen Klasse Flugzeug und die binäre Beziehung zur Klasse Airline inkl. der Angabe der Multiplizität.

Die Literaturrecherche von Shaikh et al. zeigt, dass UML als Modellierungssprache in der Industrie etabliert ist. Eingesetzt wird diese zur Software-Spezifikation, -analyse, -entwurf, -dokumentation und für die Code-Generierung [SHW+21]. Die UML stellt verschiedene Modelle bereit, mit denen die verschiedenen Aspekte in der Softwaremodellierung beschrieben und abgebildet werden können. Am stärksten zum Einsatz kommt hierbei das Klassendiagramm. Mit diesem wird ein System anhand von Konzepten, ihren Beziehungen und Einschränkungen beschrieben [SHW+21]. Eine detaillierte Vorstellung der Diagramme erfolgt in Abschnitt 3.2.3.3. Beispiele für Modellie-

rungstools für die Nutzung der UML sind Microsoft Visio, IBM Rational Rhapsody oder MagicDraw. Das Einsatzgebiet ist vor allem die Auslegung von Software-Systemen. Für die Modellierung von physischen Systemen wird die SysML als Modellierungssprache bevorzugt.

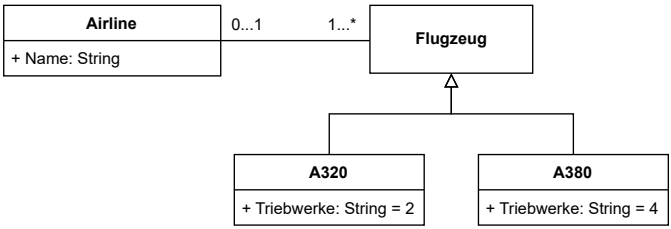



Abbildung 3.11: UML-Klassendiagramm zur grafischen Modellierung von Klassen und deren Beziehungen am Beispiel von Flugzeugen.

#### 3.2.3.2 Die Systems Modeling Language (SysML)

SysML ist eine standardisierte multidisziplinäre Modellierungssprache, die sich aus der UML 2.0 abgeleitet hat [MWH+17]. Die erste Version (SysML 1.0) wurde am 01. September 2007 veröffentlicht. Ziel der SysML ist die Unterstützung bei der Analyse, dem Design und dem Testen von komplexen Systemen, die aber nicht rein Softwaretechnischer Natur sind. Mit Hilfe der Diagramme werden nicht nur die Systemanforderungen modelliert (Abbildung 3.12) und die Struktur sowie das Verhalten eines Systems dokumentiert, sondern auch die interdisziplinäre Kommunikation zwischen den Stakeholdern unterstützt [WSSW22]. Die SysML ist eine formale, visuelle und graphenbasierte Modellierungssprache [MS18]. Dabei können formale Modelle entweder rein deskriptiv oder sowohl deskriptiv als auch ausführbar sein [MS18]. Statische Modelle sind deskriptiv und werden für die Beschreibung der Struktur eines Systems genutzt. Dynamische Modelle beschreiben das Verhalten eines Systems über die Zeit; sie sind in der Regel deskriptiv, können jedoch auch ausführbar sein, sofern sie ausreichend formalisiert sind und durch die Toolumgebung eine Simulation ermöglicht wird.

Derzeit findet die Version SysML v1.7, die im Dezember 2022 veröffentlicht wurde, eine starke Verbreitung in der Anwendung als Sprache und in vielen Modellierungsprogrammen. Damit stellt die SysML v1.7 den aktuellen Standard in der Nutzung dar. Rund zwei Jahre danach, im Dezember 2024, wurde die SysML v2 als Pilotversion veröffentlicht [Wor24]. Der Status der SysML v2 ist zum Zeitpunkt dieser Dissertation noch nicht offiziell freigegeben und daher im Reifegrad für die Modellierung im industriellen Kontext noch nicht ausgearbeitet.

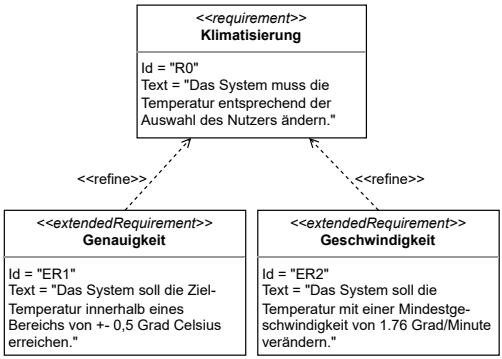



Abbildung 3.12: SysML-Anforderungsdiagramm zur diagrammbasierten Modellierung von Anforderungen am Beispiel eines Klimatisierungssystems.

Beispiele für Modellierungstools die SysML als Modellierungssprache bereitstellen sind PTC Windchill Modeler, IBM Rational Rhapsody, Capella oder Cameo Systems Modeler [SS20]. Ein im Hintergrund abgespeichertes Metamodell speichert und verwaltet die erzeugten Verbindungen zwischen den modellierten Systemelementen. Die Trennung zwischen Modellspeicher (Repository) und den Modellansichten ist ein Vorteil der SysML bei der Modellierung [MWH+17]. Während im Modellspeicher das Gesamtmodell mit allen Elementen und Verbindungen abgelegt ist, zeigen die Modellansichten nur kontextspezifische Elemente. Dies reduziert die Komplexität bei der Modellierung und Analyse des Systems. Die Umsetzung der Modellansichten erfolgt mit Diagrammen. Diese werden im folgenden Abschnitt genauer erläutert.

#### 3.2.3.3 Diagramme der UML und SysML

Sowohl die UML als auch die SysML nutzen Diagramme für die graphische Sicht auf das dahinter liegende Modell. Die Diagramme der UML kommen vor allem für die Softwareentwicklung und die Beschreibung informationstechnischer Aspekte eines Systems zum Einsatz. Für die Verwendung als Standardsprache des Systems Engineerings eignete sich die UML z.B. durch die fehlende Anforderungsmodellierung noch nicht. Zudem ist die UML stark geprägt von der Objektorientierung und kann daher laut Weilkiens zu Missverständnissen in der Projektkommunikation führen [Wei08]. INCOSE entschied daher im Jahr 2001 die UML zu erweitern und für das Systems Engineering anzupassen. Mit der Einführung der SysML wurde auf den bereits vorhandenen Diagrammen der UML aufgebaut und diese um weitere spezifische Diagramme mit Fokus aufs SE ergänzt. Damit stellt die SysML eine Anpassung an die Bedürfnisse des Systems Engineerings dar, indem UML-Elemente entfernt wurden, die im SE nicht benötigt werden (z.B. Softwarekomponenten) [Wei08]. Nach Weilkiens hat die SysML den Anwendungsbereich der Modellierungssprachen weiter vergrößert [Wei08]. Stattdessen ermöglicht die SysML die Modellierung von Geschäftsprozessen und die Beschreibung der physischen Eigenschaften (Attribute) sowie Funktionen von Systemen [Wei08].

Im Rahmen dieser Arbeit wird die SysML aufgrund ihrer zunehmenden Anwendung im industriellen Kontext des Systems Engineering in der Luftfahrt genauer untersucht, weshalb die Diagramme-Darstellung bevorzugt in der SysML-Ansicht erfolgt. Das Klassifikationsschema der einzelnen Typen sowie die typischen in der SysML Modellierung verwendeten Akronyme sind in Abbildung 3.13 dargestellt. Unterteilt werden drei Kategorien: Strukturdiagramme, Anforderungsdiagramme und Verhaltensdiagramme. Mit Hilfe von Strukturdiagrammen werden die Systembestandteile mit ihren Wechselwirkungen und Beziehungen dargestellt [Obj22]. Dazu gehören das von der UML 2.0 unverändert übernommen Paketdiagramm sowie die modifizierten Blockdefinitionsdiagramme und interne Blockdiagramme. Das Paketdiagramm wird vorwiegend zur Veranschaulichung der SysML Modellstruktur eingesetzt. Das Blockdefinitionsdiagramm zeigt die Komponenten des zu betrachtenden Systems und entspricht einer Abwandlung des im UML geläufigen Klassendiagramms, während im internen Blockdiagramm die innere Struktur der Systembestandteile dargestellt ist.

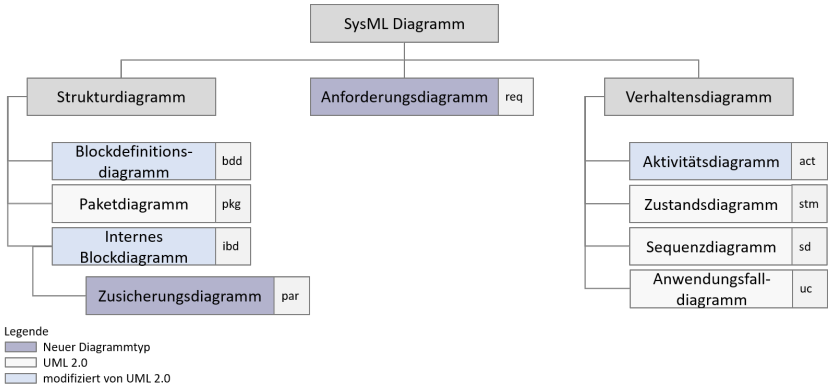



Abbildung 3.13: Klassifikationsschema der SysML und UML Diagramme.

In der verhaltensorientierten Betrachtung werden beispielsweise mit dem Aktivitätsdiagramm die systeminternen Funktionsabläufe und die Reihenfolge von Aktivitäten dargestellt. Es beschreibt eine bestimmte, festgelegte Reihenfolge von Aktivitäten, die auch zur Instanziierung von Objekten oder Datenstrukturen genutzt werden kann. Durch Objektflüsse können konkrete Instanzen erzeugt, verändert oder weiterverarbeitet werden [Fuc18]. Mit dem Zustandsdiagramm wird das Verhalten eines Systems spezifiziert und Bedingungen für Zustandsübergänge festgelegt. Zeitliche Abläufe von Interaktionen und Austausch von Nachrichten zwischen Systemelementen werden im Sequenzdiagramm beschrieben. Mit dem Anwendungsfalldiagramm werden Akteure und Anwendungsfälle mit ihren jeweiligen Beziehungen möglichst einfach dargestellt und veranschaulichen den Betrieb des Systems aus Sicht des Anwenders.

In der UML fehlen Diagramme für die Anforderungsmodellierung. Daher ist in der SysML das Anforderungsdiagramm hinzugekommen. Mit diesem werden die Systemanforderungen gegliedert und Zusammenhänge zu anderen Modellelementen aufgezeigt. Ein weiterer neuer Diagrammtyp ist das Parametrikdiagramm/Zusicherungsdiagramm, welches unterhalb des internen Blockdiagramms angeordnet ist. Mit diesem werden die mathematischen Zusammenhänge zwischen den Systemparametern abgebildet.

Einen vertiefenden Einblick in die Diagramme und zugehörige Anwendungsbeispiele gibt Anhang A.2. In diesem sind zudem die Knoten- und Verbindungselemente einzelner Diagramme genauer erläutert.

### 3.2.4 Methoden des Model-based Systems Engineering

Als dritter Bestandteil für die Erstellung eines Systemmodells ist eine Entwurfsmethode notwendig. Die Methode enthält die Definition aller relevanten Modelltypen und ihrer Beziehungen [BBJ+20]. Zusätzlich werden mit der Methode die Sichten (Views) auf das System festgelegt. Die Sichten ermöglichen die Darstellung spezieller Systemausschnitte, sodass das komplexe Gesamtmodell in kleinere und weniger komplexe Modelle zerlegt werden kann [Top22]. Weiterhin zeigt die Methode auf, wie spezifische Modelle generiert und anschließend analysiert werden können.

Mit dem MagicGrid® Book of Knowledge wurde ein Regelwerk für die Implementierung von MBSE speziell für die Implementierung mit Software von der Firma Dassault Systèmes geschaffen [AM21]. Dieses Regelwerk unterstützt bei der Modellierungstätigkeit und wächst mit dieser mit, indem es eine Zusammenstellung von bewährten Methoden bereitstellt, die je nach Industriezweig oder Kunde angepasst werden kann. Das Regelwerk ist dabei Tool-unabhängig, solange dieses die SysML unterstützt.

Ein Beispiel für eine MBSE-Methode ist das SPES Modeling Framework. Mit dieser Methode können eingebettete Systeme durchgängig modellbasiert entwickelt werden. Ziel ist dabei, die unterschiedlichen, aber in sich konsistenten Modelle desselben Systems disziplinübergreifend für unterschiedliche Abstraktionen zu erzeugen und anschließend miteinander zu verknüpfen [BBJ+20]. Drei weitere bekannte Methoden sind die OOSEM-Methode, FAS und SYSMOD. Die objektorientierte Systementwicklungsmethode (Object-Oriented Systems Engineering Method, OOSEM) ist ein modellbasierter Top-Down-Ansatz, der die Konzepte der Objektorientierung der SysML verwendet und überwiegend für die Neukonstruktion zur Anwendung kommt. Die Methode umfasst grundlegende SE-Aktivitäten wie Anforderungsanalyse und Validierung, aber auch Modellierungstechniken wie Variantenentwurf und Black-Box-Betrachtungen [FMS14]. Die FAS Methode (Functional Architecture for Systems Method, FAS) wird ebenfalls bei Neukonstruktionen angewendet. Sie beginnt die Modellierung mit Anwendungsfällen des geplanten Systems und leitet daraus eine funktionale Architektur ab [FGA+22]. Damit bietet die Methode eine lösungsneutrale Beschreibung des Systems als Grundlage für die Herleitung der physischen Systemarchitektur und Modularisierung des Systems [LW10]. Beim pragmatischen Systemmodellierungsprozess (Systems Modeling Toolbox, SYSMOD) baut die Entwicklung auf einer bestehenden technischen Ausgangslösung auf und findet vor allem in der Anpassungskonstruktion Anwendung. Ausgehend von der Basisarchitektur werden fortlaufend die Anforderungen und Architekturlösungen verfeinert (Zick-Zack Schema) [Wei08].

Ein umfassender Überblick und Vergleich der wichtigsten Methoden, die entwickelt und in der Industrie praktisch angewandt wurden, sind in der Literaturrecherche von Estefan und Weilkiens aufgeführt [EW23]. In der Praxis variieren MBSE-Ansätze stark, je nach Unternehmenskultur, Entwicklungsphase oder Detaillierungsgrad der Modellierung [FGA+22]. Eine große Herausforderung bleibt, die geeigneten Methoden für jeden Anwendungsfall auszuwählen, zu ergänzen und zu kombinieren.

### 3.2.5 Erweitertes V-Modell für das Model-based Systems Engineering

Die ursprüngliche Nutzung des Vorgehensmodells fokussierte einen dokumentenbasierten Prozess. Erweiterungen für das V-Modell sehen die Einbindung von Methoden aus dem modellbasierten Systementwurf als Unterstützung in der Analyse für den linken Flügel des „V“ vor. Insgesamt werden drei Modellierungsebenen (Abbildung 3.14) identifiziert. Ebene 1, die Modellierung und Spezifikation, auf der das System durch qualitative Modelle (deskriptiv und nicht simulierbar) beschrieben wird. Für die frühe Systembeschreibung werden Modellierungssprachen wie SysML empfohlen [EDA17]. In Ebene 2, die Modellierung und Simulation, werden quantitative Aspekte durch multidisziplinäre Simulationsmodelle integriert, die z.B. mit Matlab oder Modelica erstellt werden [Sen13, S. 90]. Ebene 3, die disziplinspezifische Modellierung, nutzt unter anderem CAx-Tools und enthält detaillierte Informationen und Aspekte für die konkrete Darstellung der Systemkomponenten, wie z.B. für die Geometriedarstellung. Parallel zu diesen drei sich überschneidenden Ebenen werden die Informationsartefakte bzw. Modellelemente in Anforderungen an das System (R = Requirements), Funktionen (F = Functions), logische Architekturelemente (L = Logical) und physische Systemteile (P = Physical Parts) unterschieden, die in Sprachen mit den Tools entlang der drei genannten Ebenen modelliert werden [EDA17].

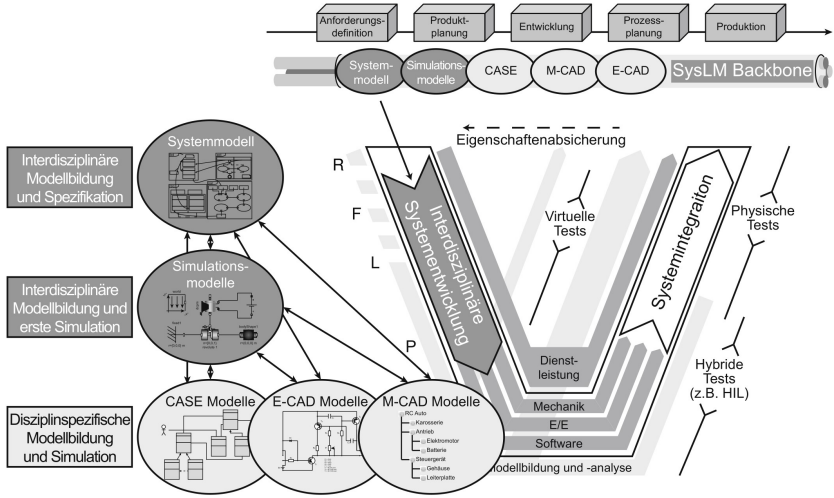



Abbildung 3.14: Erweitertes V-Modell für Model-based Systems Engineering administriert durch SysML, modifiziert von [ERZ14].

Zu Beginn der Entwicklung ist eine „vollständige Definition aller Anforderungen, Funktionen und logischen Systemelemente“ schwer zu erreichen [Sen13]. Im fortschreitenden Verlauf des Entwicklungsprozesses werden daher in inkrementellen Schleifen (Iterationen) alle Aspekte verfeinert, wodurch das Wissen über das Produkt zunimmt [Sen13].

**Rückverfolgbarkeit und Verknüpfung von Anforderungen**

Zwei wichtige Eigenschaften in Modellen sind Konsistenz und Rückverfolgbarkeit. Nach Madni und Sievers bedeutet Konsistenz, dass gemeinsame Annahmen und Definitionen auf konsistente Weise erkannt und beibehalten werden [MS18]. Rückverfolgbarkeit bedeutet, dass sich der Ursprung aller Beziehungen und Ergebnisse in einer spezifischen Anforderung, Norm oder Richtlinie wiederfinden lassen [MS18]. Diese Eigenschaften müssen idealerweise über die Systemgrenze eines Modells hinaus bestehen. Der Begriff Rückverfolgung wird in Anlehnung an die Definition der IEEE[6] für diese Arbeit folgendermaßen definiert:

**Definition 3.2.4 Rückverfolgung**

„[Rückverfolgung ist] der Grad, in dem eine Beziehung zwischen zwei oder mehreren Produkten des Entwicklungsprozess hergestellt werden kann, insbesondere zwischen Produkten, die eine Vorgänger-Nachfolger- oder Master-Unterordner-Beziehung zueinander haben [...]“ [IEE90, S. 78].

Dabei können zwei Einsatzgebiete unterschieden werden. Zum einen dient dies dem Aufbau einer vollständigen und konsistenten Darstellung von Produktinformationen. Beispielhaft ist hier der produktorientierte Modellansatz von Baïna et al. [BPM09] genannt. Zum anderen umfasst es die Überprüfung und Vernetzung von Anforderungen. Diese verfügen über diverse Schnittstellen und sind mit Systemkomponenten aus unterschiedlichen Bereichen verknüpft [Rup14, S. 375]. Die Darstellung von Anforderungen mit einem Modell ermöglicht, diese mit den Designdaten der Systemkomponenten, die ebenfalls als Modelle interpretiert werden, zu verlinken. Die Möglichkeit, Anforderungen zu beschreiben und während des gesamten Auslegungszyklus zu verfolgen, erhöht das Verständnis und Nachempfinden über den Änderungsgrund einer Systemkomponente im Entwicklungsprozess [Wol19]. Stark et al. [SBWF10] nutzen diese Verbindungen z.B. für ein proaktives Qualitätsmanagement und die Konformitätsprüfung. Anforderungsmängel haben einen vergleichsweise großen Anteil daran, wenn Leistungs- und Kostenziele nicht erreicht werden [DF84]. Dorfman [DF84] entwickelt eine Lösung, um mit Hilfe von Verlinkungen Objekte auf Einhaltung der Anforderungen zu überprüfen. Tekinerdogan und Erata [TE20] verwenden für die explizite Beziehungsdarstellung zwischen Systemelementen Anforderungsmodelle und zeichnen die Abhängigkeitsbeziehungen als zusätzliche Traceability-Links auf. Corsten [Cor06] nutzt einen zentralen Datensatz, um wiederum Anforderungen mit deren Dokumentation zu verknüpfen. Darauf aufbauend können automatisch Systemdokumente und Statusberichte über den Verifikationsfortschritt erstellt werden. Yoshikawa et al. [YHS09] verwenden ebenfalls Links für die Verbindung mit einer natürlichsprachlich geschriebenen Dokumentation, allerdings in Kombination mit Quellcode. Hierbei liegt der Nutzen in der Wiederverwendung und Wartung von Software. Für die automatisierte funktionelle Auslegung wird von Frezza et al. [FLC96] ein semantisches Graphendatenmodell verwendet, um damit bereits im frühen Designprozess das Verhalten entsprechend der Anforderungen zu simulieren.

[6] Institute of Electrical and Electronics Engineers ist ein weltweiter Berufsverband von Ingenieuren der Elektrotechnik und Informatik.

**Schlussfolgerungen**

Zur ganzheitlichen Betrachtung und Auslegung eines Systems werden mehrere Modelle und Modellierungsansätze benötigt, die in der jeweiligen Disziplin die entsprechenden Tools zur Beschreibung des Systems bereitstellen. Durch Verlinkungen zwischen den Modellen einzelner Disziplinen kann auf das vernetzte, formalisierte Wissen und auf die Änderungsauswirkungen übersichtlicher zugegriffen werden. Gleichzeitig stellen die vielen heterogenen Tools eine zentrale Hürde im MBSE dar. Die breite Toollandschaft führt zu verschiedenen Ansätzen in der Modellierung und zu Insellösungen [Pen23]. Einen einheitlichen Ansatz für die Integration heterogener Modelle gibt es nicht [Pen23]. Stattdessen müssen Lösungen und Standards gefunden werden, die eine Kopplung oder Integration der Informationen zwischen den diversen Programmen ermöglichen.

### 3.2.6 Einsatz von MBSE im Flugzeugvorentwurf

Um Innovationen zu beschleunigen und bei der Komplexität nicht die Übersicht zu verlieren, wird die Anwendung eines modellbasierten Prozesses im Flugzeugvorentwurf erprobt. Die Komplexität im Flugzeug entsteht nicht nur durch die Vielzahl verknüpfter physischer Komponenten, sondern auch durch die funktionalen Wechselbeziehungen und Abhängigkeiten zwischen den Systemen [LLL20]. Eine reine Nutzung von 3D Datensätzen ist nicht ausreichend, da diese die funktionalen Interaktionen des Systems nicht mit einbeziehen. Des Weiteren gibt es keinen alleinstehenden Modellierungsansatz, der alle Systemdisziplinen abbildet und Methoden zur Unterstützung der Entscheidungsfindung im Entwicklungsprozess bereitstellt. Eine Unterteilung der Systemmodellierung nach Aspekten mit unterschiedlichen Tools erhöht die Transparenz, Flexibilität und Adaption. Daher werden bereits im Entwurfsprozess von Flugzeugen und deren Kabinen modellbasierte Ansätze untersucht. Der Einsatz von MBSE ist im industriellen Flugzeugentwurf noch nicht etabliert. Die zentrale Herausforderung bei der Verwendung diverser Tools besteht auch im Flugzeugentwurf in der Kommunikation und der Anbindung dieser Tools untereinander – zusammengefasst unter dem Begriff Interoperabilität [LLL20].

Im Kontext von MBSE wird Interoperabilität häufig in unterschiedliche Ebenen unterteilt, wie etwa die syntaktische, semantische und organisatorische Integrationsebene [Est08]. Bei der semantischen Interoperabilität besteht die Herausforderung, eine eindeutige Interpretation der gemeinsamen Daten herzustellen, sodass sichergestellt wird, dass die Informationen zwischen Sender und Empfänger gleich verstanden werden [RM24]. Dabei bezieht sich die Interoperabilität auf die Fähigkeit Informationen zwischen mehreren digitalen Modellen miteinander auszutauschen und zu nutzen, sodass die Modelle miteinander interagieren können. Dies bedingt die Notwendigkeit einer organisatorischen Verwaltung in Form einer Infrastruktur für die Verteilung, das Auffinden, die gemeinsame Nutzung und den Austausch von Artefakten [SCC+23]. In dieser Arbeit wird Interoperabilität daher, angelehnt an die Definition von Weilkiens [Wei23], folgendermaßen definiert.

**Definition 3.2.5 Interoperabilität**

Interoperabilität im MBSE ist die Fähigkeit verschiedener Modellierungswerkzeuge und -umgebungen, Modelle und Daten ohne Informationsverlust auszutauschen und korrekt zu interpretieren, was eine kontinuierliche Systementwicklung über Disziplinen und Lebenszyklusphasen hinweg ermöglicht.

Im von der Europäischen Kommission geförderten Projekt AGILE 4.0 wurde der Flugzeugentwurfsprozess um Phasen des Systems Engineering erweitert, um weitere Entwurfsbereiche wie Fertigung, Wartung und Zertifizierung miteinzubeziehen [BCS+22]. Bussemaker et al. zeigen, dass mit Hilfe von modellbasierten Ansätzen im Rahmen von AGILE 4.0 ein Familienkonzept für Geschäftsflugzeuge entworfen werden konnte [BCS+22]. In diesem Zusammenschluss haben mehrere Einrichtungen und Unternehmen organisationsübergreifend zusammengearbeitet. Beginnend mit der Definition und Modellierung der Interessengruppen und deren Anforderungen wurde darauf aufbauend die funktionale Architektur und anschließend die logische Architektur modelliert (Abbildung 3.15). Dabei wird in der vorgelagerten Architektur eine Komponentenperspektive eingenommen, während im letzten Schritt eine disziplinäre Perspektive eingenommen wird. Die entwickelte Umgebung ermöglicht den Anschluss an Architektur-Optimierungsfunktionen und -abläufe für eine allumfassende Erkundung des gesamten Entwurfsraums [BC23].

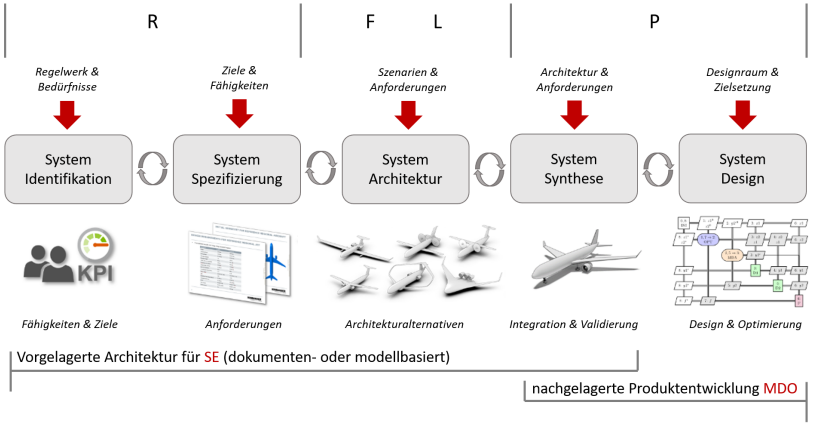



Abbildung 3.15: Der MBSE-Ansatz für die Entwicklung komplexer Luftfahrtsysteme (Flugzeuge) im EU-Luftfahrtforschungsprojekt AGILE 4.0, modifiziert von [BCS+22].

Guenov et al. untersuchten ebenfalls modellbasierte Ansätze für eine frühe Auslegung und Bewertung von Architekturen im Flugzeugentwurf. Am Beispiel der Klimakontrollsystemauslegung untersuchten sie zwei verschiedene Ansätze, um logische und rechnerische Disziplinen für die Modellierung und Simulation des Verhaltens eines Systems zu verknüpfen [GMCR+18]. Page Risueño und Nagel wiederum entwickelten ein wissensbasiertes Modell (Knowledge-based Engineering, KBE), das Produktionsprozesse (Kosten- und Leistungsabschätzung) in den Designprozess von Flugzeugen integriert [PN19] .

Ein weiteres Beispiel für den Einsatz von MBSE im Flugzeugentwurf zeigen Li et al. [LLL20]. Bei der Entwicklung und Prüfung von Softwareanwendungen für Flugzeugsysteme werden deren Testaktivitäten mit dem Entwicklungsprozess von Flugzeugsystemen im klassischen V-Modell verknüpft. Die Basis hierfür stellt die empfohlene Praxis für die Luft- und Raumfahrt (Aerospace Recommended Practice, ARP) ARP4754A. Diese Leitlinien, herausgegeben von der Society of Automotive Engineers (SAE), beschreiben die Entwicklung von Flugzeugsystemen unter Berücksichtigung der gesamten Betriebsumgebung und bieten Praktiken für den Nachweis der Produktsicherheit mit dem Ziel der Zertifizierung [Int23]. Abbildung 3.16 zeigt das V-Modell für den genannten Zusammenhang für die Flugzeugsystemintegration.

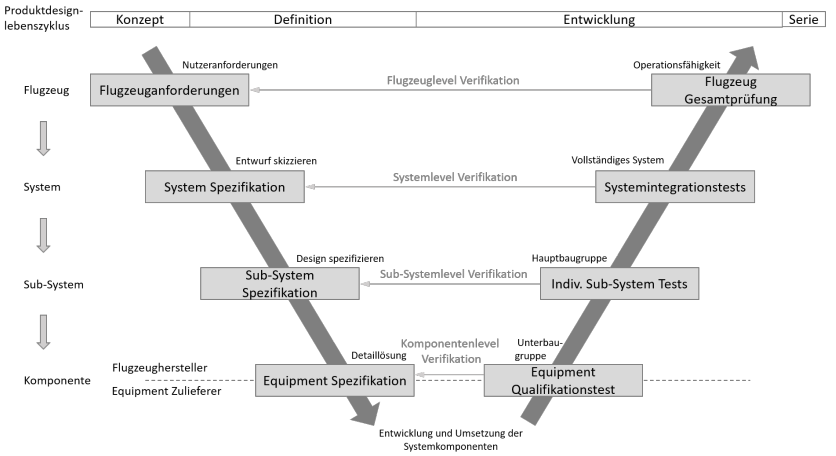



Abbildung 3.16: V-Modell zur Flugzeugsystemintegration aus Sicht des Produktdesignlebenszyklus nach [LLL20].

In der Praxis zeigt sich, dass die Philosophie des RFLP-Ansatzes nicht ganzheitlich gelebt wird. Stattdessen wird in Projekten nur ein Teil der Abstraktionsebenen bedient. So wird häufig das *L* übersprungen und die Funktionen direkt mit den physikalischen Systemarchitekturelementen verknüpft. Dies führt zu einer übergeordneten Sicht auf die Systemgestaltung mit dem Bild „erst die Struktur, dann das System“ [LLL20]. Zudem ist das Erstellen von Verknüpfungen im RFLP zwischen den einzelnen Elementen zeitaufwendig, da viele heterogene Softwareumgebungen genutzt werden und der Initialaufwand für die verschiedenen Interaktionsinformationen durch manuelle Modellierung hoch ist [LLL20]. Die Effizienz der Modellierung könnte allerdings durch Vorlagen mit vordefinierten Interaktionen erhöht werden. Basis für diese Erstellung und Pflege der Verknüpfungen könnte der Digitale Faden sein [LLL20].

Durch die automatisierte Modellentwicklung und -anpassung wird die Untersuchung von großen und komplexen interdisziplinären Systemen erleichtert. Die beobachteten Trends zeigen, dass formale Modelle und Ontologien in Zukunft eine bedeutende Rolle spielen werden, da die Interoperabilität immer mehr in den Vordergrund rückt und die neue Generation von verknüpfbaren Datenquellen auf semantisch angereicherten Informationen basieren sollte [JSH+20, S. 16]. Dennoch hat sich die Nut-

zung der SysML noch nicht als Standard etabliert. Das Ergebnis einer Befragung mit Vertretern der deutschen Wissenschaft und Wirtschaft 2020 hat ergeben, dass aus dem MBSE nur die formale Modellierung von Systemarchitekturen zur Anwendung kommt [DAG+21]. Die volle Ausschöpfung der Leistungspotenziale werden nicht genutzt. Gründe hierfür sind die erschwerte Integration in bestehende IT-Infrastrukturen, fehlende Austauschformate, einheitliche Methoden-Standards und der damit verbundene Mehraufwand.

### 3.2.7 Einschränkungen und Grenzen durch Entwicklungsumgebungen

Mit der SysML alleine ist es nicht möglich, Eigenschaften wie Verfügbarkeit oder Komfort zu überprüfen [KNK+20]. Stattdessen ist die Integration von externen Simulationstools notwendig. Dabei bieten Toolumgebungen bereits vordefinierte Schnittstellen zu anderen Entwicklungsumgebungen an. Allerdings werden nicht alle Austauschformate unterstützt, es gibt keinen Datenaustauschstandard und eine zusätzliche Schnittstellenprogrammierung als Individuallösung ist meistens notwendig. Eine beispielhafte Aufgliederung nach Systemlevel und angewandten Toolumgebungen ist in Abbildung 3.17 dargestellt. Für die semi-formale Beschreibung eines Systems kommt die standardisierte Modellierungssprache SysML zu Beginn des Entwicklungsprozesses zum Einsatz, während im weiteren Verlauf formale Modelle (z.B. CAD und CAE) auf Layout und Detailebene genutzt werden. Husung et al. leiteten daraus unter anderem die Frage ab, welche Informationen zwischen den verschiedenen Modellen und damit auch zwischen den Systemebenen für die disziplinübergreifende Entwicklung eines Systems ausgetauscht werden müssen [HWM22]. Die Art und der Umfang der Information sind entscheidend. Die Abbildung geometrischer Informationen mit der SysML ist zeitaufwendig und nicht zielführend. Daher muss eine Kopplung der verschiedenen Entwicklungsumgebungen für die holistische Betrachtung und Auslegung nach dem V-Modell stattfinden und die Abhängigkeiten zwischen den Elementen in jedem Systemlevel vollständig verstanden und abgebildet werden.

Zusammenfassend wird MBSE in der Produktentwicklung eingesetzt, um komplexe technische Systeme strukturiert zu entwerfen, zu analysieren und zu verwalten, indem die traditionelle dokumentenzentrierte Entwicklung durch digitale Modelle ersetzt wird. Dennoch bringt dieser modellbasierte Ansatz einige Herausforderungen, besonders im Datenaustausch zwischen den Modellen [SS20], mit sich:

- Einschränkung der Modellinteroperabilität und der Tool-Schnittstellen,
- Bereitstellung heterogener Daten durch die verschiedenen Disziplinen,
- Abwägung zwischen open-source und kommerziellen Tools,
- Wartung und Modellverständnis als Nicht-Experte,
- Zugänglichkeit und Benutzerakzeptanz,
- Wiederverwendbarkeit von formalisiertem Wissen und Modellen (keine Silos),
- Unterschiede im Vokabular und Begriffsverständnis.

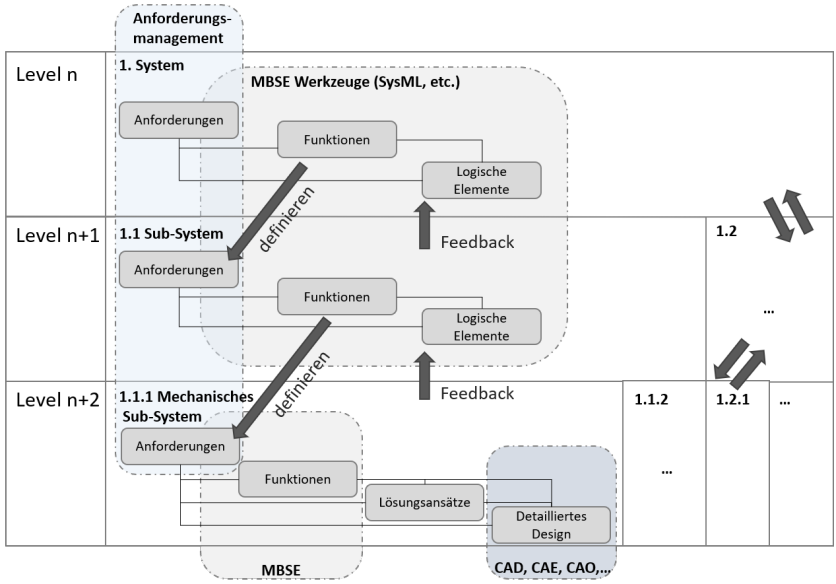



Abbildung 3.17: Systemlevel und angewandte Toolumgebungen nach [HWM22].

Wie bereits in den Abschnitten 3.1.3 und 3.2.6 aufgezeigt, können Anforderungen nicht nur mit einem Tool validiert und verifiziert werden. Stattdessen werden verschiedene Entwicklungsumgebungen und Methoden benötigt. Vor allem die unterschiedlichen Methoden erschweren die Interoperabilität zwischen den Disziplinen. Ein Lösungsansatz zur Kopplung der heterogenen Disziplinen ist eine gemeinsame Ontologie, die eine konsistente und einheitliche Grundlage für die Darstellung und Analyse bietet [MS18]. Madni und Sievers definieren Ontologie als:

**Definition 3.2.6 Ontologie**

„Eine Ontologie ist eine formale, explizite Konzeptualisierung einer Problemdomäne, die von allen Beteiligten geteilt wird; sie stellt ein kontrolliertes Vokabular dar, das eine Reihe von vereinbarten Begriffen (semantische Domäne) und Regeln für die Verwendung und Interpretation dieser Begriffe innerhalb der Domäne umfasst.“ [MS18].

Dabei werden Begriffe im semantischen Bereich beschrieben und deren Beziehungen sowie Regeln für die Verwendung dieser aufgezeigt. Eine Kontrollinstanz verwaltet diese expliziten, eindeutigen und nicht redundanten Begriffe und Beziehungen. In jeder Disziplin werden dann die identischen Begriffe und deren Verbindungen verwendet, sodass eine eindeutige Identifikation in jeder Disziplin möglich ist. Terminologien und Ontologien für Daten und Modelle verbessern somit die Zugänglichkeit der Daten [MH23]. Drei Hauptanforderungen, die dennoch resultieren und zu lösen sind, sind laut Wilking et al. Konsistenz, eine allgemeingültige Modellierungsstrategie und Zielorientierung [WSSW22].

**Konsistenz**
Sowohl SysML als auch UML erlauben eine flexible Semantik. Diese Flexibilität kann zu Mehrdeutigkeit im Modell führen, sodass entweder die Bedeutung der Diagramme oder das Verständnis der Funktionsweise des Modells nicht eindeutig ist [MS18]. Die Interpretation und das Modellverständnis sind teilweise abhängig vom Entwickler des Modells. Daher müssen Verbindungen und Abhängigkeiten im Modell so eindeutig modelliert werden, dass eine klare Interpretation und externe Verknüpfung zu anderen Disziplinen möglich ist, ohne zu viel Raum für menschliche Interpretation zu lassen.

**Modellierungsstrategie**
Wie bereits in Abschnitt 3.2.6 aufgezeigt, werden unterschiedliche Strategien und Methoden zur Modellierung der einzelnen Aspekte eines Systems angewendet. Zusätzlich werden Systemmodelle von externen Unternehmen in der Zulieferkette bereitgestellt [WSSW22]. Eine Zusammenführung dieser unterschiedlichen Modelle für eine holistische Betrachtung des Gesamtsystems verursacht mehrere Iterationen und ist mit einem höheren Managementaufwand verbunden. Daher ist eine einheitliche Modellierungsstrategie notwendig. Mit dieser werden die Modelle im Unternehmen und in der gesamten Lieferkette in gleicher Weise erstellt und spätere Iterationen verringert [WSSW22].

**Zielorientierung**
Mit der Einführung der Modellierungssprache SysML wurde ein Hilfsmittel geschaffen, das durch die Nutzung grafischer Elemente die Analyse und Evaluation von Systemen sowie die Modellierung ihrer Anforderungen ermöglicht, indem es verschiedene Sichten auf ein System bereitstellt. Andererseits ermöglichen die Tools eine verständliche Kommunikation von Systeminformationen zwischen den verschiedenen Stakeholdern. Wilking et al. empfehlen vor Beginn der Modellierung den Zweck festzulegen, wofür die Diagramme genutzt werden sollen. Während einige Diagramme vorwiegend zu Kommunikationszwecken geschaffen werden und dabei helfen, die anderen Diagramme zu verstehen, sind letztere auf eine maschinelle Erkennung ausgelegt [WSSW22]. Eine Vermischung der beiden Diagramme soll daher vermieden werden, damit Informationen und Informationsflüsse vollständig und konsistent gehalten werden können.

Gausemeier et al. und Wilking et al. stellen die Hypothese auf, dass der Einsatz von SysML Modellen die Komplexitätsbeherrschung fördert und damit langfristig die Produktqualität steigert [GDS+13, WSSW22]. SysML Modelle können einen Überblick über die unterschiedlichen Verhaltensweise in der Entwurfsphase gewährleisten, im weiteren Verlauf bei der korrekten Modellierung des Systemverhalten unterstützen und eine direkte Verifikation ermöglichen [WSSW22].

### 3.2.8 Schlussfolgerungen

Beim Einsatz von dokumentenbasierten Ansätzen im Systems Engineering für die Entwicklung von technischen Systemen werden Dokumente zur Kommunikation und als zentrale Informationsquelle für alle Projektbeteiligten genutzt. Mit der zunehmenden Komplexität dieser Systeme bringen solche dokumentenzentrierten Ansätze jedoch zunehmend Herausforderungen mit sich. Unter anderem werden sie anfällig für Fehlinterpretationen [Kan21]. Hinzu kommt, dass Anforderungen oder Prozessschritte nicht zurückverfolgt und damit nachvollziehbar verstanden werden können. Darüber

hinaus können die Dokumente nur bedingt wiederverwendet werden [Kan21]. Hierbei schaffen die Formalisierung von Wissen und die Darstellung in Form von Modellen Abhilfe, indem sie eine vereinfachte Realität abbilden. Das Ziel von MBSE ist es, die Defizite wie Inkonsistenzen durch unzusammenhängende Dokumente der traditionellen, dokumentenzentrierten SE-Ansätze zu überwinden. Mit einer Sammlung zusammenhängender und integrierter Modelle kann generell eine einzige, allgemein zugängliche Datenquelle (single source of truth) geschaffen werden. Wie beim PLM wird eine zentralisierte Daten- und Informationsverwaltung für die Zusammenarbeit über alle Disziplinen hinweg angestrebt. Zudem wurde MBSE eingeführt, um die Kommunikation in interdisziplinären Teams zu verbessern und Fehler früher im Entwicklungsprozess zu detektieren. Die Rule of ten besagt, dass die Kosten zur Behebung eines Fehlers mit jedem Schritt im Entwicklungsprozess exponentiell steigen, sodass Fehler, die erst spät im Prozess erkannt werden, deutlich teurere Änderungen erfordern als in den frühen Phasen, wie der Anforderungsanalyse.

Durch die Anwendung von MBSE wird eine Beherrschung der Komplexität in der Entwicklung von Produkten versprochen. Die Allgemeine Modelltheorie von Stachowiak beschreibt bereits 1973, dass Modelle ein Abbild bzw. eine Abstraktion eines zu repräsentierenden Originals sind, um mit der Vielschichtigkeit der Realität umgehen zu können [Sta73]. Durch die Modellbildung wird eine Reduzierung auf jene Attribute gelegt, die dem Modellerschaffer für den Modellzweck relevant erscheinen [Sta73]. Dabei entstehen nicht zwangsläufig einfache Modelle; insbesondere komplexe Systeme bedingen oftmals Modelle mit entsprechender Komplexität. Die Herausforderung dabei besteht darin, die Realität in seiner Komplexität soweit akkurat abzubilden und zu beschreiben, dass der Modellzweck erreicht wird. In der Praxis zeigt sich, dass dies zu immer größeren, umfangreicheren Einzelmodellen führt, in denen die interne Modellkomplexität schnell steigt. Letzteres wird zusätzlich durch unterstützende Elemente im Modell erhöht, die nicht dem realen System zugeordnet werden können. Diese Elemente ermöglichen aber eine gesteigerte Funktionalität des Modells. Durch Gegenmaßnahmen, wie eine Anpassung der Viewpoints und damit die Möglichkeit verschiedener Ansichten pro Anwender auf das Modell, werden zwar die Modelle einzeln übersichtlicher, aber die Größe und Komplexität des Gesamtmodells steigen dennoch. Weiterhin müssen auch Nicht-Experten Modelle und Beziehungen untereinander verstehen können, um Feedback zu geben.

Eine weitere Herausforderung liegt in der breiten vorliegenden Toollandschaft. Bei der Modellierung von Architekturen oder der qualitativen Analyse eines Systems wird die SysML verwendet. Für umfangreichere Berechnungen oder geometrische Untersuchungen müssen hingegen analytische Modelle genutzt werden. Beispiele aus Abschnitt 3.2.6 zeigen den Einsatz von modellbasierten Ansätzen im Flugzeugentwurf. Diese Ansätze sind nur teilweise mit anderen Disziplinen gekoppelt, oder eine Kopplung und Integration der verschiedenen Modelle ist nicht umgesetzt. Des Weiteren beschreiben die Modelle jeweils einen Teilaspekt und verwenden dabei eine eigene Ontologie. Damit eine konsistente Repräsentationsgrundlage und eine semantische Interoperabilität geschaffen werden kann, sollten aber alle Modelle die gleiche semantische Basis verwenden, indem eine Ontologie die gemeinsamen Begriffe und Beziehungen standardisiert.

## 3.3 Zusammenfassung und Problemanalyse

Für eine ganzheitliche Betrachtung und Auslegung eines technischen Systems werden im Entwurfsprozess mehrere Disziplinen miteinander gekoppelt. Eine frühzeitige Vernetzung aller Disziplinen im Entwurf verbessert die Qualität des Produktes und seine Entwicklungszeit. Je mehr Disziplinen dabei für die Entwicklung eines neuen Produkts benötigt werden, desto mehr Heterogenität liegt in der Abstraktion der notwendigen Modelle vor, die das Verhalten des Produktes vorhersagen oder bewerten. Dabei variieren das notwendige Abstraktionslevel und der Detaillierungsgrad der disziplinären Modelle stark. Folglich werden unterschiedliche Methoden und damit auch (digitale) Tools für den Aufbau der Modelle benötigt. Für eine Analyse und Bewertung des technischen Gesamtsystems müssen diese wiederum miteinander verbunden werden. Dabei können die Tools nicht immer direkt miteinander kommunizieren und verfügen nur bedingt über standardisierte Schnittstellen. Das Resultat sind eine Vielzahl an unterschiedlichen Schnittstellen, wodurch die Komplexität des multidisziplinären Gesamtmodells für die Auslegung und Analyse eines technischen Systems steigt. Gleichzeitig steigt der Bedarf des Kunden an Individualisierung und der Bereitstellung von Produktvarianten, die wiederum Einfluss auf die Modellierung nehmen. Für die Variantenvielfalt müssen, getrieben durch Innovationen oder neuartige Technologien, neue Features oder Subsysteme laufend in die Modelle integriert werden. Folglich resultiert die im Rahmen dieser Arbeit zugrundeliegende Forschungsfrage:

> *Wie kann methodisch eine Modularität des Gesamtsystems ermöglicht werden, damit Varianten eines komplexen technischen Systems abgebildet und für die multidisziplinäre Analyse des Systemverhaltens kollaborativ modelliert werden können?*

Wie bereits in Abschnitt 3.2.7 erwähnt, wurde sich mit der Einführung von MBSE eine Beherrschung der Komplexität versprochen. In der Anwendung der Modellierungssprache *Systems Modeling Language* zeigen sich in der Literaturrecherche jedoch einige Herausforderungen. Die folgenden Anforderungen **A.1** bis **A.3.** beziehen sich daher vor allem auf die Modellierung mit der SysML. Bei der Modellierung werden neben kontextspezifischen Elementen auch weitere unterstützende Elemente eingebunden, die keinen Bezug zum realen System haben. Verstärkt wird die daraus resultierende induzierte Modellkomplexität durch die Abbildung aller potentiellen Varianten in einem SysML-Modell. Aufgrund der zunehmenden Verflechtungen steigt der Mehraufwand für die Pflege der Modelle. Darüber hinaus wird das Systemverständnis für Nicht-Experten erschwert. Hieraus leitet sich die nachfolgende Anforderung an die Modellierung mit der SysML ab.

**A.1** **Induzierte Modellkomplexität** - Komplexe Systeme führen zu komplexen Modellen. Dabei muss der Einfluss auf die induzierte Modellkomplexität durch Zusatzelemente oder die Variantenabbildung begrenzt werden, sodass das Modell für den Menschen noch interpretierbar und bearbeitbar bleibt.

Ein weiterer Aspekt ist die vorherrschende Variantenvielfalt. Im Rahmen des Customizings werden Modifikationen oder Nachrüstungen an der Kabinenausstattung vorgenommen. Zudem entwickeln sich die Subsysteme der Kabine durch Innovationen oder

neue Technologiemodelle stetig weiter. Daraus leitet sich die folgende Anforderung an die Flexibilität einer Modellierungsmethode ab.

**A.2** **Erweiterbarkeit der Modelle** - Die Methode erlaubt Modularität, sodass eine Erweiterung und Modifikation der Modelle um neue Varianten realisierbar ist. Somit wird die Integration neuer Technologien in das bestehende Gesamtsystemmodell ermöglicht.

Abgeleitet aus Anforderung **A.2** ergibt sich ein Bedarf an eine Wiederverwendbarkeit der Modelle. Dadurch kann bei zukünftigen Produktentwicklungen die Entwicklungszeit beschleunigt werden, indem die bereits erbrachte Entwicklungsleistung im Modell wiederverwendet wird. Besonders in der Variantenentwicklung ist dies vorteilhaft, da einige Subsysteme wiederkehrend in unterschiedlichen Varianten Anwendung finden. Die zugehörige Anforderung ist im Folgenden aufgeführt.

**A.3** **Wiederverwendbarkeit der Modelle** - Die Methode ermöglicht die Wiederverwendung von Modellen, ohne umfangreiche Anpassungen an diesen vorzunehmen, um je nach Zusammensetzung unterschiedliche Varianten eines Gesamtsystems erzeugen zu können.

Dabei muss die Methodik auch den kollaborativen Charakter heutiger Entwicklungsprozesse einbeziehen. In der Industrie sind Entwicklungsprozesse zunehmend interdisziplinär und teamorientiert organisiert. Zudem kommen unterschiedlichste Modellierungsumgebungen zum Einsatz. Ausgehend von der oben genannten Forschungsfrage zur kollaborativen Entwicklung von Varianten ergibt sich als ergänzende Forschungsfrage, eine Methodik zu entwickeln, die im industriellen Kontext praktisch anwendbar ist.

> *Wie kann eine praxisorientierte Methodik zur kollaborativen Entwicklung von Varianten gestaltet werden, die im industriellen Kontext anwendbar ist und Komplexität sowie Aufwand-Nutzen-Verhältnis angemessen berücksichtigt?*

Für die Beantwortung leiten sich die nachfolgenden Anforderungen an eine Modellierungsmethode für die flexible Zusammenarbeit zwischen Disziplinexperten und ihren in der Detailtiefe variierenden Modelle ab. Die Anforderungen **A.4** bis **A.5** beziehen sich dabei auf den allgemeinen Kontext des MBSE.

**A.4** **Multidisziplinäre Zusammenarbeit** - Für die Modellierung eines multidisziplinären Gesamtmodells werden mehrere Experten benötigt, die jeweils eigene Methoden und Modelle für die Beantwortung ihrer disziplinspezifischen Fragestellungen bereitstellen. Für die ganzheitliche Systembetrachtung und die Vermeidung von Datensilos ist daher eine Modellkopplung bzw. ein durchgängiger Datentransfer zwischen den verschiedenen Modelldisziplinen notwendig.

Neben der Zusammenarbeit innerhalb eines Unternehmens, müssen auch Experten und dessen Modelle über die Unternehmensgrenze hinweg eingebunden werden. Besonders in der Flugzeugkabinenentwicklung sind eine Vielzahl an Lieferanten für die Auslegung der Kabinenkomponenten beteiligt, sodass eine enge Verflechtung mit externen Experten in der Betrachtung von Kabinenkonfigurationen besteht. Der Datenaustausch erfolgt überwiegend dokumentenbasiert. Dabei müssen rechtliche oder IT-spezifische Einschränkungen berücksichtigt werden. Zudem kommen kommerzielle

als auch open-source Tools zum Einsatz, die unterschiedliche Schnittstellenformate bereitstellen. Daraus ergibt sich die folgende Anforderung an eine flexible Modellierungsmethode zur Zusammenarbeit im Entwicklungsprozess.

**A.5** **Unternehmensübergreifende Zusammenarbeit** - Die Entwicklung und Auslegung der Flugzeugkabine ist stark geprägt durch die Zusammenarbeit mit einer Vielzahl an Zulieferern. Somit ergibt sich der Bedarf an eine automatisierte und durchgängige Kopplung bzw. Integration der Modelle oder deren Daten von externen Partnern im Produktentwicklungsprozess.

Zur erfolgreichen Beantwortung der zugrundeliegenden Forschungsfrage dieser Arbeit ist die Erfüllung der Anforderungen **A.1** bis **A.5** notwendig.

# 4. Aktuelle Entwicklungen in der modellbasierten Variantenbildung

> *„Die explizite und durchgängige Modellierung der Varianz aller Entwicklungsartefakte wird in aktuellen Ansätzen der modellbasierten Systementwicklung sowie in Modellierungssprachen wie der Systems Modeling Language (SysML) nicht hinreichend unterstützt." [MWR+22, S. 4]*

Mit der Digitalisierung der Prozesse im Flugzeugentwurf ergeben sich neue Möglichkeiten für Unternehmen. Prozesse können automatisiert, Maschinen/Roboter durch die Integration von Informationssystemen gesteuert und neue Technologien schneller integriert werden [PV22]. Zudem ermöglicht dies eine Untersuchung von Produktanpassungen und die vielfältige Auslegung eines Systems speziell nach Kundenwünschen. Varianten erhöhen die Zufriedenheit bei den Kunden, unterstützen die Produktion individueller Lösungen und erweitern das Produktportfolio eines Unternehmens. Gleichzeitig steigt damit auch die interne Systemkomplexität. Für die Auslegung von Varianten wird in der Entwicklungs- und Produktionskette eine hohe Flexibilität gefordert.

Um die offene Forschungsfrage aus Abschnitt 3.3 zu beantworten, wird eine Methode zur Variantenmodellierung benötigt, die die Anforderungen **A.1** bis **A.5** erfüllt. Daher werden in diesem Kapitel bereits entwickelte methodische Lösungsansätze im MBSE diskutiert. Dieses Kapitel bildet einen weiteren Teil der Descriptive Study I der DRM, indem der Stand der Technik für die Variantenmodellierung analysiert wird, um daraus den Bedarf an einer neuen Methode zu konkretisieren. Begonnen wird mit einer Einführung und Definition des Begriffs Variante. Anschließend wird das Konzept der Produktfamilie im Flugzeugentwurf vorgestellt und ein Einblick in die individuelle Produktanpassung im Rahmen des Customizings für die Flugzeugkabine gegeben. Zum Schluss werden Methoden für die Modellierung von Varianten vorgestellt. Dabei werden zum einen die Ansätze der Variantenmodellierung in der rechnerunterstützten Konstruktion und zum anderen die Ansätze in der SysML näher beleuchtet. Gleichzeitig werden die Grenzen der Ansätze aufgezeigt und unter Betrachtung der Anforderungen **A.1** bis **A.3** bewertet. Zudem findet eine Abgrenzung zum Konzept der Modularisierung und dem Baukastenprinzip statt.

## 4.1 Einführung und Definition von Varianten

Die Industrie sieht sich in den letzten Jahren vermehrt mit der Herausforderung konfrontiert, eine große Anzahl von Produktvarianten anzubieten [Wei16]. Laut Weilkiens

entwickelt sich die Industrie in Richtung Massenanpassung statt Massenproduktion [Wei16]. Weiterhin stellen Jiao und Chen heraus, dass Unternehmen mehr Fokus auf die Kundenbedürfnisse legen sollten, da die Produktzufriedenheit mit der Funktionalität des Produkts an sich und einer angemessenen Anzahl an Wahlmöglichkeiten steigt [JC06]. Das Resultat ist ein breiteres Spektrum an Produktvarianten. Der Duden erläutert den Begriff Variante mit „Abwandlung“ und „leicht veränderte Art [oder] Form von etwas“ [Cor23]. Änderungen können dabei technischer oder ästhetischer Natur sein. Eine Variante von etwas weist eine hohe Anzahl identischer Gruppen oder Teile auf und umfasst Bestandteile, die eine ähnliche Funktion und/oder Form aufweisen [Ren07]. In dieser Arbeit wird für den Begriff Variante die Definition von Renner verwendet [Ren07].

**Definition 4.1.1 Variante**

„Varianten sind Gegenstände mit einem in der Regel hohen Anteil identischer Komponenten, die Ähnlichkeiten in Bezug auf mindestens eines der Merkmale Geometrie, Material oder Technologie aufweisen.“ [Ren07]

Mehrere Varianten einer Komponente werden auch als Feature bezeichnet [Wei16]. In der Luftfahrt wird den individuellen Anforderungen des Kunden durch eine breite Auswahl an Features begegnet. Dabei erhöht jede zusätzliche Variation die Anzahl potentieller Kombinationen. Abbildung 4.1 zeigt, wie mit beispielhaften Features für die Kabinenkomponenten Sitz (Economy- oder Businessklasse) und Gepäckablage unterschiedliche Kabinenvarianten erstellt werden können. Insgesamt können aus den beiden Features 16 Varianten erzeugt werden.

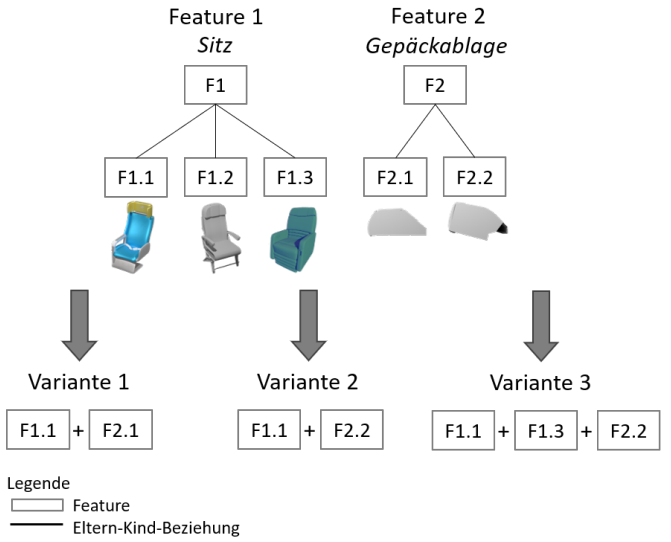



Abbildung 4.1: Kombination mehrerer Features von Kabinenkomponenten zur Bildung von Kabinenvarianten. Die Anzahl der Varianten für das Minimalbeispiel beträgt 16.

Im Entwurf von Flugzeugen, beispielsweise bei Produktfamilien, kommt das Konzept der Modulbauweise zum Einsatz. Dabei wird ein System in unabhängige Teile, sogenannte Module, zerlegt [BBH+18]. Diese abgeschlossenen Einheiten erfüllen jeweils eine bestimmte Funktion, wobei mehrere flexible Module die gleiche Funktion anbieten. Zudem hat ein Modul klar definierte und eindeutige Schnittstellen. Abbildung 4.2 zeigt die Modulbauweise beispielhaft für das Flugzeugkabinensystem Bordküche (Galley). Für die Subsysteme Ofen, Stauraum und Trolley gibt es je nach Hersteller unterschiedliche Modulvarianten. Erst der Zusammenschluss einzelner Module zu einer Einheit ermöglicht die Ausführung einer bestimmten Gesamtfunktion. In dem gezeigten Beispiel ist dies die Bewirtung der Passagiere durch Aufbewahren der Lebensmittel und Erwärmen der Gerichte. Hervorzuheben ist, dass bei dem Konzept der Modulbauweise ein festes Set an Kernmodulen existiert, die nicht verändert werden dürfen. Die Produktarchitektur beschreibt wie die physischen Einheiten angeordnet sind und zusammenwirken, um die geforderten Funktionen zu implementieren [BBH+18]. Aufgrund der Austauschbarkeit der Module untereinander können viele Varianten technischer Systeme zusammengestellt und das Systemprodukt entsprechend der Kundenbedürfnisse angepasst werden. Dennoch ist der Personalisierungsprozess sehr komplex und bringt für den Bereich Kabine diverse Abhängigkeiten mit sich.

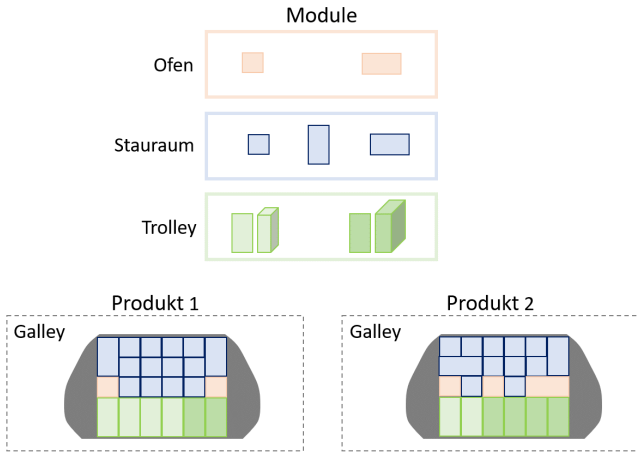



Abbildung 4.2: Beispielhafte Darstellung der Modulbauweise für das Kabinensystem Bordküche (Galley) adaptiert von [HSS+21].

Unternehmen verwalten ihre Varianten häufig in einem Systemmodell, mit dem eine Abbildung der Produktlinie oder kundenspezifischer Produkte möglich ist. Mit zunehmender Vielfalt an Komponenten bzw. Features wird das Management und der Umgang mit Daten sowie die Verwaltung interner Abhängigkeiten immer wichtiger und anspruchsvoller [HSS+21]. Neben der Herausforderung, wie Varianten modelliert werden können, spielt die Speicherung von Varianten und deren Daten eine ebenfalls wichtige Rolle. Aufbauend auf den historischen Daten können zukünftig Empfehlungssysteme Produktportfolios wettbewerbsfähig machen, indem datengetriebene Empfehlungen ausgesprochen werden können [SMM22]. Diese Eigenschaft ist allerdings

abhängig von historischen Daten und ist limitiert, sobald neue Merkmale/Features in Produkte implementiert werden, da für diese keine Datensätze vorliegen. Zudem ist dies mit einem hohen Initialaufwand und einer hohen Rechenleistung verknüpft, da eine große Datenmenge verarbeitet werden muss.

## 4.2 Produktfamilien im Flugzeugentwurf

Während früher einzelne Produkte entwickelt wurden, verschiebt sich heutzutage die Produktentwicklung in Richtung von Produktfamilien. Eine Familie besteht aus mehreren, ähnlich aufgebauten Produktvarianten, die sich in einzelnen Produktmerkmalen wie Funktion oder Baugröße unterscheiden [LHK21].

In der Luftfahrtindustrie wird mit einer Standard-Spezifikation begonnen. Ausgehend von dieser können Airlines Änderungen oder Ergänzungen vornehmen, die durch vorqualifizierte Auswahlmöglichkeiten aus einem Variantenpool bereitgestellt werden [Ack13]. Damit können individuelle Anforderungen und Wünsche von Kunden gezielter berücksichtigt werden. Ein Beispiel für eine Produktfamilie stellt die A320-Familie für Mittelstreckenflugzeuge dar. Diese Baureihe des Flugzeugherstellers Airbus umfasst vier Schmalrumpfflugzeuge, wobei die A320 das Basismodell darstellt. Die Modelle unterscheiden sich in der Länge des Rumpfes; die A318 und A319 sind die kürzeren Varianten während die A321 eine gestreckte Version zeigt. Veranschaulicht sind die vier Flugzeugmodelle von Airbus in Abbildung 4.3.


Abbildung 4.3: Produktfamilie der Mittelstreckenflugzeuge A3XX von Airbus[7].

Ziel ist die zukünftige Entwicklung von Flugzeugmustern nach einem Baukastenprinzip, bei dem viele Grundkomponenten gleich aufgebaut sind und je nach Wunsch

[7] https://aircraft.airbus.com/en/aircraft/a320-the-most-successful-aircraft-family-ever

des Kunden unterschiedliche Module hinzugefügt oder weggelassen werden können. Konfigurierte Module für Subsysteme und/oder Komponenten werden zu einem Produkt zusammengeführt, wobei der Skaleneffekt bei der Wiederverwendbarkeit bereits geleisteter Arbeit genutzt wird [SSK19]. Das Prinzip wird bislang im Ansatz bei der Entwicklung der A350-Baureihe angewandt. Bei älteren Flugzeugmustern, wie der A320-Familie, sind Änderungswünsche nur mit einem erheblichen Entwicklungsaufwand möglich. Darüber hinaus verfolgen Forschungseinrichtungen wie das DLR eine technologieoffene Strategie in der Luftfahrt zur Erforschung von evolutionären und revolutionären Technologien, um disruptive Änderungen am Konzept untersuchen und bewerten zu können [Deu21b]. Die Ziele sind eine Effizienzsteigerung sowie die Erreichung einer klimaneutralen Luftfahrt [Deu21b].

Jedoch steigt durch die Varianten die Komplexität der Produktentwicklung. Das Verwalten der Daten nimmt eine wichtigere Rolle ein. Neben den unterschiedlichen Anforderungen und zusätzlichen geometrischen Informationen, müssen auch die Wechselwirkungen untereinander für jede Variante abgebildet werden können [SSK19]. In der Flugzeugindustrie entstehen Varianten erst, wenn diese vom Kunden gewünscht werden oder durch Nachrüstungen. Deshalb sollte der Produktionsentwicklungsprozess mit seinen Modellen so gestaltet sein, dass Varianten auch im Nachhinein erzeugt werden können.

## 4.3 Customizing in der Kabine

Die Relevanz einer flexiblen und modular aufgebauten kollaborativen Methodik, wie in der Forschungsfrage gefordert, zeigt sich insbesondere im Kontext des Customizings und des Entwicklungsprozess von Flugzeugkabinen. Anstelle einer frühzeitigen Festlegung auf vordefinierte Varianten sollte eine Methodik die Fähigkeit besitzen, adaptiv auf sich dynamisch verändernde Kundenanforderungen oder den Wechsel von Zulieferern zu reagieren. Wie bereits im Abschnitt 4.2 aufgezeigt, ist die Kabine von einer hohen individuellen Anpassung seitens der Airlines geprägt. Die Variantenvielfalt in der Kabine ist groß. Das sogenannte Customizing beschreibt die Anpassung eines Angebotes an die speziellen Wünsche des Kunden[8]. In dieser Arbeit wird daher der Begriff Customizing folgendermaßen definiert:

**Definition 4.3.1 Customizing**

Customizing ist die individuelle Anpassung eines Angebots (Serienprodukt, Dienstleistung, Software) an die Bedürfnisse und Wünsche eines Kunden.

Ein anpassbares Produkt zeichnet sich durch eine Reihe von Standardkomponenten aus, die durch Auswahl des Kunden um zusätzliche Optionen erweitert wird [BBH+18]. Abbildung 4.4 zeigt die zeitliche Einordnung des Customizing in den Entwicklungsprozess und die Abgrenzung zur Variante am Beispiel eines Flugzeugs. Varianten sind vordefinierte Optionen, die in bestimmten Merkmalen voneinander abweichen. Mit Version 1.1 werden drei verschiedene Varianten eines Flugzeugs untersucht und vorausgelegt. In dem gezeigten Entwicklungskontext bezeichnet eine Version ein vorläufiges Zwischenprodukt oder ein Endprodukt mit einer abgeschlossenen Form [Pro22].

[8] Oxford Learner's Dictionaries, 2023 Oxford University Press.

Im zeitlichen Verlauf des Produktentwicklungsprozesses wird Variante 1.1b weiterverfolgt und als finale Variante n.1b schlussendlich ausgelegt (1). Allerdings kann nun eine spezifische Anpassung seitens des Kunden erfolgen und die ausgewählte Variante n1.b wird modifiziert. Das Resultat ist eine neue, customizierte Variante n1.e, die sich von Variante n.1b ableitet (2).

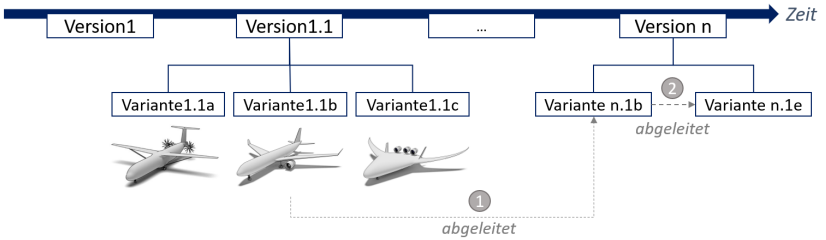



Abbildung 4.4: Zeitliche Einordnung einer Variante (1) im Entwicklungsprozess und im Customizing (2) mit Abgrenzung zur Version.

Flugzeughersteller bieten den Kunden eine Vielzahl an Variationen in der Kabine an. Der Kunde kann aus einem Katalog unterschiedliche Optionen beispielsweise für die Gepäckablage oder für Lichtszenarien auswählen. Änderungen innerhalb der Kabine haben große Auswirkungen auf andere Subsysteme im Flugzeug wie Struktur, Wasser/Abwasser oder Elektrik [RW17]. Abbildung 4.5 zeigt die Variationsmöglichkeiten am Beispiel der Lavatory vom Zulieferer Diehl Aviation. Der Kunde hat neben der Auswahl von Farbe und Material auch die Möglichkeit, die Komponenten mit Funktionalitäten an sich zu variieren. Dies führt dazu, dass die Architektur jeder Flugzeugversion einzigartig ist, selbst wenn diese innerhalb der gleichen Flugzeugfamilie an denselben Kunden geliefert wird [BBH+18]. Diese einzigartige Kundenversion wird auch Head-of-Version genannt [RW17, S. 94]. Design, Entwicklung und Zertifizierung (siehe Abschnitt A.1.2) werden einmalig dafür durchgeführt. Alle nachfolgenden Flugzeuge der gleichen Kundenflotte werden als Re-Build-Flugzeuge bezeichnet [RW17, S. 94].

Obwohl die kundenspezifische Anpassung als Strategie viele Vorteile bietet, sind bei der Umsetzung einige Aspekte zu berücksichtigen. Ein hoher Grad an individuellen Anpassungen im Rahmen des Customizings führt sowohl zu einer Erhöhung der Komplexität im Entwicklungsprozess als auch der Kosten im Bereich Design, Produktion, Lagerhaltung und Logistik [HKRT05, DJT06]. Darüber hinaus werden nur vergleichsweise geringe Stückzahlen jeder einzelnen Variante von Flugzeug und Kabinenausstattung umgesetzt [LHK21].

Ein Grund für die Erhöhung der Komplexität und der Kosten ist die enge Verzahnung zwischen Zulieferern und dem OEM des Flugzeugs. Während die Installation der verschiedenen Kabinenelemente ins Flugzeug durch den Flugzeughersteller durchgeführt wird, werden diese zum Großteil von externen Lieferanten (Supplier) entwickelt und bereitgestellt. Der Hersteller ist daher bei den Anpassungsoptionen für die Fluggesellschaften abhängig von den Zulieferern. Trotz der begleitenden Phase bei der Entwicklung der Flugzeugkabine zwischen dem Kunden (Airline) und dem

Flugzeughersteller sowie der früh stattfindenden Berücksichtigung von kundenspezifischen Anpassungen, entstehen nachträgliche Änderungen und Anpassungswünsche am Design. Die anschließende Durchführung dieser Änderungen erhöhen die Head-of-Version Kosten, den Arbeitsaufwand und die Prozesszeiten. Gründe hierfür sind der zeitintensive Datenaustausch mit den Zulieferern, um die Änderungen (Kundenwunschdefinition) voll umfassend zu überprüfen und umzusetzen (Konfigurationsumsetzung) sowie der Datentransfer, der hauptsächlich dokumentenbasiert stattfindet.

Abbildung 4.5: Customizingauswahl am Beispiel der Lavatory von Diehl Aviation [Die23]. Der Kunde wählt zwischen Materialien, Funktionalitäten wie Beleuchtung oder berührungslosen Bedienelementen und Stauraum.

Ziel ist daher, in Richtung definierte Produktvarianten und -familien mit einer flexiblen Lieferkette zu denken [RW17, S. 9]. Die Digitalisierung unterstützt dabei als Treiber. Digitale Technologien ermöglichen die gemeinsame Entwicklung von Produkten zwischen Kunde und Hersteller [PV22]. Ein digitaler end-to-end Prozess mit durchgängigen automatisierten Prozessen und Tools sowie einem integrierten, einheitlichen Datenmodell könnte diese Herausforderungen in der Entwicklung, Fertigung und Installation adressieren.

#### Zwischenfazit

Varianten spielen eine große Rolle im Flugzeugentwurf. Zum einen variiert die Gesamtarchitektur eines Flugzeugs. Dazu gehören zum Beispiel die Flugzeugstruktur (Rumpfform) oder verschiedene Antriebssysteme (Kerosin oder Wasserstoff). Zum

anderen ist die Flugzeugkabine besonders von kundenspezifischen Anpassungen geprägt. Im Rahmen des Customizings hat die Fluggesellschaft die Möglichkeit, die Kabine individuell nach ihren Vorgaben zu designen und anzupassen.

Gründe für die Variantenmodellierung sind unter anderem schnell auf neue Märkte und Technologieentwicklungen zu reagieren sowie den vorhandenen Entwurfsbereich zu erkunden. Zudem können dadurch Produktfamilien modelliert, Designalternativen analysiert und miteinander bewertet werden. Aus der Literatur abgeleitet zeigt sich, das ein Hindernis für die Variantensteuerung die diversen Dokumentations- und Modellarten darstellt. Die vielen im Customizing-Prozess verwendeten heterogenen Modelle erschweren das Variantenmanagement und Einpflegen von Änderungen. Hinzu kommt, dass Produktkonfigurationen eine hohe Flexibilität in den Produktions- und Lieferkettenprozessen fordern. Daher ist eine methodische Unterstützung gefragt, die sowohl Modularität im System für Varianten als auch eine Modularität in der Methodenstruktur für die Anbindung heterogener Modelle ermöglicht, sodass nachträgliche Anpassungen einfacher durchgeführt werden können. Methoden, die bereits bei der Variantenmodellierung zum Einsatz kommen, werden im nächsten Abschnitt vorgestellt.

## 4.4 Methoden zur Variantenmodellierung

Vor dem Hintergrund der in dieser Arbeit formulierten Forschungsfrage, die auf eine methodische Unterstützung von Modularität im Gesamtsystem sowie auf die systematische Abbildung von Varianten komplexer technischer Systeme abzielt, ist eine umfassende Auseinandersetzung mit dem Stand der Technik im Bereich der Variantenmodellierung im Rahmen der Descriptive Study I erforderlich. In diesem Kontext haben sich bereits einige Methoden und Ansätze etabliert. Ziel dieser Methoden sind die Aufbereitung von Informationen für das Variantenmanagement, um erst Variantenvielfalt zu reduzieren und somit Komplexität zu vermeiden und anschließend die verbleibende Komplexität zu beherrschen. Der folgende Abschnitt stellt verschiedene Ansätze für die Variantenmodellierung aus den Bereichen SysML, Konstruktion und Modulbauweise vor.

### 4.4.1 Variantenmodellierung in der SysML

Ausgehend von der Forschungsfrage dieser Arbeit, wie Varianten modelliert werden können, rückt im Kontext des MBSE die Anwendung der SysML als Methode zur Variantenmodellierung in den Fokus. Im Folgenden wird untersucht, welche Möglichkeiten und Grenzen die Ansätze in der SysML in diesem Kontext bieten.

Aus Studien zur Anwendung von SysML mit Teilnehmern aus Industrie und Forschung wie von Santos und Soares [SS21, SS23] oder Wolny et al. [WMC+20] zeigt sich, dass die SysML vor allem für die Modellierung von Anforderungen und die Spezifikation von Systemarchitekturen in der Design- und Validierungsphase verwendet wird. Darüber hinaus wird die SysML für die Untersuchung von Varianten verwendet. Zur Unterstützung des Variantenmanagements werden mit der SysML v2 daher neue Elemente eingeführt, um die Variantenmodellierung zu verbessern [FGBN23]. In der

Literatur gibt es bereits eine Reihe von Ansätzen und Methoden Varianten innerhalb eines Modells und mit der Modellierungssprache SysML zu modellieren. Menninger und Wiechel haben in einer Literaturrecherche mehrere Ansätze untersucht und daraus Anforderungen an das modellbasierte Variantenmanagement abgeleitet, damit insgesamt eine durchgängige, modellbasierte Abbildung von Varianz stattfinden kann [MWR+22]. Diese sind unter anderem Rückverfolgbarkeit von Abhängigkeiten zwischen den Modellartefakten und die explizite Abbildung derer Varianz. Weitere Anforderungen sind die Nutzung einer formalisierten Modellierungssprache für die Variantenmodellierung und die Unterstützung durch entsprechende Tools [MWR+22]. Erfüllen sich diese Anforderungen, kann eine holistische Betrachtung und Modellierung von Varianten stattfinden. Dabei hoben sich die drei Ansätze (i) VAMOS, (ii) FODA und (iii) das 150%-Modell hervor, die nachfolgend vorgestellt werden.

#### Variantenmodellierungsmethode für SysML (VAMOS)

Für die Modellierung von Varianten gibt es die von Weilkiens definierte VAMOS-Methode zur Modellierung von Varianten mit SysML. VAMOS (Variant Modeling Method for SysML) basiert auf der bereits in Abschnitt 3.2.4 vorgestellten Methode SYSMOD. Die SysML bietet keine Modellelemente für Variantenkonzepte [Wei16]. Dennoch kann ein Systemmodell mit Varianten mit Hilfe allgemeiner Modellelemente wie der Generalisierungsbeziehung aufgebaut werden. Im Rahmen der VAMOS-Methode wird die Sprache SysML um ein Konzept zur Variantenmodellierung durch zusätzliche Stereotypen wie Variationspunkte, Variation und XOR Bedingung erweitert [Wei16]. Letzteres beschreibt, dass eine Klasse A entweder eine Assoziation zu einer weiteren Klasse B oder C hat, aber nicht gleichzeitig zu beiden oder zu keiner. Die Variantenstereotypen können auch ohne SYSMOD angewandt werden. Abbildung 4.6 zeigt die Begriffe von VAMOS in einem Blockdefinitionsdiagramm. Alle Elemente, die in jeder Systemzusammenstellung enthalten sind und unabhängig von Variantenelementen sind, werden im Kern (the core) abgebildet. Der Variationspunkt dient als Andockpunkt für ein Variantenelement. Eine Variation beschreibt den Typ und dient als Unterscheidungsmerkmal für Varianten (Beispiel: Motortyp), während die Variante den zugehörigen vollständigen Satz von Elementen darstellt (Beispiel: elektrisch, hybrid, Diesel). Die Variantenbedingungen *XOR* und *REQUIRES* definieren Regeln für die Zusammenstellung von Varianten. So können entweder Varianten ausgeschlossen werden, sobald eine bestimmte Variante ausgewählt ist oder explizit eine weitere Variante hinzugefügt werden, da diese erforderlich ist [Wei16]. Abhängige Parts (durch Komposition dargestellt) erben automatisch den VariationPoint und müssen nicht nochmals entsprechend benannt werden. In der Variantenkonfiguration wird letztlich festgelegt, welche Varianten- und Kernelemente für das Gesamtsystem genutzt werden.

Eine Beispielanwendung der VAMOS-Methode ist die systematische Entwicklung einer föderierten Datenbankinfrastruktur und eines Managementsystems durch Melzer et al. [MPWT22]. Melzer et al. nutzten VAMOS, um drei verschiedene Datenbanktypen zu modellieren, sodass Projektteams ihre eigene Datenbank unabhängig voneinander entwickeln und dabei dieselbe Infrastruktur nutzen können. Ein weiteres Beispiel für die Modellierung von Varianten in SysML wurde von Forlingieri und Weilkiens auf dem Gebiet der modellbasierten Produktlinienentwicklung (Product Line Engineering, PLE) aus einer industriellen Perspektive betrachtet [FW22]. Sie verglichen zwei Varianten

der Modellierung, wobei die Verwendung von VAMOS mehr Aufwand bei der Modellierung von Variation erfordert, aber eine bessere Verwaltung und Beherrschung der Variabilität innerhalb des Modells bietet [FW22].

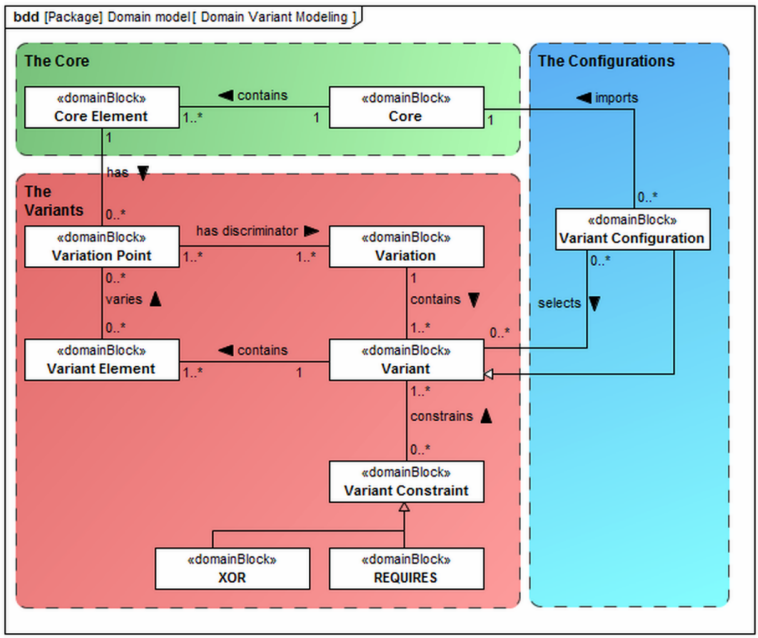



Abbildung 4.6: Variantenmodellierungsdomäne der VAMOS-Methode nach [Wei16].

Insgesamt wird mit VAMOS die Varianz auf physischer Ebene modelliert, während diese auf der funktionalen und logischen Ebene vernachlässigt wird [MWR+22]. Daher entwickelten Madeira et al. den Ansatz aus [FW22] bei Airbus weiter. Mit dem Ansatz MBPLE4MOFLT ist eine funktionsbasierte Produktlinienentwicklung und ein Variantenmanagement einschließlich der Auswirkungen der Variabilität bei der Betriebsanalyse sowie bei den funktionalen, logischen und technischen Architekturen möglich [MdSPF23]. Mit der Methodik wird eine Co-Architektur für Produkt, Fertigung und Dienstleistung unterstützt [MdSPF23]. Am Beispiel eines Flugzeugs mit optionalen Möglichkeiten für den Bodenbetrieb (Ground Operations) wurde die Methode demonstriert. Der Ansatz wird aktuell erweitert, um die Variabilität auch in anderen Bereichen wie 3D-Entwurfselementen oder textuellen Anforderungen zu berücksichtigen [Air23b]. Bei dem Ansatz zeigt sich allerdings, dass aufgrund der starken Vernetzung untereinander und der Aufteilung in diverse Variationspunkte eine nachträgliche Anpassung der Architektur in großen Systemmodellen erschwert ist.

#### Funktionsorientierte Domänenanalyse (FODA)

Die Funktionsorientierte Domänenanalyse (Feature Oriented Domain Analysis, FODA) ist eine Methode für die graphische Modellierung von Produkt- und Serienvarian-

ten mit Hilfe von Varianten- oder Merkmalbäumen von Kang et al. [KCH+90]. Dabei werden einfache Konfigurationsregeln verwendet und die Variabilität aus der Perspektive der Beteiligten modelliert. Bei der Analyse werden die Merkmale des Systems, ihre Variabilität und Einschränkungen zwischen den Varianten gezeigt und unterstützen beim Verständnis in Bezug auf den Anwendungsbereich und die gemeinsamen Anforderungen [Wei16, KCH+90]. Primäres Ziel der FODA-Methode ist die Wiederverwendbarkeit von Domänenprodukten [KCH+90]. Aus diesem Grund liegt der Schwerpunkt auf der Identifizierung von parallellaufenden Prozessen und gemeinsamen Modellen. Durch die Aufteilung der Architektur in Schichten und Modellierung pro Ebene können Auswirkungen von technischen und Anforderungsänderungen auf das Modell lokalisiert werden.

Toolumgebungen, wie der PTC Integrity Modeler, haben die FODA-Notation bereits implementiert und das Konzept wird in vielen Variantenbeschreibungen und Modellierungsansätzen verwendet [Abu22, Wei16]. Ein Variantenbaummodell hat eine Baumstruktur mit einer Eltern-Kind-Beziehung zwischen den verschiedenen Features. Diese ermöglicht einen Überblick über die Domäne in Bezug auf Umfang, Merkmale und gemeinsame Anforderungen. Abbildung 4.7 zeigt einen beispielhaften Merkmalbaum für ein Flugzeug. Pflichtmerkmale werden mit einer Linie und optionale mit einem Kreis am Ende der Linie symbolisiert. Die Darstellung alternativer Merkmale erfolgt durch einen Bogen zwischen zwei Linien. Die zugehörigen Features sind exklusiv und können nicht gleichzeitig im selben Produkt existieren. Mit Kompositionsregeln wird die Semantik zwischen Features definiert, die nicht im Merkmalsbaum ausgedrückt werden kann. In diesem Beispiel muss bei Auswahl der Lounge eine Kabine mit einer Mindestlänge von 25 m ebenfalls ausgewählt werden. Alternative Merkmale werden nicht willkürlich gewählt, sondern erfolgen auf Grundlage von Wünschen des Kunden. Daher können zusätzliche Informationen in Gründen festgehalten werden, um bei der Auswahl eines Features zu unterstützen.

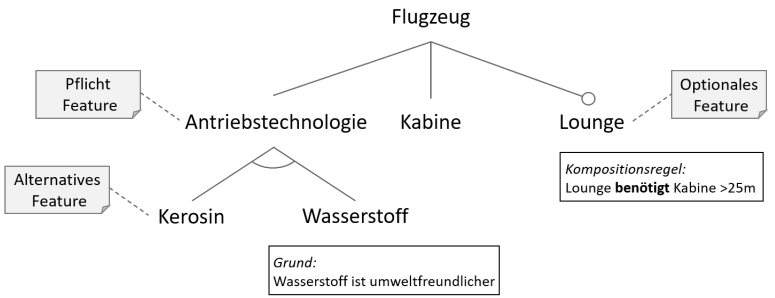



Abbildung 4.7: Beispiel eines FODA-Merkmalsbaums mit Features nach [KCH+90].

Die FODA-Methode setzt sich aus drei Phasen zusammen: der Kontextanalyse, einer anschließenden Domänenmodellierung und der Architekturmodellierung [KCH+90]. In der Kontextanalyse werden die Domänengrenzen und die Umgebung, in der die Applikation genutzt wird, definiert. In der Domänenmodellierung werden zuerst die allgemeinen Fähigkeiten der zu entwickelnden Applikation aus Sicht des Nutzers analysiert und Varianten identifiziert. Anschließend erfolgt die Entwicklerperspektive auf die zu

entwickelnde Applikation mit der Implementierung interner Funktionen. Die funktionale Analyse verknüpft funktionale Gemeinsamkeiten und Unterschiede der Anwendung zusammen mit den Anforderungen. Abschließend wird in der Architekturmodellierung eine Softwarelösung für das zuvor definierte Problem bereitgestellt. Dieses Architekturmodell dient als Grundlage für den detaillierten Entwurf und für die Konstruktion der Komponenten. Der systematische Ansatz der FODA-Methode kann daher als Unterstützung zu Beginn in den (Software-)Entwicklungsprozess für die Aufnahme und Verwaltung von Anforderungen sowie für die Identifikation wiederverwendbarer Funktionalitäten (RFLP-Ansatz) eingebunden werden [Mar06]. Menninger et al. stellen heraus, dass die Methode FODA durch die disziplinspezifische Fokussierung auf Software beschränkt ist [MWR+22].

#### Das 150%-Modell

Das 150%-Modell oder auch der annotative Ansatz benötigt kein separates Variabilitätsmanagementtool. Stattdessen wird die Variabilität durch Erweiterung von Profilen und Stereotypen in der SysML unterstützt. Alle Systemvarianten werden innerhalb eines einzigen überspezifizierten Modells modelliert (Problembereich). Dieses Modell enthält alle Systemvarianten. Dabei kann ein Feature durch beliebig viele Modellelemente abgebildet werden [BBS+19]. Im Lösungsbereich erfolgt die Spezifikation und Implementierung der gemeinsamen und variablen Merkmale [BCS+20]. Die Varianten werden erzeugt, indem Merkmale oder Features entfernt werden, sodass nur die gewünschten Teile im System verbleiben (100%-Systemmodell). Die gemeinsamen Merkmale sind immer in jeder Produktvariante enthalten [BCS+20]. Idealerweise werden die Merkmale so standardisiert erstellt, dass keine Änderung am Gesamtprodukt bei Ergänzungen stattfinden muss [Bil20]. Der annotative Ansatz ist limitiert, indem keine Modellierung von Variabilität in Verhaltensdiagrammen möglich ist und eignet sich besonders für Varianten mit einem hohen Grad an Gemeinsamkeiten [MWR+22]. Produktfamilienkonzepte (siehe Unterkapitel 4.2) lassen sich besonders gut mit diesem Ansatz auslegen. Abbildung 4.8 zeigt den generischen Ansatz für die Modellierung einer Systemvariante mit dem 150%-Modell.

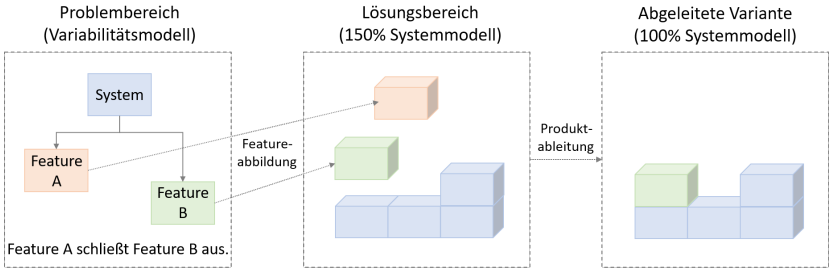



Abbildung 4.8: Darstellung des 150%-Modells nach [BBS+19].

Bilic et al. nutzen den Ansatz des 150%-Modells für die Modellierung von Variabilität im Systementwurf [BBS+19] und kombinierten MBSE mit der Produktlinienentwicklung. Als Beispiel diente die Auslegung von Volvo Baumaschinen, die sich eine

gemeinsame Motorplattform teilen, aber jeweils unterschiedliche Anforderungen erfüllen müssen und variierende Schnittstellen zu weiteren Subsystemen (Getriebe oder Hydraulikpumpe) aufweisen. An diesem Beispiel wurde gezeigt, dass Merkmale auf verschiedenen Abstraktionsebenen nachverfolgt werden können, gleichzeitig die Vorlaufzeit durch die systematische Wiederverwendung verkürzt wird und somit auf eine Verbesserung der Effizienz im Entwicklungsprozess hinweist [BCS+20]. Weitere Beispiele für den annotativen Ansatz sind der Einsatz in der Luftfahrt für die Entwicklung von Produktlinien auf Grundlage der Variabilität im Funkmanagement [HIL+12], die Auslegung von Produktfamilien mit Varianten [HH15] oder die Erstellung von Supersets für die Erzeugung produktspezifischer Instanzen von Getrieben in der Automobilbranche [YCP+17]. Ein großer Vorteil dieser Methode liegt auf der Nachvollziehbarkeit von Designentscheidungen für den Stakeholder, indem Architekturinformationen gespeichert werden und verschiedene Ansichten auf die Variante pro Produkt möglich sind [YC19].

#### Zwischenfazit

Mit Blick auf die Fragestellung und den abgeleiteten Herausforderungen in der Modellierung von Varianten (siehe Abschnitt 2.1) werden die vorgestellten Lösungsansätze aus der Literatur nach den ermittelten Anforderungen **A.1** bis **A.3** bewertet. Der Erfüllungsgrad der jeweiligen Anforderung wird mit Hilfe der in Tabelle 4.1 dargestellten Bewertungsskala bestimmt.

Tabelle 4.1: Bewertungsskala.

| Bewertung | Symbol |
|---|---|
| Sehr hoch | ● |
| Hoch | ◕ |
| Niedrig | ◐ |
| Sehr niedrig | ◔ |
| Nicht erfüllt | ○ |

Betrachtet werden der Einfluss auf die induzierte Modellkomplexität, der Entwicklungsaufwand zur Erweiterbarkeit um neue Varianten und das Potential zur Wiederverwendbarkeit der Modelle bei umfangreichen, komplexen Systemen. Tabelle 4.2 gibt einen Überblick der vorgestellten Ansätze zur Modellierung von Varianten und stellt mit Hilfe einer Bewertungsskala den Erfüllungsgrad der jeweiligen Anforderung dar. Zusammenfassend zeigt sich, dass die vorhandenen Lösungsansätze zur Variantenmodellierung bei der Abbildung umfangreicher Systeme mit viel Varianz alle drei Anforderungen nur bedingt erfüllen. Durch die Vielzahl an benötigten Modellelementen und der Modellierung aller Varianten innerhalb eines Einzelmodells erhöht sich die induzierte Modellkomplexität. Folglich erhöht sich der Entwicklungsaufwand für die Erweiterung um neue Varianten aufgrund der Vielzahl an Verschachtlungen und der Modellkomplexität.

Tabelle 4.2: Überblick der Anforderungserfüllung durch Ansätze zur Variantenmodellierung.

| | ***Anforderungen*** | | |
|---|---|---|---|
| **Ansätze** | **A.1** Induzierte Modellkomplexität | **A.2** Erweiterbarkeit | **A.3** Wiederverwendbarkeit |
| VAMOS | ◑ | ◔ | ◑ |
| FODA | ◑ | ◔ | ◕ |
| 150% Modell | ◔ | ◔ | ◕ |
| Bewertungsskala: | sehr hoch ● | hoch ◕ | niedrig ◑ | sehr niedrig ◔ | nicht erfüllt ○ | |

### 4.4.2 Variantenmodellierung in der rechnerunterstützten Konstruktion

Die Modellierung von Varianten stellt eine Herausforderung in vielen Domänen und Disziplinen dar. Im Bereich der rechnerunterstützten Konstruktion gibt es ebenfalls einige Ansätze für die (kollaborative) Modellierung von Varianten und Umsetzung von Familienkonzepten. Mit Hilfe eines Produktdatenmanagementsystems werden die Daten der Produktfamilien verwaltet, Produktdaten vernetzt und Varianten entwickelt [Män00]. Änderungen an den Produktfamilien finden innerhalb des PDM-Systems statt [Män00]. Für die Wiederverwendung von bereits implementierten CAD-Modellen für ähnlich aufgebaute Konstruktionsanwendungen bietet die parametrisch-assoziative Konstruktion, kurz PAKo Methode, einen Ansatz. Dabei werden in den Anwendungsprogrammen nur die Modellierungsschritte gespeichert, sodass die Endgeometrie nach jeder Änderung neu berechnet wird [Abu11]. Im Kontext dieser Arbeit bietet die PAKo Methode potenzielle Synergien zur Beantwortung der Forschungsfrage an. Ein Einblick in das Vorgehen wird im Folgenden gegeben.

#### Parametrisch-assoziative 3D-CAD-Modellierung (PAKo Methode)

Die PAKo Methode ermöglicht die direkte Ableitung neuer Konstruktionen aus dem Gesamtmodell, sobald sich die Konstruktionsgrößen ändern oder ausgetauscht werden (Abbildung 4.9). Diese Modelle sind einfach rekonfigurierbar und ermöglichen eine schnelle 3D Darstellung individueller Produktkonfigurationen. Neben der Produktgestalt wird bei dieser Methode ebenfalls die Konstruktionsabsicht abgespeichert [Tec08]. Dadurch können unterschiedliche Varianten vorher ausgelegt und Familienkonzepte entwickelt werden.

Das Grundprinzip dieser Methode beruht darauf, dass die geometrischen Objekte miteinander verknüpft werden und durch uni- oder bidirektionale Elementverknüpfungen (geometrische Zwangsbedingungen oder Parameter) nachträglich veränderbar

sind (assoziativ) [Obe17]. Parameter können dabei über Regeln intern oder von externen Quellen wie Excel gelesen oder über programmiert Makros aus anderen CAE-Programmen gesteuert werden [Tec08]. In Konstruktionstabellen wiederum können Parameterfamilien definiert werden, mit denen die Einzelteile oder Zusammenbauten der Konstruktion gesteuert werden [Tec08]. Schlussendlich dient der Ansatz der Optimierung im Hinblick auf die Hauptanforderungen Gestaltung, Funktionalität und Fertigung.

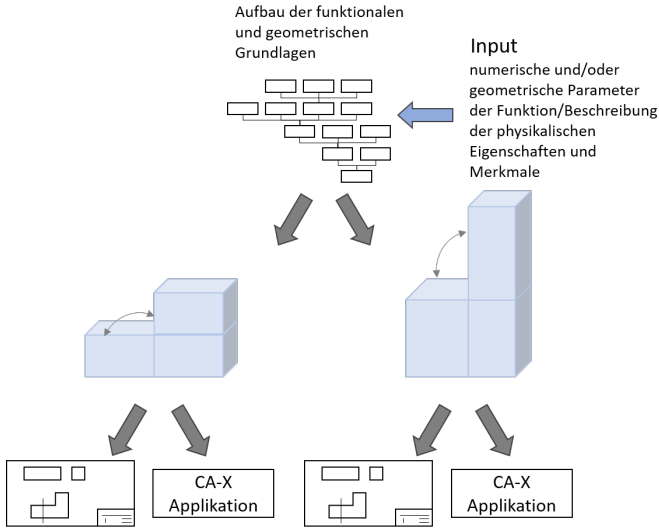



Abbildung 4.9: Parametrisch-assoziatives Design nach [Tec10].

Für die Datenverarbeitung kam zudem das EVA-Strukturierungsprinzip im Konzeptentwicklungsprozess zum Einsatz [Tec08, S. 6]. Das Kürzel steht für Eingabe-Verarbeitung-Ausgabe. Dabei werden die Informationen strukturiert in Ordnern entsprechend den drei Kategorien hinterlegt und in einer bestimmten Reihenfolge verarbeitet. Dies führt zu einer optisch klaren Trennung der Abläufe und Aufbereitung der Informationen. Abbildung 4.10 veranschaulicht die Ordnerstruktur. Im Ordner “Eingabe“ befinden sich Parameter und Daten, die entweder aus dem Adapter oder über ein anderes externes Datenformat importiert wurden. Der zweite Ordner “Verarbeitung“ enthält alle Informationen und Funktionen für die strukturierte Auslegung oder Konstruktion des Produktes. Zudem werden hier die Ergebnisse gesammelt. Anschließend erfolgt die Rückgabe und Veröffentlichung der Ergebnisdaten im Ordner “Ausgabe“.

Mit moderner CAD-Software wie CATIA V5 (Nachfolgesystem V6), NX oder ProE ist die Verknüpfung in CAD-Systemen mit wissensbasierten Methoden aus dem Engineering (KBE) möglich [Tec08]. Die Formelbeziehungen stellen Parameter in Abhängigkeit zueinander und ermöglichen die Überprüfung von Anforderungen bereits während der Konstruktion. Zudem können mit Hilfe der objektorientierten Schnittstellenprogrammierung weitere CAx-Programme angebunden werden. Limitiert ist der Ansatz dadurch, dass diese Herangehensweise höhere Lizenzkosten verursacht, da für

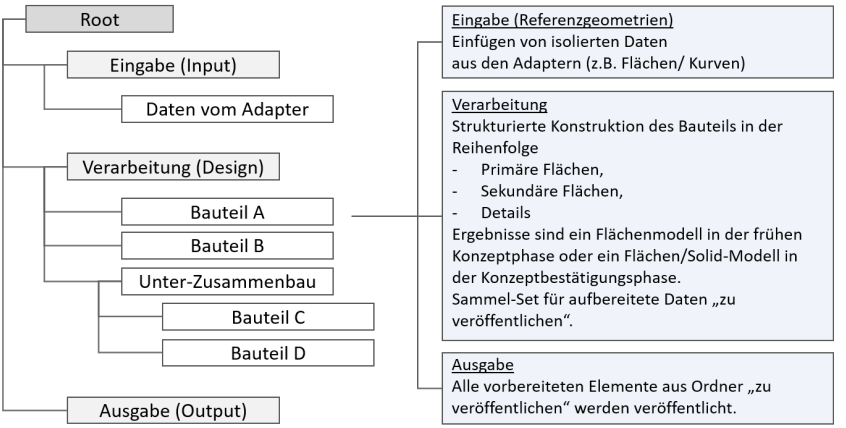



Abbildung 4.10: Darstellung des EVA-Prinzips nach [Tec08].

die wissensbasierte Integration Sonderfunktionen notwendig sind. Weiterhin ist die Methode nur für vorab definierte Varianten geeignet. Ein weiterer Nachteil ist der erhöhte Aufwand bei der Erstellung der Startmodelle. Darüber hinaus modelliert jeder Konstrukteur individuell, wodurch Modelle nicht ohne weiteres von anderen Nutzern übernommen werden können. Weitere Herausforderungen sind die hohe Elementanzahl im Modell und die starke Vernetzung der einzelnen Geometrieelemente untereinander. Strategien zur Beherrschung der Herausforderungen sind die Einführung von Standards in Form von Regeln für den Aufbau des Modells, eine vordefinierte Namenskonvention für die CAD-Modellelemente und die Nutzung von Adaptermodellen [Abu11]. Allgemein stellt sich im Kontext dieser Arbeit das EVA-Prinzip aber als ein vielversprechender strukturierender Ansatz innerhalb der Modelle zur Beantwortung der Forschungsfrage dar.

### 4.4.3 Variantenmanagement und Abgrenzung zur Modularisierung und dem Prinzip der Baukästen

Neben den bisher betrachteten Methoden bietet die Literatur eine Vielzahl weiterer Ansätze zur Variantenmodellierung, insbesondere aus der Verkehrsindustrie, die im Folgenden näher erläutert werden. Zur Bewältigung der variantenbedingten Komplexität in der Entwicklung setzen Lösungsansätze auf eine variationsorientierte Modularisierung der Produktstruktur sowie auf Standardisierung im Variantenmanagement [Sch02]. Dabei wird ein System oder Produkt unterteilt. Die Untergliederung erfolgt in mehrere, funktional und physisch voneinander unabhängige Subsysteme, die Module. Die Module werden wiederum über definierte Schnittstellen miteinander verbunden, um ins Gesamtprodukt eingebunden die übergeordneten Anforderungen an das Produkt zu erfüllen [Sch02]. Insgesamt reduziert der Ansatz der Modularisierung die Komplexität in der Produktstruktur und vereinfacht die Produktvariantenbildung. Nach Schmidt handelt es sich hierbei um sogenannte Modulbaukastensysteme [Sch02].

#### 4.4.3.1 Produktstrukturstrategien

In der Literatur werden die Begriffe Baukasten, Modulbaukasten, Bausteine und Module zur Beschreibung ähnlicher Inhalte synonym verwendet [KVI+21]. Im Folgenden wird daher ein tiefer gehender Einblick auf das Baukastenprinzip und die Modulbaukastenstrategie gegeben und diese anschließend voneinander abgegrenzt.

**Baukastenprinzip**

Das Prinzip der Baukastensysteme kann nach Krause und Gebhardt als ein ideelles Grundprinzip der Produktstrukturierung verstanden werden, um Stückzahlen zu erhöhen und dabei Einsparungen durch Skaleneffekte innerhalb einer Produktfamilie zu ermöglichen [KG18]. Für die Entwicklung von Baukästen wird von „Neukonstruktionen oder bereits bestehenden Produkten ausgegangen, die bisher nicht durch einen Baukasten systematisiert sind“ [KVI+21]. Dabei verfolgt das Prinzip der Baukasten die schnelle Konfiguration verschiedenster Produktvarianten mit einer begrenzten Anzahl an Bausteinen [KG18]. Der Baukasten ist ein Kombinationssystem von Bausteinen (Bauteile oder Baugruppen). Die Bausteine erfüllen unterschiedliche Funktionen mit einer einheitlichen Schnittstelle [KG18]. Die Klassifizierung der Bausteine wird durch die Funktionsstruktur bestimmt und orientiert sich nach den Teilfunktionen des Produkts. Eine Unterteilung der Gesamtfunktion erfolgt in die Kategorien Grund-, Hilfs-, Sonder-, Anpass- und auftragsspezifische Funktionen [KVI+21]. Abbildung 4.11 zeigt

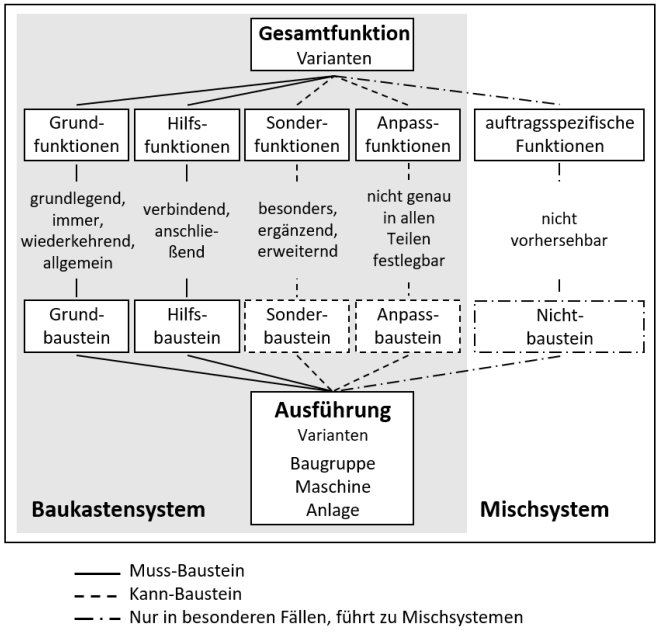



Abbildung 4.11: Unterscheidung von Funktions- und Bausteinarten nach dem Baukastenprinzip von Pahl und Beitz nach [KVI+21].

die Funktionsarten sowie die jeweils entsprechend definierten Bausteinarten. Grundfunktionen finden Verwendung in mehreren Produktvarianten, während Hilfsfunktionen mehrere von ihnen miteinander verbinden. Die Sonder- und Anpassfunktionen sind optional und von unterschiedlicher Ausprägung. In Einzelfällen (nicht vorhersehbar) werden auftragsspezifische Funktionen ergänzt und das Baukastenprodukt mit Nicht-Bausteinen kombiniert [KVI+21]. Diese Erweiterung führt zu Mischsystemen.

**Modulbaukastenstrategie**

Im übergeordneten Sinn beschreibt der Begriff Modulbaukasten das Potential, im Gegensatz zum Baukasten eine breitere Mehrfahrverwendung von Modulen auch über das Produktprogramm hinaus zu verfolgen, indem eigenständige Einheiten (Module) gebildet werden [KG18]. Dabei übernehmen einzelne, austauschbare Module jeweils eine Funktion oder einen klar definierten Funktionsumfang [KG18]. Die Module orientieren sich stark an den Kundenbedürfnissen und deren Variation. Ziel ist es, mit der Kombination einer minimalen Anzahl an Modulen verschiedene Produktkonfigurationen zu konfigurieren [KG18]. Die erstellten Module können ebenfalls für andere Produktfamilien genutzt werden. Abbildung 4.12 zeigt diese Zusammenhänge. Die Pfeile veranschaulichen eine Wiederverwendung der vordefinierten Module sowohl innerhalb der Produktfamilie als auch produktfamilienübergreifend.

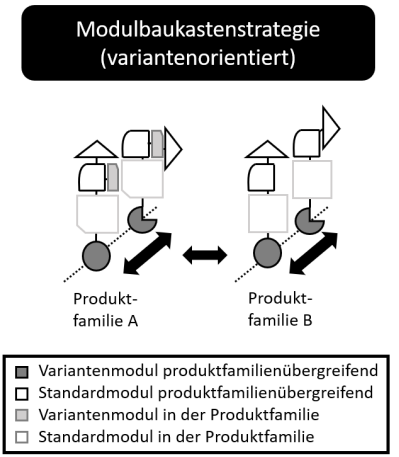



Abbildung 4.12: Modulbaukastenstrategie für die Produktstruktur von Krause und Gebhardt nach [KG18].

Von dieser Strategie erhoffen sich Krause und Gebhardt zum einem eine Reduktion der unternehmensinternen Komplexität sowie die Nutzung von Stückzahleneffekten durch die hohe Wiederverwendung von bereits erbrachter Entwicklungsleistung in den Modulen [KG18, KVI+21]. Darüber hinaus muss für ein Gesamtsystem keine vollständige Baukastensystematik entwickelt werden und die Strategie des Modulkasten

kann auch nur für ausgewählte Bereiche Anwendung finden [KVI+21]. Für die Kombinierbarkeit mit verschiedenen Modulen benötigt es allerdings mehr Schnittstellen, sodass die Schnittstellengestaltung eine der zentralen Herausforderungen darstellt. Dies kann dazu führen, dass die Schnittstellen größer dimensioniert werden, als es für eine Einzelkonstruktion der Varianten erforderlich wäre.

Für die Modulbildung von Subsystemen in der Produktstruktur finden unterschiedliche Methoden Anwendung. Dafür müssen bei der Modularisierung von Subsystemen und dem Management von Varianten die Beziehungen der Komponenten untereinander sortiert und analysiert werden. Der Darstellung von Beziehungsnetzen bedient man sich zwei unterschiedlicher Formen. Diese sind zum einen Graphendiagramme und zum anderen Matrixdiagramme. Graphendiagramme eignen sich für die Darstellung von Verknüpfungsreihenfolgen und -pfade, während Matrixdiagramme für die Analyse von Beziehungsmustern Anwendung finden [Tit21, Mau07].

#### 4.4.3.2 Graphendiagramme

Graphen eignen sich für die Darstellung von komplexen Strukturen, um Muster zu erkennen oder Beziehungen zwischen Objekten zu modellieren. Darüber hinaus wird dieser Modelltyp für die Strukturierung oder Abbildung von erfahrungsbezogenem Wissen verwendet [Wac21, S. 6]. Die visuelle Repräsentation eines Graphen ist ein Graphendiagramm. Die Methoden für die Analyse und Darstellung zwischen Objekten mit netzartigen Strukturen wird in der Graphentheorie festgehalten [Tit21]. Die Graphen bestehen aus Elementen (Knoten) und deren Verbindungen (Kanten). Unterschieden wird zwischen gerichteten Graphen, wenn die Kanten eine bestimmte Richtung haben, und ungerichteten Graphen, wenn die Kanten keine Richtung haben. Bei gerichteten Graphen entspricht die Verbindung zwischen zwei Knoten einer Eltern-Kind-Beziehung. Kanten wiederum könnten gewichtet sein, indem zusätzliche Informationen über die Beziehung angegeben werden. Knoten werden durch beschriftete Rechtecke oder Kreise repräsentiert, Kanten durch Linien [Tit21]. In einem Graphendiagramm können die Knoten entweder zufällig angeordnet sein oder einer hierarchischen Struktur folgen [Tit21]. Die Graphentheorie bietet eine Vielzahl an Algorithmen zur Lösung der Graphen im Hinblick auf die Analyse von Beziehungsmustern oder kritischen Pfaden an [Wac21, Tit21]. Zwei Beispiele für Graphen sind in Abbildung 4.13 dargestellt.

Die Graphentheorie bietet die Grundlage für einige angewandte Methoden in der Produktentwicklung und der Untersuchung von Abhängigkeiten zwischen Systemelementen. In der Graphentheorie werden Systemkonstellationen durch die strukturellen Substrukturen und Attribute charakterisiert [Mau07]. Dabei stellt die Graphentheorie Operationen zur Verfügung, mit der z.B selbst verstärkende Effekte zwischen Produktkomponenten identifiziert werden können [Mau07]. Chamas et al. wiederum nutzen Metriken der Graphentheorie um Funktionsdiagramme in der SysML zu modifizieren mit dem Ziel, die strukturelle Komplexität zu reduzieren und SysML-Modelle quantitativ bewerten zu können [CMRP18, S.152].

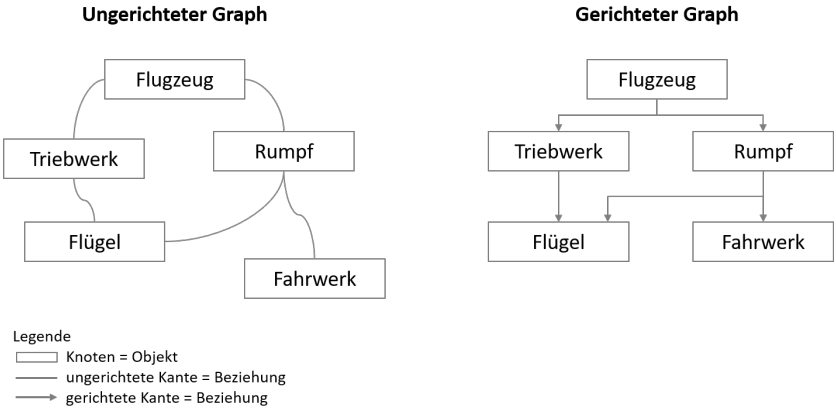



Abbildung 4.13: Beispiele für jeweils ein ungerichtetes und gerichtetes Graphendiagramm.

#### 4.4.3.3 Matrixdiagramme

In Matrixdiagrammen erfolgt die Abbildung von Elementbeziehungen in einer zweidimensionalen und tabellarischen Darstellung. Allgemein kann jedes Beziehungsnetz, das in einem Graphendiagramm dargestellt ist, in eine Matrix überführt werden [Tit21]. Die Matrix bildet dabei den Graph nur in einer anderen Darstellungsform ab [Mau07]. Nach Maurer werden Matrixdiagramme zunehmend eingesetzt, um die Komplexität technischer Systeme und Produktentwicklungsprozessen zu managen [Mau07]. Unter anderem werden mit den Matrizen Komponenten unterschiedlicher Disziplinen untersucht, um z.B. die Abhängigkeiten zwischen den Produktfunktionen und den Kundenanforderungen zu betrachten. Im Ingenieurwesen findet die diagrammartige Darstellung überwiegend als Design Structure Matrix (DSM) Verwendung. Die DSM zeigt eine systematische Abbildung der Elemente und ihrer Beziehungen in einer quadratischen n x n-Matrix. Für jede Sicht auf den Entwicklungsprozess können mehrere Arten von DSM-Modellen erstellt werden. Der Vorteil bei Matrixdiagrammen ist die eindeutige Abbildung ohne Schnittkanten, was ermöglicht, diese mit Methoden der Matrixalgebra automatisch zu analysieren und zu optimieren [MIKO99, dA08]. Einen vertiefenden Einblick in den Aufbau von Matrixdiagrammen geben Steward [Ste81] und Maurer [Mau07].

#### 4.4.3.4 Einsatz von Matrixdiagrammen für die Modularisierung

Mit Hilfe von Matrixdiagrammen können Submodule eines Systems zu Clustern gruppiert werden. Der Ablauf der Umwandlung eines Beziehungsnetzes in einen Graphen über eine Matrix in eine sortierte Matrix ist in Abbildung 4.14 dargestellt. Der ungerichtete Graph links zeigt die Bindung zwischen den einzelnen Submodulen bzw. Komponenten (symbolisiert durch Zahlen) des Gesamtsystems. Der nächste Schritt

überführt den Graph in eine DSM. In dieser sind die Komponenten sowohl auf der vertikalen als auch auf der horizontalen Achse aufgeführt. Die Kreuze in den einzelnen Zellen markieren eine Beziehung zwischen den jeweiligen Elementen. Auf der Diagonalen sind die Zellen ausgegraut und kennzeichnen die direkte Zuordnung zum selben Element. Unter Einsatz von Sortierungsmethoden (z.B. binäre Sortierung) wird die Matrix dann neu angeordnet [Sch02, S. 112]. Die resultierende Matrix auf der rechten Seite der Abbildung zeigt die potentiell größeren Modulpaarungen entlang der Hauptdiagonalen. Ein Anwendungsszenario für die DSM ist das Produktcustomizing. Mit Hilfe der Teilmengen können Entwickler Produkte identifizieren, bei denen Anpassungsmaßnahmen durch die identifizierten Verknüpfungen keine Auswirkungen auf weitere Submodule haben [Mau07, S. 56]. Darüber hinaus dient die Gruppierung als Grundlage für die Strukturdefinition von Entwicklungsteams [Mau07].

Einige Hersteller setzen in ihrer Produktentwicklung auf modulare, baukastenartige Produktstrukturen. Limitiert ist dieses Vorgehen allerdings bei stark miteinander verknüpften Teilmengen und bei spezifischen Individuallösungen, die erst im Laufe der Entwicklung konkretisiert werden, wie es unter anderem beim Customizing der Flugzeugkabine der Fall ist. Daher werden Ansätze zum Variantenmanagement im Bereich der Modularisierung unter Zuhilfenahme von Matrixdiagrammen oder der Graphentheorie im Rahmen dieser Arbeit nicht weiter verfolgt.

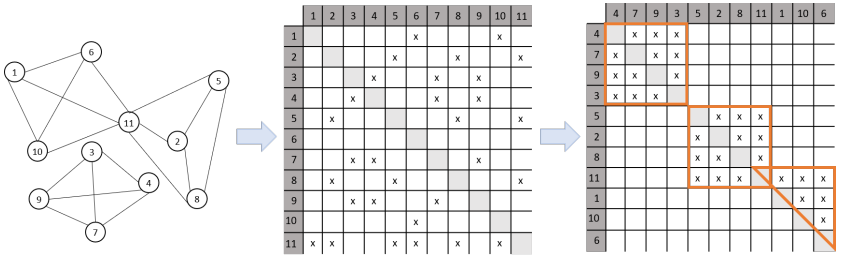


Abbildung 4.14: Schematische Darstellung der Überführung eines Graphen in eine Matrix zur Modularisierung von Produkten nach [Mau07, S. 102].

## 4.5 Kapitelfazit

Die Flugzeugkabine ist von einer hohen individuellen Anpassung geprägt und bietet eine Vielzahl an Variationsmöglichkeiten für Subsysteme an. Für die Modellierung von Varianten oder für das Customizing und zur Beherrschung der damit verbundenen Komplexität in der Produktvielfalt sind einige Methoden entwickelt worden. Die Literatur zeigt Ansätze aus der CAD-Modellierung wie die parametrisch-assoziative 3D Modellierung (PAKo) von Varianten. Im Bereich der modellbasierten Systementwicklung mit der SysML zeigen die Methoden VAMOS, FODA und das 150% Modell Herangehensweisen für die Abbildung von Variabilität im Systementwurf.

In allen gezeigten Ansätzen findet sich ein Einzelmodell, indem die bereits vorher bekannten Produktvarianten integriert und modelliert werden. Dabei sind zum Entwicklungsstart alle Konfigurationen oder Produktfamilien bekannt. Begrenzt sind diese Methoden, sobald neuartige Varianten untersucht werden sollen, die erst durch

die Weiterentwicklung von Technologien entstehen oder sich durch neue Kundenbedürfnisse ergeben. Eine nachträgliche Integration neuer Features in die bestehende Modellarchitektur ist mit einem erhöhten Aufwand verbunden. Die Flexibilität ist eingeschränkt. Zudem wachsen durch die steigende Anzahl an Komponenten ebenfalls die Wechselwirkungen untereinander. Die Modellierung in nur einem Modell wird zu komplex und verschachtelt. Stattdessen muss die Methode zur Abbildung von Varianz modular und erweiterbar sein.

Zusammenfassend sind die Anforderungen **A.1** bis **A.3.** mit den bereits vorhandenen Lösungsansätzen nur in einem gewissen Umfang erfüllt. Die Literaturbetrachtung der Variantenmodellierung mit der SysML ist abgeschlossen. Für die erfolgreiche Beantwortung der Forschungsfrage müssen im Folgenden die Anforderungen **A.4** und **A.5** für die Kopplungsmöglichkeiten innerhalb eines Unternehmens als auch über die Unternehmensgrenze hinweg mit weiteren domainspezifischen als auch disziplinübergreifenden Modellen untersucht werden.

# 5. Aktuelle Entwicklungen und Ansätze zur Modellkopplung

> *„Eine Herausforderung die oft nicht ausreichend betrachtet wird, ist die Schnittstelle und Verknüpfung zwischen domänenübergreifender und domänenspezifischer Modellierung. Bisher fehlen Ansätze zur Verknüpfung und Integration domänenübergreifender und -spezifischer Modelle [...]. “* [SI21, S. 2]

Im vorangegangenen Kapitel wurden verschiedene Lösungsansätze zur Modellierung von Varianten mit der SysML aufgezeigt und anhand der ermittelten Anforderungen **A.1** bis **A.3** bewertet. Für die Beantwortung der zugrundeliegenden Forschungsfrage müssen darüber hinaus die Anforderungen **A.4** und **A.5** an eine multidisziplinäre als auch unternehmensübergreifende Zusammenarbeit erfüllt werden. Schumacher und Inkermann sehen hierbei als große Herausforderung die fehlenden Ansätze für die Verknüpfung von domänenunabhängigen und domänen-spezifischen Modellen (hier gleichzusetzen mit dem Begriff Disziplin) [SI21]. Infolgedessen werden in diesem Kapitel Schnittstellentechnologien und Lösungsansätze für die Modellkopplung im Hinblick auf eine Kollaboration mit mehreren (externen) Experten genauer beleuchtet. Hierbei wird zwischen zwei Arten der Zusammenarbeit unterschieden.

Zum einen wird die Kopplung mehrerer SysML-Modelle betrachtet, die mit der gleichen Modellierungssoftware aufgesetzt wurden. In Kapitel 4 wurde dargelegt, dass die Modellierung von Varianten überwiegend in einem Einzelmodell stattfindet. Gleichzeitig wirken mehrere Experten an der Modellierung mit, indem sie einen spezifischen Teilaspekt des Systems oder ein ganzes Subsystem modellieren. Aufgrund dessen werden in diesem Kapitel Lösungsansätze für die kollaborative Zusammenarbeit mit mehreren Experten in einem SysML-Modell und für die Kopplung mehrerer SysML-Modelle genauer untersucht.

Zum anderen wird die Kopplung multidisziplinärer Modelle mit SysML-Modellen untersucht. Bereits in Kapitel 3.1 wurde aufgezeigt, dass die Entwicklung von Flugzeugen in einem heterogenen Umfeld stattfindet. Damit eine ganzheitliche Betrachtung des Systems stattfinden kann, müssen verschiedene Disziplinen und deren dazugehörige Modelle miteinander kommunizieren. Durch die Verknüpfung und einen automatisierten Informationsaustausch zwischen den unterschiedlichen Tools kann der Entscheidungsprozess unterstützt werden, indem Änderungen und Einflüsse auf weitere Subsystemgruppen über die eigene Systemgrenze hinaus nachverfolgt werden können [FJN22, MWR+22]. Ohne die Kopplung sind Wechselwirkungen nicht direkt erkennbar. Beispiele für Wechselwirkungen zwischen Systemgruppen reichen von der Beanspruchung gleicher Bauräume bis hin zu Sicherheitsabständen [Mah06]. Zudem ermöglichen neue Materialien und Technologien multifunktionale Eigenschaften von

Komponenten. Diese Komponenten übernehmen dadurch mehrere Funktionen und sind nicht mehr klassisch einer Systemgruppe zuzuordnen. Beispiele sind Seitenwände aus Aerogelen, die sowohl eine akustische und thermische Isolation ermöglichen [ASP+23] sowie multifunktionale Seitenwände, die als Lautsprecher dienen oder Informationen für den Passagier auf die Oberfläche projizieren [Mis20, Ham21]. Dadurch ist keine eindeutige Modellierungsabgrenzung mehr möglich, sodass mehrere Experten in einem Modell zusammenarbeiten müssen.

Das folgende Kapitel beleuchtet Standards und historische Ansätze für den Datenaustausch mehrerer Modelle. Zudem stellt es Herangehensweisen für die interne Kopplung von Modellen der SysML und die externe Modellkopplung mit heterogenen Modellen vor. Abschließend werden Herausforderungen und Grenzen von Lösungsansätzen aufgezeigt. Damit schließt das Kapitel die Descriptive Study I der DRM ab. Zusammen mit den vorangegangenen Kapiteln erfolgt eine komprimierte Analyse des Stands der Forschung und Technik im Kontext der in dieser Arbeit behandelten Forschungsthematik.

## 5.1 Interne Modellkopplung zwischen SysML-Modellen

Abschnitt 4.4.1 zeigt, dass für die Modellierung von Varianten mit der Sprache SysML mit den Methoden meistens ein Einzelmodell aufgebaut wird, welches alle Systemaspekte abbildet. Für die gezeigten Anwendungsbeispiele funktionieren die Ansätze gut. Allerdings führt dies zu Herausforderungen, sobald das abzubildende System wächst. Dadurch wird ebenfalls das Einzelmodell größer und aufgrund der steigenden Verbindungen untereinander komplexer. Zudem sind an der Modellierung des Gesamtsystems Flugzeug mehrere Experten gleichzeitig beteiligt. Die Entwicklung ist auf Experten in verschiedenen Abteilungen und Entwicklungsteams aufgeteilt [VB24]. Darüber hinaus findet zunehmend auch eine kollaborative Arbeit bei der modellbasierten Entwicklung zwischen mehreren Unternehmen statt [LFH24]. Dabei bringt jeder eigenes Fachwissen zur Modellierung seines bestimmten Systemaspekts oder Subsystems mit. Um dieser verteilten Systementwicklung entgegenzuwirken, wird eine Speicherung aller Informationen in einem zentralen Modell angestrebt [Pow23, VB24]. Daher bieten Modellierungstoolentwickler bereits Lösungen für die parallele Entwicklung von verschiedenen Subsystemen mit der SysML an, die anschließend wieder zu einem einzelnen Gesamtmodell zusammengeführt werden.

Im Folgenden wird der Stand der Technik zur Zusammenarbeit mit mehreren Experten an einem SysML-Gesamtmodell durch gekoppelte SysML-Modelle untersucht. In der Literatur [VB24, FW22, Das19, MdSPF23] und der praktischen Projekterfahrung in Zusammenarbeit mit Industriepartnern im Luftfahrtkontext finden sich viele Beispiele zur Anwendung der SysML mit dem Modellierungstool Cameo Systems Modeler, sodass hierauf ein Fokus bei der Betrachtung gelegt wird. Insgesamt hoben sich die drei Ansätze zur Kopplung mittels Share-Package Funktion, Server und OSLC hervor.

### 5.1.1 Datenaustausch durch Share-Package Funktion

Die integrierte Share-Package Funktion des Cameo Systems Modeler ermöglicht die Freigabe von ausgewählten Paketen einschließlich seiner eigenen Elemente an wei-

tere SysML-Modelle [No 22b]. Beide Projekte können die freigegebenen Pakete nutzen. Dabei wird nicht der gesamte Inhalt eines verwendeten Projekts im Hauptprojekt sichtbar. Nur die gemeinsamen Inhalte werden geteilt und können genutzt werden. Das Konzept ist vergleichbar mit den öffentlichen Attributen einer Klasse in Programmiersprachen wie Java [No 22b]. Abbildung 5.1 zeigt ein Beispiel für die Freigabe von Paketen. Im Containment-Baum des teilenden Projektes wird das freigegebene Paket durch ein Symbol mit einer gebenden Hand (1) am Ordner gekennzeichnet. Im Projekt 2 wird das geteilte Paket importiert und durch ein weißes Rechteck (2) gekennzeichnet. Zudem ist der Ursprungsort des Projektes in hellgrau (2) angezeigt. Die zugehörigen Elemente können im neuen Projekt für die weitere Verwendung genutzt werden (Kompositionsbeziehung in Abbildung 5.1 unten).

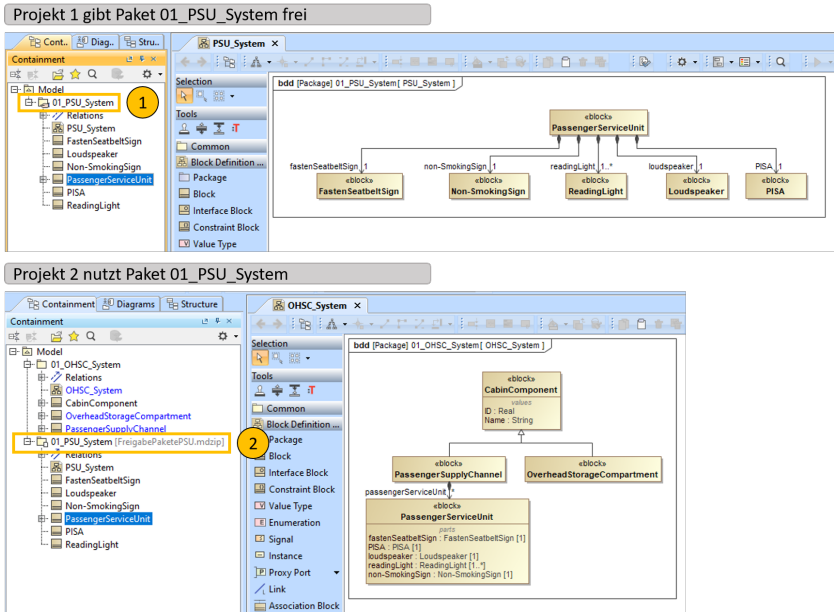



Abbildung 5.1: Freigabe von Paketen und deren Elementen zwischen zwei SysML-Projekten.

Die Modelldekomposition muss auf Paketebene erfolgen, kleinere Elemente können nicht aufgeteilt werden. Die Entscheidung, wo man die Grenze zur Aufteilung zieht, hängt davon ab, dass die Subelemente eine logisch vollständige Struktur bilden und nicht stark von anderen Strukturen abhängig sind [No 22b]. Die Partitionierung eines Modells kann daher nicht bei einem stark gegenseitig abhängigen Subsystem angewandt werden, sondern nur bei einseitig abhängigen Modellteilen. Zusätzlich ist diese Funktion nur für lokale Projekte und die Editionen Standard, Professional, Architect und Enterprise verfügbar.

Ein Beispiel für die Nutzung der geteilten Pakete zeigt Mahboob [Mah21, S. 100]. Dabei werden Submodelle eines Systems in einzelnen SysML Modellen erstellt und

diese anschließend in ein übergeordnetes Kernmodell eingeladen. Das Kernmodell verwendet die importierten Submodelle für die weitere Auslegung des Systems und stellt jede Lösungsarchitektur für bestimmte Probleme dar [Mah21].

### 5.1.2 Datenaustausch über Server

Eine Alternative für die individuelle Entwicklung separater Modellelemente zu Ansätzen, wie der Freigabe von Paketen und deren späterer Zusammenfügung, stellt die kollaborative Modellierung mit Hilfe eines Servers, wie der Teamwork Cloud von CSM, dar. Hierbei wird ein verwaltendes Verzeichnis (Repository) zur Speicherung von Projekten und Nutzern eingesetzt [No 22b]. Dieses unterstützt ebenfalls bei der Versionierung von Projekten. Die Projekte werden über das Netzwerk auf dem Server gespeichert, sodass jedes dienstanfordernde Gerät (Client) mit dem installierten Cameo Systems Modeler darauf zugreifen kann. Je nach Rolle verfügen die Clients über unterschiedliche Berechtigungen für das Projekt. Die Teamwork Cloud unterstützt eine Versionierung auf Elementebene und eine rollenbasierte Zugriffskontrolle. Bei der Benutzung kann parallel auf dasselbe Modell und sogar Diagramm zugegriffen und abgeändert werden. Dabei erhält jeder Nutzer sofort die neueste Version. Verfügen Teammitglieder über keinen Zugang zum Netzwerk, erlaubt der Server die Speicherung einer lokalen Datei, die geteilt werden kann. Nach Ausführung der Änderungen am Projekt durch Dritte wird das ursprüngliche Projekt mit der lokalen Datei abgeglichen und aktualisiert. Abbildung 5.2 zeigt die Teamwork Cloud Architektur für die parallele Nutzung eines SysML-Modells.

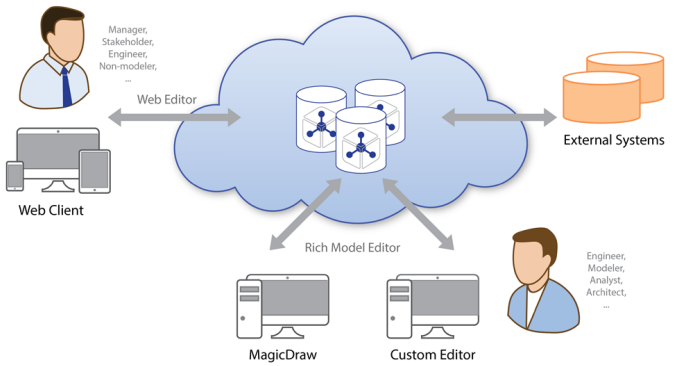



Abbildung 5.2: Darstellung der Teamwork Cloud Architektur von [Das23].

Die Teamwork Cloud (blaue Wolke) ist für die Arbeit mit großen Modellen konzipiert worden. Die Rechenleistung wird reduziert, indem die übertragene Datenmenge abhängig von der Größe der Änderung ist. Das Produkt kann bei beliebigen Cloud-Service Anbietern eingesetzt werden und ermöglicht die Integration von externen Tools über standardisierte Programmierschnittstellen (Application Programming Interface, APIs) [Das23]. Genutzt wird die Teamwork Cloud beispielsweise bei Airbus für die Modellierung von Systemen mit Variabilität auf verschiedenen hierarchischen

Ebenen [MdSPF23]. Dabei wird das SysML Modell des Gesamtsystems in Submodule je nach spezifischem System und Fachabteilung unterteilt und über einen Integrator an die Teamwork Cloud in ein großes CSM Modell gekoppelt. Die Definition der Schnittstellen/Interfaces und Abgrenzung der Submodule untereinander stellt dabei eine der größten Herausforderungen dar. Eine zusätzliche Funktion, die der Server bereitstellt, besteht darin, dass er als OSLC-Provider fungieren kann, um Modellelementdaten sowie weitere Tools zu integrieren. Einen Einblick dazu gibt der folgende Abschnitt.

### 5.1.3 Datenaustausch mit OSLC

Ein weiterer Ansatz zur Kopplung von Modellierungstools und Integration von Daten stellt die Initiative OSLC dar (Open Services for Lifecycle Collaboration). Ziel dieses übergeordneten Ansatzes ist die Integration von Tools, um Daten zu verbinden und damit den Digitalen Faden über Disziplinen, Anwendungen und Organisationen hinweg zu erreichen (Abbildung 5.3) [OAS23]. OSLC ist ein offener Datenmodell- und Dienstleistungsstandard in einem Verbund von Tools [HG23]. Das OSLC-Framework ist somit für eine Service-orientierte Architektur (SOA) ausgerichtet und ermöglicht die Interaktion zwischen Toolumgebungen und Disziplinen, indem Ressourcen als Dienste angeboten und von Clients oder Konsumenten entdeckt werden [Ins21]. Diese multidisziplinäre Integration wird im Systems Engineering in den Bereichen Anwendungs- (Application Lifecycle Management, ALM) und Produktlebenszyklusmanagement eingesetzt [Wös15]. OSLC definiert hierbei die Ontologie für den gesamten Lebenszyklus.

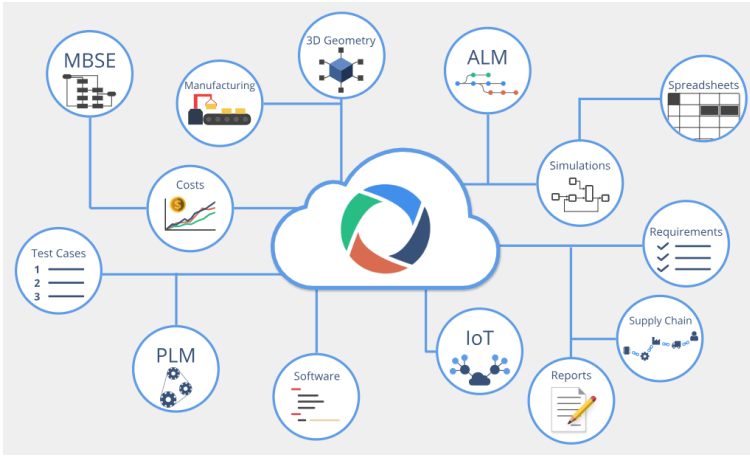



Abbildung 5.3: OSLC als Integrator zwischen Disziplinen und Anwendungen von [OAS23].

Das Prinzip von OSLC basiert auf dem Resource Description Framework (RDF) des World Wide Web Consortiums (W3C) und ist ein offener Standard. Die dabei verwendete Methode Linked Data beschreibt frei im Internet verfügbare, strukturierte Daten,

die durch einen Identifikator eindeutig identifiziert sind, und wie diese miteinander verbunden werden. Die Basisstruktur ist ein Tripel und spiegelt einen Graphen wider [HG23]. Es besteht aus einen Subjekt (Objektelement 1), einem Prädikat (Beziehung) und einem Objekt (Objektelement 2). Dabei können die Objektelemente sowohl Eigenschaften, Objekte oder Anforderungen sein. Das Ergebnis ist eine Adresse (Uniform Resource Identifier, URI). Indem auf die URI der jeweiligen Daten verwiesen wird, können Nutzer Quellen- und Datenbank-unabhängig auf verschiedene Informationen in den diversen Tools zugreifen. Die Abhängigkeiten bzw. Verlinkungen existieren allerdings nur in eine Richtung und folgen einer bestimmten Hierarchie. Toolumgebungen, in denen die Links nicht existieren, fragen daher vor Ausführung ab, ob Informationen (Parameter) aus anderen Toolumgebungen für diese zur Verfügung stehen. Anschließend erfolgt die Zuordnung dieser Informationen an die Parameter innerhalb der eigenen Toolanwendung. Gründe hierfür sind vor allem die Sicherstellung von Datenkonsistenz sowie die Pflege und Verwaltung von Änderungen. Bei dem Ansatz von verknüpften föderierten Daten existieren keine Replikationen von Daten. Der Ansatz ist standardisiert, kollaborativ und unterstützt digitale Lebenszyklusübergreifende Analysen und Sichtweisen [HG23]. Durch die Wiederverwendung von Artefakten und Komponenten ist zudem eine Ableitung von Varianten aus einer Baseline möglich.

Der OSLC-Ansatz wird bereits von einigen Toolumgebungen wie Matlab, Cameo Systems Modeler, Modelica oder IBM Doors unterstützt. Dabei ermöglicht dieser die Integration bestehender Tools, ohne dass deren Implementierung bekannt ist [HG23]. Stattdessen wird vor jedes Tool ein Interface geschaltet, dass dann wiederum an den Server ankoppelt und auf die OSLC Architektur zugreift (Abbildung 5.4).

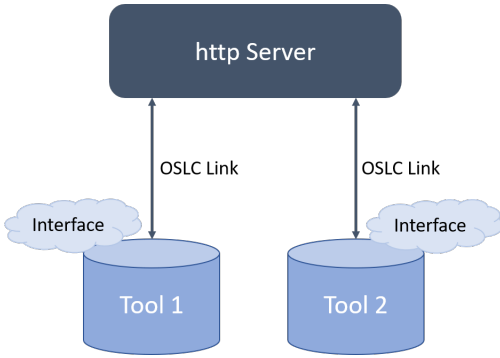



Abbildung 5.4: Schematische Darstellung der Kopplung von zwei Tools mit OSLC.

Für die Interfaces gibt es bereits Standards, die genutzt und an das jeweilige Tool angepasst werden können. Allerdings befindet sich dieser Ansatz noch in der Entwicklungsphase und befasst sich nicht nur speziell mit der Kopplung von SysML-Modellen. Als allgemein gehaltener Ansatz soll OSLC bereits etablierte Standards langfristig ersetzen und in der SysML v2 fest integriert werden [Wös15]. Anwendung findet der OSLC-Ansatz beispielsweise für die Verknüpfung von verschiedenen Tools im Bereich des Anforderungsmanagements [HNC+24].

### 5.1.4 Herausforderungen und Grenzen

Aufbauend auf der Darstellung verschiedener Methoden für die Kopplung von mehreren SysML-Modellen in den vorangegangenen Kapitelabschnitten erfolgt im Folgenden eine Betrachtung der dabei auftretenden Herausforderungen. Denn auch wenn die Kopplung vielfältige Potenziale für die Integration und Zusammenarbeit bietet, ergeben sich in der praktischen Umsetzung technische, methodische und organisatorische Fragestellungen, die bei der Anwendung berücksichtigt werden müssen. Diese sind:

- Modellkonsistenz,
- Pflege und Wartung der Artefakte,
- Modellgröße.

Für die Gewährleistung der Modellkonsistenz müssen alle Modelle den gleichen Modellierungsansatz und kompatible Methoden zur Integration verwenden [SI21]. Nur dadurch ist eine widerspruchsfreie Beschreibung und Darstellung möglich.

Bei der direkten Verwendung von Paketen, Diagrammen und Elementen aus anderen Projekten, wie bei dem vorgestellten Ansatz der Share-Package Funktion, spielt die Datenpflege eine wichtige Rolle. Zwar werden Änderungen eines geteilten Pakets in den anderen Projekten übernommen. Inwieweit diese zu Problemen bei der weiteren Verwendung führen, weil Informationen fehlen oder Konflikte in Simulationen verursachen, ist nicht direkt nachvollziehbar. Stattdessen müssen die Änderungen manuell überprüft werden. Eine Mitteilung über eine Änderung gibt es durch das Modellierungstool nur bei der Nutzung der Teamcloud. Auftretende Konflikte müssen dann manuell aufgelöst werden. Die Teamcloud bietet hier die Funktion an Elemente zu sperren, um Konflikte bei der Modellierung zu vermeiden [Das23].

Werden Artefakte aus einem SysML-Modell in ein anderes SysML-Modell über die Share-Package Funktion eingeladen, so wächst die Größe des Modells an. Da bei dem Import immer alle notwendigen Diagramme, Verknüpfungen und Blöcke mit übertragen werden, steigt die Modellgröße und der gewünschte Effekt einer jeweiligen Reduzierung der Modellkomplexität sowie der Rechenleistung tritt nicht ein.

Im Kontext der Forschungsfrage spielen die genannten Herausforderungen eine wichtige Rolle für die Anwendbarkeit in der Praxis. Abgeleitet aus der ergänzenden Forschungsfrage dieser Arbeit ergeben sich weitere Kriterien an die Methode, die für eine praktische Anwendung im industriellen Kontext relevant sind. In der industriellen Praxis führt eine steigende Modellgröße oft zu Intransparenz, längeren Verarbeitungszeiten und höherem Abstimmungsaufwand zwischen den Experten. Daher muss eine Alternative geschaffen werden, um die Komplexität zu reduzieren. Parallel dazu ist das Aufwand-Nutzen-Verhältnis ein entscheidender Erfolgsfaktor für die Anwendung in der Praxis. Besonders im Bereich der Modellpflege, also bei der kontinuierlichen Anpassung, Erweiterung und Nachverfolgung von Änderungen, muss der Pflegeaufwand durch strukturierte und nachvollziehbare Methoden möglichst gering gehalten werden. Gleichzeitig soll der Nutzen in Form von Wiederverwendbarkeit, Konsistenz und Effizienzsteigerung deutlich erkennbar sein. Aufbauend auf diesen Erkenntnissen

werden die Anforderungen **A.4** und **A.5** für die disziplinspezifische als auch disziplinübergreifende Kopplung verfeinert. Daraus leiten sich für die multidisziplinäre Zusammenarbeit aus **A.4** die Anforderungen **A.4.1** und **A.4.2** ab.

**A.4.1** **Einfluss auf Modellgröße** - Für die Betrachtung des Gesamtsystems müssen mehrere Modelle miteinander gekoppelt werden. Dabei darf die Implementierung der Schnittstelle die Modellkomplexität nur bedingt erhöhen und muss sich in einem vertretbaren Mehraufwand halten.

**A.4.2** **Modellpflege** - Die Pflege der Schnittstelle soll vereinfacht werden, sodass diese von jedem Experten für sein spezifisches Modell selbst durchgeführt werden kann, ohne dabei das Gesamtsystem zu kompromittieren.

Neben den methodischen Herausforderungen spielen weitere Anforderungen im Kontext der Anwendbarkeit in der industriellen Praxis mit Fokus auf der unternehmensübergreifenden Zusammenarbeit mehrerer Experten eine Rolle. Die Anforderung **A.5** betrachtet die unternehmensübergreifende Zusammenarbeit, woraus sich die Anforderungen **A.5.1** zur Lizenzunabhängigkeit und **A.5.2** für die Automatisierung des Datentransfers für die Ermöglichung einer flexiblen Kommunikation ableiten.

**A.5.1** **Lizenzabhängigkeit** - Bei der Verwendung einer allgemeingültigen, standardisierten und kostenneutralen Schnittstelle ist der Aufwand zur Integration bei externen Partnern geringer, wodurch die Akzeptanz für eine Zusammenarbeit mit Gesamtverbund gefördert wird. Die Erfolgschancen für ein dezentrales partnerübergreifendes, multidisziplinäres Gesamtmodell steigen.

**A.5.2** **Automatisierung** - Der Austausch von Ergebnissen oder Daten zwischen Modellen erfolgt überwiegend unter der Verwendung von Dokumenten. Ein automatisiertes Ein- und Auslesen von Informationen zur Kopplung der Modelle ermöglicht eine effizientere Zusammenarbeit.

#### Schlussfolgerungen

In der Literatur finden sich für die Kopplung mehrerer SysML-Modelle bzw. für den Aufbau eines zentralisierten Modells bereits einige Ansätze. Tabelle 5.1 fasst die Ansätze zusammen. Im Hinblick auf die Beantwortung der Forschungsfrage werden die vier Anforderungen **A.4.1** bis **A.5.2** betrachtet und anhand derselben Bewertungsskala aus Abschnitt 4.4.1 evaluiert. Diese sind der Einfluss auf die Modellgröße aufgrund der Kopplungsart, die Modellpflege, die Lizenzabhängigkeit und das Potential zur Automatisierung der Schnittstelle.

Sowohl der Datenaustausch durch die Freigabe-Funktion von Paketen als auch die direkte Kopplung von Daten durch das OSLC Format ermöglichen die Kopplung von mehreren SysML-Modellen. Bei der Nutzung des bereitgestellten Teamwork Cloud Servers von Dassault Systèmes können zwar mehrere Experten gleichzeitig an einem SysML-Modell arbeiten und Einzelpakete zur individuellen Arbeit exportieren, letztendlich wird dennoch ein einzelnes SysML Gesamtmodell entwickelt. Die Herausforderungen, wie Unübersichtlichkeit durch die Modellgröße und Modellpflege aufgrund auftretender Konflikte durch unterschiedliches Modellierungsvorgehen der Experten, bleiben bestehen. Bei allen Ansätzen ist eine Automatisierung der Schnitt-

stelle und damit ein durchgängiger, nicht-manueller Datenabruf gegeben. Sowohl die Share Package Funktion als auch die Team Cloud setzen eine Lizenz voraus.

Tabelle 5.1: Überblick der Anforderungserfüllung durch Ansätze zur Kopplung von mehreren SysML-Modellen für den Cameo Systems Modeler.

| | ***Anforderungen*** | | | |
|---|---|---|---|---|
| | **A.4** Multidisziplinäre Zusammenarbeit | | **A.5** Externe Zusammenarbeit | |
| **Ansätze** | **A.4.1** Einfluss auf Modellgröße | **A.4.2** Modellpflege | **A.5.1** Lizenz-abhängigkeit | **A.5.2** Automatisierung |
| Share Package Funktion | ○ | ◔ | ○ | ● |
| Server/ Team Cloud | ○ | ◔ | ○ | ● |
| OSLC | ◑ | ◑ | ◕ | ● |
| Bewertungsskala: | sehr hoch ● | hoch ◕ | niedrig ◑ | sehr niedrig ◔ | nicht erfüllt ○ |

Für die multidisziplinäre Zusammenarbeit mit internen als auch externen Experten müssen neben der disziplingleichen Kopplung auch weitere Modelltypen angebunden werden. Lediglich der in der Entwicklungsphase für den Datenaustausch zwischen SysML-Modellen befindende OSLC Standard bietet hier die Chance, SysML-Modelle und darüber hinaus externe heterogene Modelle miteinander zu koppeln, ohne die genannten Herausforderungen mitzubringen. Daher werden im folgenden Abschnitt weitere Schnittstellentechnologien für die multidisziplinäre Kopplung untersucht.

## 5.2 Externe Modellkopplung von SysML-Modellen und heterogenen Modellen

Modellbasierte Ansätze und speziell die Modellierungssprache SysML wird häufig nur bis zur abstrakten Architekturdefinition eingesetzt, sodass Entwurfsentscheidungen nicht systematisch auf Grundlage der funktionalen Architektur des Systems bewertet werden [HJZ+21]. Die Systemauslegung erfolgt in verschiedenen Disziplinen. Dabei werden unterschiedliche Arten und Formen von Modellen im Entwicklungsprozess erarbeitet und verwendet [HST+23]. Für eine holistische Betrachtung und Auslegung eines Produktes werden weitere Modellumgebungen benötigt. Dies sind unter anderem Simulations-, Konstruktions-, Visualisierungs- oder Analyseumgebungen. Die Interoperabilität zwischen deskriptiven Systemmodellen und funktionalen Modellen, wie z.B. CAD, sind noch nicht ausreichend erforscht [HST+23]. Bei der Modellkopplung bzw. dem Datenaustausch zwischen den verschiedenen Disziplinen wird zwischen

zwei Herangehensweisen unterschieden. Erstens, stellen Modellierungsumgebungen der SysML bereits interne Anbindungsfunktionen bereit. Zweitens, werden Daten über einen Adapter, ein Hilfsmittel, das als Austauschformat und Mittler zwischen den Disziplinen dient, übertragen.

### 5.2.1 Einführung in Standards und historische Ansätze zum Datenaustausch

Durch die wachsende Bedeutung von rechnerunterstützten Anwendungssystemen steigt die Anzahl der Modellierungsumgebungen und zum Einsatz kommender Softwarelösungen. Ein Datenaustausch zwischen den verschiedenen Systemen im Produktentstehungsprozess ist erforderlich, um einen integrierten Verbund ohne Insellösungen zu erhalten. Die Suche nach geeigneten Lösungen für Schnittstellen- oder Datenaustauschformate ist nicht neu. Bereits in den 90er Jahren wurde eine Lösung für die Kopplung der unterschiedlichen CAD-Systeme und deren dreidimensionaler Daten entwickelt [SK97]. Dabei wird zwischen internen und externen Schnittstellen unterschieden. Bei internen Schnittstellen kann im Produktdatenmanagement innerhalb einer CAD-Baugruppe auf zusätzliche Informationen wie Abmessungen und allgemeine Hinweise intuitiv durch die dreidimensionale Darstellung eines Produktes zugegriffen werden [JSM+20]. Bei externen Schnittstellen werden CAD-Systeme mit anderen CAx-Systemen oder Programmsystemen gekoppelt. Etabliert hat sich dabei der Standard STEP (Standard for the Exchange of Product Model Data).

Den Kern von STEP bilden Informationsmodelle [SK97]. Dabei wird nach Krause et al. zugrunde gelegt, dass ein Produktmodell in weitere Submodelle, sogenannte Partialmodelle untergliedert werden kann [KKK+93]. Nach Spur und Krause zählen zu den Partialmodellen unter anderem das Basis-, Material-, Produktstruktur-, Prozess, Geometrie- und Topologiemodell [SK97]. Diese Partialmodelle enthalten eine semantisch abgeschlossene Teilmenge der Produktmodellinformationen über den gesamten Produktlebenszyklus [BKL90, Pic15]. Indem die einzelnen Daten der Partialmodelle miteinander in Beziehung gesetzt und ihre Attribute modellübergreifend verknüpft werden, lässt sich aus der Kombination der Partialmodelle ein integriertes Produktmodell ableiten [BKL90].

Im Vergleich zu anderen Schnittstellenformaten ermöglicht STEP neben dem Geometrietransfer auch den Austausch von Produktdefinitionen [SK97, S.55]. Zum letzteren gehören Topologien, Toleranzen, Materialeigenschaften, Texturen oder Daten aus späteren Lebenszyklen wie Wartung. Das Dateiformat ist durch die ISO 10303 genormt. STEP dient sowohl als Kopplungsbaustein zwischen unterschiedlichen Anwendungssystemen als auch zur Integration von Prozessketten in einer Punkt-zu-Punkt-Kopplung für die Erzeugung eines virtuellen Produkts [SK97, S. 544 ff.]. Die STEP Datei wird sowohl für den Austausch als auch für die Speicherung und Archivierung von Produktdaten eingesetzt mit dem Ziel einer vollständigen und systemunabhängigen Darstellung aller produktbezogenen Daten während des Produktlebenszyklus [KKK+93].

Die formale Sprache, die innerhalb von STEP zur Dokumentation der Partialmodelle verwendet wird, ist die formale Beschreibungssprache EXPRESS [BKL90]. Mit dieser wird die Semantik der Daten formal definiert. Das mit nach dem EXPRESS-Schema

enthaltene Datenmodell besteht ähnlich zur SysML aus Klassen mit Attributen, Referenzen und Regeln. STEP unterteilt sich in verschiedene Teile, die die Standardmechanismen für den Datenaustausch spezifizieren. In diesen werden zum Beispiel die Methoden und Datenaustauschmechanismen beschrieben. Je nach Anwendungskontext werden internationale Standards für branchenspezifische Datenmodelle in sogenannten Applikationsprotokollen (AP) definiert und die Leistungsfähigkeit von STEP angepasst. APs definieren den Inhalt und Umfang der austauschbaren Produktinformationen [SK97, S. 555]. AP233 hat sich dabei als Standard für den Austausch von Systems Engineering Daten etabliert. Es unterstützt unter anderem die Systemmodellierung inklusive deren Struktur und Verhalten [EGZ12]. Ein weiterer Austauschstandard, der im Systems Engineering genutzt wird, ist die eXtensible Markup Language (XML). Seid Einführung Ende der 90er Jahre wird XML zwischen Softwarelösungen für den plattformunabhängigen Austausch von Daten verwendet. Nach Guo et al. [GWCS22] wird AP233 bereits als Datenaustausch für die SysML und die Anbindung an CAD-Tools genutzt. Erste Untersuchungen nutzen STEP AP233 zusammen mit einem XML Metadata Interchange (XMI) Parser für den Datenaustausch zwischen SysML und CAD-Modellen [SI21]. Ausgetauscht werden Anforderungen, physikalische Elemente, die Konfiguration sowie Verhaltens- und Strukturbeschreibungen. Dennoch merken Schumacher und Inkermann an, dass das verwendete AP233 nicht von allen CAD-Autorentool unterstützt wird [SI21].

Als weiteres Datenaustauschformat für eine heterogene Toollandschaft kommt das XML-basierte und objektorientierte Format AutomationML für die Beschreibung von Maschinen, Anlagen und Automatisierungssystemen zum Einsatz [Dra21, KV23]. Einsatzzweck ist der Datenaustausch zwischen verschiedenen Werkzeugen bei der Entwicklung von Produktionssystemen [RKS24]. AutomationML ist „ein offenes, herstellerneutrales und standardisiertes“ Datenformat [GF16]. Das Format verwendet unter anderem für die Abbildung hierarchischer Strukturen das Metamodell CAEX (Computer Aided Engineering eXchange), um damit zum Beispiel das Ressourcenmonitoring zu unterstützen [GF16]. Zusätzlich ermöglicht es Geometrien und Kinematik mit dem Datenformat COLLADA und Verhalten sowie Prozessbeschreibungen oder Steuerungsprogramme durch PLCopen XML abzubilden. AutomationML wandelt sich zunehmend von einem reinen Datenaustauschformat zu einer primären Modellierungssprache für Automatisierungssysteme und ähnelt dabei den Intentionen der SysML [BBL+16]. Erste Untersuchungen befassen sich daher für eine ganzheitliche und effiziente Konfiguration von Montagesystemen mit der Verknüpfung von SysML-Produktmodellen mit den entsprechenden AutomationML-Darstellungen der zugehörigen Montagesysteme [RKS24]. Nach Fay ermöglicht AutomationML als Schnittstellenformat die Verknüpfung aller Planungs- und Betriebsinformationen in einer Industrieanlage, sodass damit der Aufbau eines Digitalen Zwillings möglich ist [Dra21, S. V].

Mit der Einführung von STEP sollte ein integriertes Datenmodell für alle Lebenszyklusaspekte geschaffen werden. Allerdings wurde eine Aufteilung in mehrere APs notwendig. Gründe waren die steigende Komplexität, die verschiedenen Entwicklergruppen und die unterschiedlichen Entwicklungsprozessgeschwindigkeiten. In überschneidenden Bereichen im Entwicklungsprozess führt dies jedoch zu einer Interoperabilität der APs und eine Kopplung der Daten ist nicht immer gewährleistet [EEGE20, EMS05]. Dies unterstreicht nochmals die Notwendigkeit zur Einbindung datentechnischer Partialmodelle in ein übergreifendes Gesamtkonzept für die Beschleunigung der Produktentwicklung.

### 5.2.2 Integrierte Toolanbindungen von SysML-Umgebungen

Für die Interoperabilität zwischen deskriptiven Systemmodellen und funktionalen Modellen – und damit für den Austausch von Daten – bieten SysML-Modellierungsumgebungen bereits verschiedene integrierte Lösungen an. Im Folgenden werden Modellierungsumgebungen wie unter anderem der Cameo Systems Modeler oder PTC Integrity Modeler vorgestellt, die interne Lösungen zur Anbindung externer Tools bereitstellen und hinsichtlich ihrer Anwendbarkeit untersucht werden. Dabei unterscheiden sich die kommerziellen Tools von open-source Lösungen. Erstere bieten nur eine begrenzte Auswahl an direkten Kopplungen an, während bei den anderen über Programmierschnittstellen nahezu jedes Tool angebunden werden kann.

#### Cameo Systems Modeler

Das Cameo Simulation Toolkit umfasst standardmäßig mehrere Skriptsprachen. Diese sind JavaScript, Groovy, Ruby, Python, Beanshell und Built-in Math [No 22a]. Letzteres löst einfache mathematische und logische Formulierungen. Hervorzuheben ist, dass sowohl Python (Jython) als auch Ruby (Jruby) in Cameo in einer Java-Implementierung der jeweiligen Programmiersprache vorliegen. Dadurch ist eine Ausführung der Programme auf der mit Java implementierten Plattform Cameo möglich, verlangsamt aber die Ausführung aufgrund der Übersetzung. Eine weitere implementierte Schnittstelle ist der automatisierte Import und Export von Daten aus Exceltabellen. Dadurch können beispielsweise Informationen für Anforderungen eingelesen und automatisiert Anforderungsinstanzen im CSM erzeugt werden. Zudem ermöglicht das Simulationstoolkit von CSM die Verwendung externer Analysatoren wie Matlab, Maple™, Mathematica und OpenModelica. Sobald die Analyseanwendung auf demselben Rechner installiert ist, kann diese über den implementierten Integrator im Cameo Systems Modeler eingebunden werden. Folglich werden die entsprechenden Sprachen in die Liste der unterstützten Sprachen aufgenommen. Zusätzlich können weitere JSR-223[9] kompatible Sprachen heruntergeladen und installiert werden [No 22a].

Die Anwendung der Sprache erfolgt in ausführbaren Informationsabläufen wie dem Aktivitätsdiagramm. Abbildung 5.5 zeigt die Integration von Matlab. Mit Hilfe des Elements Opaque-Action (1) werden die externen Umgebungen angesteuert. Dieses Element verfügt über eine spezifische Semantik und nutzt alle hinterlegten Sprachen. Wird beispielsweise Matlab ausgewählt, so verwendet das Cameo Simulation Toolkit die Matlab-Syntax, startet die Software im Hintergrund und führt die folgenden Programmfunktionen in Matlab (2) aus. Dadurch können Funktionen oder Skripte, die mit Matlab bereits erzeugt wurden, ebenfalls in Cameo für die Berechnung genutzt und die Ergebnisse für weitere Analysen innerhalb der SysML-Umgebung verwendet werden.

[9] Die Standard-Skripting-API für Sprachen im Java Quellcode.

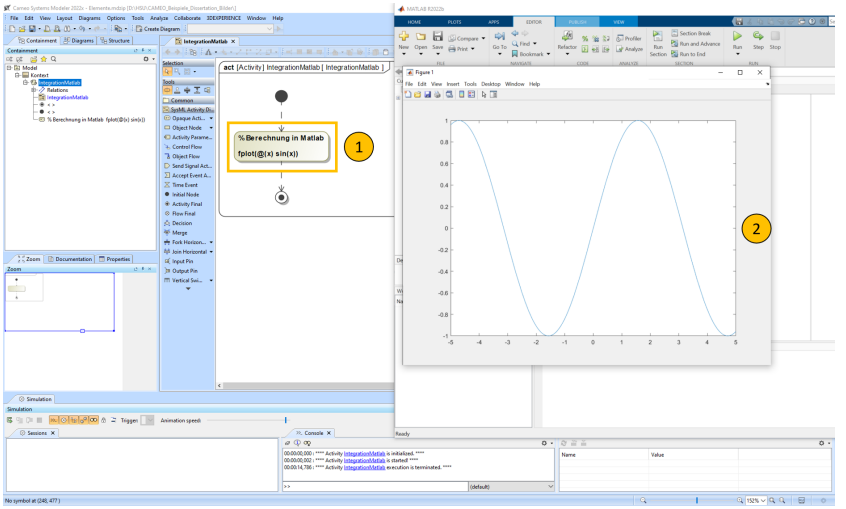



Abbildung 5.5: Integration von Matlab (2) ins Cameo Simulation Toolkit mit Aktivitätsdiagramm und Element Opaque Action (1).

### PTC Integrity Modeler, Papyrus, Capella und Enterprise Architect

Weitere SysML-Modellierungstools bieten ebenfalls integrierte Schnittstellen zu externen Modellumgebungen an. Für Simulationsumgebungen wie Modelica bietet Papyrus den Import der Modelica Standard Library an, um Modelle direkt in Papyrus zu modellieren [Wös15]. Der PTC Integrity Modeler bietet SySim als Ansatz, um mit Visual Studio grafische Simulationsmodelle anzufertigen [Wös15]. Bei Enterprise Architect wird die SysML durch die SysPhS Spezifikation (SysML Extension for Physical Interaction and Signal Flow Simulation) erweitert, sodass SysML-Modelle direkt in Simulationsplattformen wie Modelica und Matlab/Simulink/Simscape übersetzt werden können. Die direkte Ansteuerung von mathematischen Funktionen erfolgt beispielsweise über die Programmierschnittstelle von Matlab in laufenden Simulationen, sodass konsistente Modelle erzeugt werden. Capella bietet die Integration mit Anforderungsmanagement-Software wie IBM Rational DOORS (Dynamic Object Oriented Requirements System) oder Lifecycle Management Lösungen wie Polarion von Siemens an. Aufgrund der open-source Oberfläche von Capella können weitere Softwarelösungen über das Interface, zum Beispiel Matlabfunktionen durch eingebetteten Python-Code in Funktionselementen, direkt ins Capella-Modell integriert werden.

### Anwendungsbeispiele

Colletti et al. zeigen auf, dass viele Forschungen SysML als zentralen Speicher zur Verwaltung von Anforderungen und Definition von Studien verwenden. Die Durchführung der Studien erfolgt allerdings in externen Simulationsumgebungen, die ihre Ergebnisse wiederum an das SysML Modell zurücksenden. Für die Simulation

der Systemdynamik können die Modelica-Sprachkonstrukte in die SysML über das SysML4Modelica Profil eingebunden werden, was die Integration von Simulationsexperimenten ermöglicht [CQN+22, S. 296]. Weitere Beispiele für interne Toolanbindungen sind die Nutzung der integrierten Matlab-Schnittstelle für einerseits Lastüberprüfungen auf die Hauptwelle einer Windenergieanlage [BHZ+22] und andererseits für die Analyse und Optimierung von Systemarchitekturen für die elektrische Energieverteilung in der Flugzeugkabine [GFBN23] sowie der Datenexport aus Exceltabellen für die Kalkulation der hydrodynamischen Abmessungen einer Pumpe [HJZ+21].

### 5.2.3 Externe Modellkopplung mit SysML-Modellen über Schnittstellenformate

Um externe Modelle mit SysML-Modellen zu koppeln, wurden bereits einige Schnittstellentechnologien untersucht. Im Folgenden werden diese vorgestellt und sind darüber hinaus in Tabelle 5.2 zusammengefasst. Fokussiert bei der Auswahl der Ansätze werden die Entwicklungsumgebungen und die Art des Datentransfers zwischen den Disziplinen. Schwerpunkt liegt auf dem Cameo Systems Modeler als SysML-Umgebung, da diese ebenfalls als Entwicklungsumgebung in dieser Forschungsarbeit untersucht wird.

**XML-Dokument**

Brahmi et al. untersuchen einen Ansatz zur Kopplung in den Bereichen MBSE und CAD. Anhand einer Fallstudie eines mechatronischen Industrieprodukts wie dem Fahrrad wird die kollaborative Zusammenarbeit zwischen dem Systemingenieur und dem Designer verbessert [BBH+21]. Mit dem Cameo Systems Modeler werden die geometrischen Anforderungen der Standardkomponenten sowie die Zusammenbauarchitektur durch den Systemingenieur modelliert. Der Designer wiederum analysiert und konstruiert das 3D-Design der vorgeschlagenen Lösung. Der Datenaustausch zwischen den zwei Disziplinen erfolgt mit Hilfe einer XML-Datei.

Ein weiteres Beispiel für die Nutzung von XML-Dateien als Austauschformat nutzen Kotronis et al. [KNK+20]. Sie verknüpfen SysML-Modelle mit externen Simulationsmodellen für die Überprüfung von Qualitätskriterien für Schienenverkehrssysteme. Im Fokus steht die Betrachtung des Passagierkomforts. Die SysML-Umgebung dient als Metamodell für die Erstellung des Schienenverkehrssystems. Währenddessen erfolgt die Simulation für die Performance-Analyse unter Anwendung der ereignisdiskreten Abstraktion (Discrete Event System Specification, DEVS) im Hintergrund. Die Ergebnisse der Simulation werden anschließend in XML und CSV (Comma-Seperated Values) Format-fähigen Dokumenten gespeichert. Letztlich werden die Ergebnisse in Form von Entitäten und deren Attribute an das Metamodell in SysML zurück übergeben.

Wösle untersuchte in seiner Forschungsarbeit mehrere Schnittstellentechnologien und ermittelte XML als geeignetste Schnittstelle zum Austausch von Daten [Wös15, S. 31]. Vorteile sieht Wösle in dem standardisierten strukturellen Aufbau sowie der Speicherung von Metadaten und Inhalten des Systemmodells [Wös15]. Am Beispiel eines chaotischen Pendels zeigt er die Kopplung zwischen dem Systemmodell in Enterprise Architect und der Simulation des Systems mit dem kommerziellen Simulationstool Dymola auf Basis der Sprache Modelica.

Dmitriev et al. verknüpften für die Entwicklung von sicherheitskritischer Software für die urbane Luftmobilität Designmodelle und Testfälle aus der numerischen Plattform Matlab/Simulink mit Anforderungen aus dem Tool Polarion[10] [DZS+20]. Dabei verifizieren die Ergebnisse der modellbasierten Testfälle die Anforderungen in Polarion. Die Übertragung der Daten erfolgt über das Ein- und Auslesen von XML-Dateien.

Tabelle 5.2: Übersicht über Schnittstellenformate zwischen SysML-Modellen und externen Modellen.

| Kopplungsart | Modellkopplung | Anwendung | Quelle |
|---|---|---|---|
| XML-Dokument oder CSV-Dateiformat | Cameo Systems Modeler und CAD | Kollaborative Zusammenarbeit zwischen Konstrukteur und Systemingenieur bei der Auslegung mechatronischer Industrieprodukte. | [BBH+21] |
| | MagicDraw und DEVS | Überprüfung von Qualitätskriterien wie Passagierkomfort, Sicherheit oder Umweltauswirkungen für Schienenverkehrssysteme. | [KNK+20] |
| | Enterprise Architect und Modelica | Kopplung zwischen Modellierung und physikalischer Simulation eines Systems am Beispiel eines chaotischen Pendels. | [Wös15] |
| (Matlab-) Datenbank | Cameo Systems Modeler, Matlab/Simulink, Simcenter Amesin und CATIA | Verknüpfung der Systemarchitektur mit Simulationsmethoden, um physikalisches Produktverhalten einer elektrisch-mechanischen Kühlmittelpumpe vorherzusagen. | [HJZ+21] |
| Internetprotokoll | Cameo Systems Modeler und Unity | Transfer von Positions-, Orientierungs- und Skalierungsdaten von Objekten aus der SysML zur Visualisierung in einer virtuellen Realitäts-Umgebung am Beispiel eines Staubsaugers. | [Mah21] |
| Visual Basic | PTC Integrity Modeler und Matlab | Vorauslegung und Dimensionierung von Passagierflugzeugen basierend auf der Missionsdefinition. | [FD22] |
| Pure::variants | Cameo Systems Modeler, Ansys, MathWorks und IBM Doors | Ganzheitliches Variantenmanagement und automatisierter Datenaustausch in allen Entwicklungsphasen. | [PTC23] |

#### Datenbanken

Datenbanken als Adapter zwischen verschiedenen Modellierungsumgebungen werden von Köpfner et al. untersucht [HJZ+21]. Sie verknüpfen die Systemarchitektur aus der SysML mit Simulationsmethoden in Simulink und Simcenter Amesim[11], um physikalisches Produktverhalten einer elektrisch-mechanischen Kühlmittelpumpe vorher-

[10] Webbasierte Lifecycle Management Applikation von Siemens Industry Software GmbH.

[11] Systemsimulationsplattform von Siemens, um Leistung von Systemen virtuell zu bewerten und optimieren.

zusagen. Ein parametrisches CAD-Modell visualisiert anschließend die Designergebnisse basierend auf den berechneten geometrischen Parametern. In einer externen Datenbank werden die Parameter der Modellierung gespeichert und mit den Parametern im CSM-Modell verknüpft. Werden während der Simulation oder Berechnung neue Parameter erzeugt, werden diese an die Datenbank übergeben und können von den disziplinspezifischen Tools für weitere Berechnungen ausgelesen werden. Die Parameterübertragung erfolgt im CSM mit dem Aktivitätsdiagramm. In diesem Anwendungsfall werden Matlab-Datendateien verwendet, das Konzept ist allerdings auf andere Datenbanktypen übertragbar [HJZ+21].

**Internetprotokolle TCP und UDP**

Mahboob untersucht den Transfer von Positions-, Orientierungs- und Skalierungsdaten von Objekten aus den dynamischen Verhaltensdiagrammen in SysML mit dem Cameo Systems Modeler zur Visualisierung in einer VR-Umgebung wie Unity [Mah21]. Eingaben, die vom Nutzer des VR-Systems in der virtuellen Welt getätigt wurden, werden von einem Interaktionsgerät erfasst und an das Produktmodell in SysML übermittelt [MHW+19]. Ziele der Forschung sind die frühzeitige Integration von virtueller Realität im Entwicklungsprozess und die automatisierte Erstellung von VR-Modellen basierend auf dem Verhalten eines Produktes durch wiederverwendbare Submodelle, um den Initialisierungsaufwand zu reduzieren. Transport der Daten erfolgt über das Übertragungssteuerungprotokoll TCP (Transmission Control Protocol) und verbindungslose Netzwerkprotokoll UDP (User Data Protocol) [Mah21].

**Visual Basic**

Friedrichs-Dachale untersucht die Kopplung zwischen dem PTC Integrity Modeler und der Simulationssoftware Matlab. Anwendungsfall ist die Vorauslegung und Dimensionierung von Passagierflugzeugen. Das Flugzeugmodell in der SysML enthält die Top-Level Anforderungen, die Flugzeugarchitektur sowie das Missionsprofil. Das numerische Modell in Matlab enthält Funktionen für eine vorläufige Auslegung des Flugzeugs. Gekoppelt sind die beiden Modelle über einen reinen Parameteraustausch. Dieser Datentransfer zwischen den beiden Modellen erfolgt über Visual Basic, einer objektorientierten Programmiersprache, die für die Erstellung von grafischen Nutzeroberflächen (Graphical User Interface, GUI) verwendet wird. Das Visual Basic Programm interagiert über die API vom PTC Integrity Modeler und über den Automation Server von Matlab mit dessen Daten [FD22, S. 95]. Durch seinen Ansatz ist kein manueller Datentransfer mehr notwendig [FD22, S. 116].

**Software-Lösung pure::variants**

Das Unternehmen PTC bietet die Softwarelösung pure::variants für das Management von Produktlinien in allen Entwicklungsphasen an [PTC23]. Die Softwarelösung kann in bereits bestehende Prozessketten und Tools integriert werden. Das ganzheitliche Variantenmanagement erfolgt über einen automatisierten Datenaustausch zwischen verschiedener Software. Grundlage bildet ein Feature-Modell (Merkmalbaum), welches mit der pure::variants Software erstellt wird [PTC23]. In diesem sind alle Produkteigenschaften, Varianten (Features) sowie Querverbindungen in Form von Regeln abgebildet. Das dadurch entstehende 150% Modell ermöglicht sowohl eine topdown als auch button-up Modellierung. Die Konfiguration einer Variante entsteht durch die Auswahl von Features im Feature-Modell. Anschließend werden die ausgewählten Merkmale und deren Daten in die anderen Disziplinen verlinkt. Dafür stellt pu-

re::variants Interfaces und Oberflächen für andere Tools bereit, über die dann von überall auf die Daten (Parameter, Variablen) im Feature-Modell zugegriffen werden kann. Das Feature-Modell beinhaltet alle Parameter und Daten für die Definition einer Variante und wird als single-source-of-truth verwendet. Dabei erhalten die einzelnen Disziplinen nur so viel Zugang zu den Parametern, wie für die Entscheidungsfindung oder die Modellierung innerhalb der eigenen Umgebung benötigt wird.

Die Toollandschaft, die pure::variants unterstützt, umfasst unter anderem Dassault Systemes, MathWorks, Ansys und IBM. Zusätzlich bietet die Software eine Anbindung an OSLC oder einen zentralen Server. Einige Unternehmen aus den Bereichen Luftfahrt, Automobil und Transport nutzen bereits diese Lösung für das Management ihrer Produktlinien [PTC23].

### 5.2.4 Herausforderungen und Grenzen

Aus den vorgestellten Ansätzen zur Kopplung von SysML-Modellen mit heterogenen, externen Modellen lassen sich im Rahmen der in dieser Arbeit durchgeführten Descriptive Study I spezifische Herausforderungen ableiten, die für eine erfolgreiche Implementierung adressiert werden müssen. Dabei müssen verschiedene Aspekte berücksichtigt werden, um potenzielle Integrationsprobleme frühzeitig zu erkennen und zu minimieren. Diese sind:

- Modellkonsistenz,
- Datenaustausch,
- Modellgröße und Übersichtlichkeit,
- Manuelle Anpassung.

Ein wichtiger Aspekt wie bei der internen Kopplung ist die Sicherstellung der Modellkonsistenz. Bei heterogenen Modellen werden verschiedenartige Modellelemente miteinander verknüpft, sodass die Konsistenz nur durch Strukturmerkmale gewährleistet werden kann [SI21].

Bei Austauschformaten wie der XML-Datei werden nur reine Parameter zwischen den Modellen ausgetauscht [FD22, S. 116]. Abhängigkeiten und Verknüpfungen der Daten untereinander werden nicht abgebildet. Stattdessen muss in jedem Modell eine neue Modellierung und Zuordnung der Daten erfolgen.

Für die integrierte Toolanbindung werden unter anderem Funktionselemente mit Programmcode in Diagrammen eingesetzt, um externe Modellierungsumgebungen aufzurufen und zu koppeln. Dabei besteht der Nachteil der Unübersichtlichkeit. Werden viele Anfragen an externe Tools innerhalb einer Ausführung gestellt oder viele Parameter übergeben, ist das Diagramm (z.B. Aktivitätsdiagramme beim Cameo Systems Modeler) mit Blöcken überfüllt [HJZ+21]. Die interne Komplexität der Darstellung steigt, wodurch das Systemverständnis für den Menschen sinkt. Mit zunehmender Modellgröße steigt ebenso die Rechenintensität.

Eine weitere Einschränkung ist die sehr anwendungsspezifische Herangehensweise bei der Modellierung und Kopplung. Bislang existiert kein einheitlicher Standard für den Datenaustausch und die Kopplung heterogener Modelle. Zudem müssen Funktionen zum Auslesen von Parametern jedes Mal neu und manuell geschrieben werden, sobald Parameter hinzukommen, deren Bezeichnung sich ändert oder ganz wegfallen [HJZ+21].

### Schlussfolgerungen

In der Literatur finden sich einige Ansätze zur Verknüpfung verschiedener Disziplinen, die im vorherigen Abschnitt vorgestellt sind. Gleichartig zu den Kriterien aus Abschnitt 5.1.4 werden die Ansätze nach vier Anforderungen bewertet. Eine Übersicht der untersuchten Ansätze inklusive dem Erfüllungsgrad der Anforderungen ist in Tabelle 5.3 gegeben. Zusammenfassend zeigt sich, dass die Kopplung von deskriptiven und analytischen Modellen mit den untersuchten Ansätzen realisierbar ist und die Lösungsansätze im Vergleich zu der Kopplung mehrerer deskriptiver Modelle bei den meisten Kategorien ähnliche positive Resultate erzielen.

Tabelle 5.3: Überblick der Anforderungserfüllung durch Ansätze zur Kopplung von SysML- und heterogenen Modellen.

| | ***Anforderungen*** | | | |
|---|---|---|---|---|
| | **A.4** Multidisziplinäre Zusammenarbeit | | **A.5** Externe Zusammenarbeit | |
| **Ansätze** | **A.4.1** Einfluss auf Modellgröße | **A.4.2** Modellpflege | **A.5.1** Lizenz-abhängigkeit | **A.5.2** Automatisierung |
| CSM* integriert | ● | ◕ | ○ | ● |
| XML | ● | ◕ | ◕ | ● |
| Datenbanken | ◕ | ◑ | ◕ | ● |
| Internet-protokoll | ◕ | ◑ | ● | ● |
| Visual Basic | ◕ | ◑ | ◔ | ● |
| pure::variants | ● | ◕ | ○ | ● |
| Bewertungsskala: | sehr hoch ● | hoch ◕ | niedrig ◑ | sehr niedrig ◔ | nicht erfüllt ○ |
| *CSM = Cameo Systems Modeler | | | | |

Lediglich beim Kriterium der Lizenzabhängigkeit schneiden die integrierte Schnittstelle beim Cameo Systems Modeler sowie die Software-Lösung pure::variants und Visual Basic aufgrund der Lizenzbindung schlechter ab. Zudem zeigten die Lösungsansätze bei den Datenbanken, dem Internetprotokoll und Visual Basic einen erhöhten Aufwand bei der Modellpflege. Gründe hierfür sind die manuelle Anpassung der

Schnittstellen bei Änderungen im SysML-Modell, keine Skalierbarkeit bei der Speicherung von Daten oder eine Überfüllung der SysML-Diagramme mit Import- und Export-Funktionen.

Hervorzuheben ist, dass eine Schnittstellenkopplung zwischen Tools mit Hilfe von einer XML-Datei als Kommunikationsmittel als sehr positiv bewertet wird. Dabei dient diese XML-Datei als Adapter, indem sie alle zu teilenden relevanten Parameter enthält. Dadurch wird ein Datentransfer über Toolgrenzen hinweg ermöglicht, da diese Schnittstelle weit verbreitet ist und von vielen Entwicklungsumgebungen bereitgestellt wird. Im Vergleich zu den anderen Ansätzen ist eine einfache Erweiterbarkeit durch den Anschluss einer Vielzahl an Entwicklungsumgebungen und damit mit deren Modellen gegeben.

## 5.3 Kapitelfazit

Die Literatur zeigt, dass für die Kopplung von zwei oder mehreren SysML-Modellen kaum Untersuchungen vorliegen. Stattdessen werden Systeme häufig mit einem SysML-Modell abgebildet. Durch integrierte Lösungen seitens der SysML-Modellierungsumgebung können Partialmodelle erzeugt und über einen Server oder freigegebene Pakete miteinander gekoppelt werden. Allerdings führt dies mit zunehmender Systemkomplexität wieder zu einem großen Einzelgesamtmodell. Für die Zusammenarbeit mehrerer Experten in einer Modellumgebung ist dies weiterhin mit hohen Abhängigkeiten, einer Vielzahl an Modellelementverflechtungen und einer aufwendigen Modellpflege verbunden. Zudem sind die Lösungen Tool-spezifisch oder kostenpflichtig, was die Zusammenarbeit mit externen Experten erschwert oder aufgrund von Lizenz- und IT-Grenzen unmöglich macht.

Folglich stellen diese Ansätze für die unternehmensübergreifende Zusammenarbeit keine Lösung bereit. Darüber hinaus eignen sich die betrachteten Schnittstellentechnologien nicht zur Anbindung multidisziplinärer Modelle. Daher wurden weitere Methoden und Ansätze für die Verknüpfung von heterogenen Modellen verglichen. Die Softwarelandschaft ist vielfältig und einige Ansätze für Schnittstellenlösungen wie XML-Dateien oder Internetprotokolle haben sich bereits etabliert. Zusätzlich stellen Modellierungsumgebungen wie der Cameo Systems Modeler für die Anbindung von SysML-Modellen bereits einige integrierte Lösungen zur Kopplung an externe Tools bereit.

Insgesamt zeigt sich, dass keiner der untersuchten Lösungsansätze die Anforderungen **A.1** bis **A.5** vollständig erfüllt. Stattdessen existieren viele Einzellösungen, die sehr anwendungsspezifisch entwickelt wurden. Für die ganzheitliche Betrachtung eines Systems und Entwicklung werden aber viele verschiedene Modellierungsumgebungen genutzt, die alle miteinander gekoppelt werden müssen. Gleichzeitig zeigt sich in der industriellen Praxis mit unternehmensübergreifenden Partnern, dass bei externen Experten bereits etablierte Modellierungsumgebungen zum Einsatz kommen, die nicht ohne erheblichen Mehraufwand verändert oder ersetzt werden können. Zudem erfordert die beschleunigte Produktentwicklung eine hohe Anpassungsfähigkeit der Modelllandschaft, um einerseits neue Innovationen zeitnah in bestehende Varianten integrieren zu können und andererseits flexibel auf wechselnde externe Partner und

deren heterogene Modellierungsansätze reagieren zu können. Daher ist eine Methodik erforderlich, die modular ist und die Zusammenarbeit unterschiedlicher (externer) Experten unterstützt. Zur Beantwortung der in Kapitel 3.3 ermittelten Forschungsfrage resultiert somit ein Bedarf an einer kollaborativen Methode, mit der eine Modularität des Gesamtsystems zur Untersuchung von Varianten unter Einbindung multidisziplinärer Analysen ermöglicht werden kann.

# 6. Handlungsbedarf und Lösungsansatz

In diesem Kapitel wird der bisherige Stand der Erkenntnisgewinnung aus der Descriptive Study I genutzt, um den Handlungsbedarf an die zu entwickelnde Unterstützung abzuleiten. Zusätzlich werden die Ergebnisse der Anforderungsuntersuchungen zu bereits bestehenden Entwicklungen in der Variantenmodellierung und der Modellkopplung für die Konkretisierung der Forschungsfrage genutzt. Daraus ergibt sich der Lösungsansatz, der im Folgenden vorgestellt wird. Damit stellt dieses Kapitel die Prescriptive Study der DRM für die Entwicklung der Unterstützung dar.

## 6.1 Zusammenfassung bisheriger Erkenntnisse

Die Ergebnisse der Literaturrecherche aus den vorherigen beiden Kapiteln zeigen, dass ein Bedarf an einer **kollaborativen Methode zur Kopplung multidisziplinärer Modelle als auch zur Variantenmodellierung** besteht. In Abbildung 6.1 sind die Ergebnisse der Recherche zusammengetragen. Dabei wurden die Ansätze nach den erarbeiteten Anforderungen aus Abschnitt 3.3 bewertet.

Die Literaturanalyse in Abschnitt 4.4.1 ergab, dass bereits Ansätze für die Modellierung von Varianten mit der SysML existieren. Dabei wird ein Einzelmodell aufgebaut, in dem alle Varianten gemeinsam modelliert werden. Hierbei zeigt sich allerdings, dass nachträgliche Änderungen aufgrund der starren Kopplung der Varianten im Modell nicht realisierbar sind. Zudem ist die Erweiterbarkeit um neue Varianten ab einer gewissen Größe des Einzelmodells und dessen starren Aufbaus nicht mehr praktikabel umsetzbar. Darüber hinaus wird aufgrund der steigenden Vielzahl an Modellelementen die induzierte Modellkomplexität erhöht, wodurch Zusammenhänge für den Menschen bei der Modellierung im Modell nicht mehr einfach nachvollzogen werden können und eine Modellpflege nicht mehr durchführbar ist.

Inwieweit eine Modellierung von Varianten mit diesen Ansätzen in der SysML in der disziplinären Zusammenarbeit mit mehreren Experten umgesetzt werden kann, wurde in Abschnitt 5.1.4 untersucht. Die Ergebnisse des Stands der Technik zeigen, dass durch die Verwendung eines SysML-Einzelmodells die Anforderungen an die Wartbarkeit dieses Modell nicht in der Praxis erfüllbar sind und eine Zusammenarbeit mit externen Partnern im industriellen Kontext aufgrund von IT-Grenzen oder Lizenzgebühren nur mit großem Aufwand möglich ist. Zudem haben die untersuchten Ansätze einen erheblichen Einfluss auf die Modellgröße, wodurch wiederum die induzierte Modellkomplexität steigt.

Darüber hinaus müssen für eine ganzheitliche Betrachtung der Systemauslegung und um eine bessere Entscheidung treffen zu können, die SysML-Modelle wiederum an weitere Disziplinmodelle angeschlossen werden. Folglich wurden in Abschnitt 5.2.4 Lösungsansätze für die Kopplung heterogener Modelle mit SysML-Modellen zusammengetragen. Die meisten Ansätze erfüllen die hier gestellten Anforderungen an die kollaborative und multidisziplinäre Modellierung nur eingeschränkt. Überwiegend wurden diese für die Kopplung von nur zwei Disziplinen sowie unternehmensintern eingesetzt.

**Variantenmodellierung**

| Lösungsansätze | Anforderungen | | |
|---|---|---|---|
| | **A.1** Induzierte Modellkomplexität | **A.2** Erweiterbarkeit | **A.3** Wiederverwendbarkeit |
| VAMOS | ◑ | ◔ | ◑ |
| FODA | ◑ | ◔ | ◕ |
| 150 % Modell | ◔ | ◔ | ◕ |

**SysML zu SysML Kopplung**

| Lösungsansätze | Anforderungen | | | |
|---|---|---|---|---|
| | **A.4** Multidisziplinäre Zusammenarbeit | | **A.5** Externe Zusammenarbeit | |
| | **A.4.1** Einfluss auf Modellgröße | **A.4.2** Modellpflege | **A.5.1** Lizenzabhängigkeit | **A.5.2** Automatisierung |
| Share Package Funktion | ○ | ◔ | ○ | ● |
| Server / Team Cloud | ○ | ◔ | ○ | ● |
| OSLC | ◑ | ◑ | ◕ | ● |

**Heterogene Modellkopplung**

| Lösungsansätze | Anforderungen | | | |
|---|---|---|---|---|
| | **A.4** Multidisziplinäre Zusammenarbeit | | **A.5** Externe Zusammenarbeit | |
| | **A.4.1** Einfluss auf Modellgröße | **A.4.2** Modellpflege | **A.5.1** Lizenzabhängigkeit | **A.5.2** Automatisierung |
| CSM* integriert | ● | ◕ | ○ | ● |
| XML | ● | ◕ | ◕ | ● |
| Datenbanken | ◕ | ◑ | ◕ | ● |
| Internetprotokoll | ◕ | ◑ | ● | ● |
| Visual Basic | ◕ | ◑ | ◔ | ● |
| pure::variants | ● | ◕ | ○ | ● |

Bewertungsskala: sehr hoch ● hoch ◕ niedrig ◑ sehr niedrig ◔ nicht erfüllt ○ *CSM = Cameo Systems Modeler

Abbildung 6.1: Zusammenfassung der Bewertung aktueller Ansätze.

Das Ergebnis zeigt, dass die bisherigen Methoden und Ansätze nicht alle gestellten Anforderungen und Randbedingungen erfüllen, um den Herausforderungen einer wachsenden Variantenvielfalt und der damit einhergehenden induzierten Komplexität in den Modellen zu begegnen. Aus diesem Grund liefert diese Arbeit einen neuen methodischen Ansatz als Lösung auf die Forschungsfrage, der zusätzlich die Anbindung

weiterer heterogener Modelle für ganzheitliche Analysen und die Zusammenarbeit mit mehreren Experten ermöglicht.

Durch eine verbesserte Zusammenarbeit und Kommunikation der vielen (Disziplin-) Experten wird sich eine Beschleunigung der Produktentwicklung versprochen. Daraus abgeleitet ergeben sich zwei Schlüsselfaktoren. Schlüsselfaktoren sind als vielversprechende Ansatzpunkte anzusehen, die durch die zu entwickelnde methodische Unterstützung positiv beeinflusst werden können. Abbildung 6.2 zeigt das Impact Model in Anlehnung an die DRM (erweitertes Reference Model aus Abschnitt 2.1) mit den identifizierten Schlüsselfaktoren und die Anknüpfung der Aspekte der Unterstützung zu diesen. Die entwickelte Unterstützung verfolgt das Ziel, den messbaren Erfolgsfaktor *Entwicklungszeit* positiv zu beeinflussen, indem Änderungen innerhalb der Systemmodelle durch Anpassungen im Rahmen der Variantenmodellierung schneller und einfacher durchgeführt werden können.

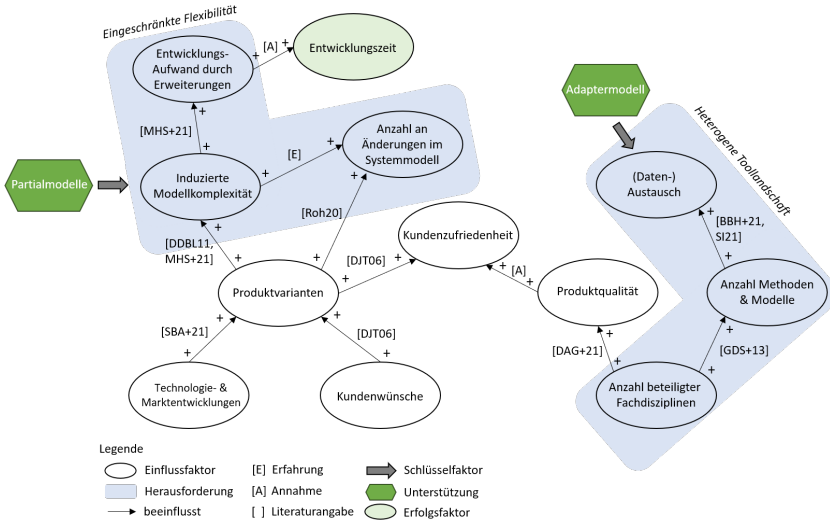



Abbildung 6.2: Impact Model dieser Arbeit für den Variantenauslegungsprozess in Anlehnung an die DRM von Blessing und Chakrabarti [BC09].

## 6.2 Vorstellung des methodischen Lösungsansatzes

Zusammengefasst bilden die erarbeiteten Anforderungen die Grundlage und Randbedingungen für die in dieser Arbeit entwickelte Methode. Im Folgenden wird eine Einführung für einen methodischen Lösungsansatz föderierter Partialmodelle gegeben, um die genannten Herausforderungen zu adressieren. Zudem werden die zugrundeliegenden Annahmen und Hypothesen vorgestellt.

In Abschnitt 4.3 und A.1 wurden der Customizingprozess sowie der Entwicklungsprozess in der Luftfahrt vorgestellt. Zuerst entwickelt der OEM eine Standardkon-

figuration eines Flugzeugmusters (Baseline). Diese stellt die vertragliche Konfigurationsgrundlage dar und umfasst Standardkonfigurationen, wie z.B. eine maximale Economy-Bestuhlung und eine Zwei-Klassen-Version, die in der Aircraft Characteristics [Air24] festgehalten werden [NM09]. Kundenindividuelle Anpassungen erfolgen erst nach Auftragseingang im Rahmen des Customizing-Prozesses (Abbildung 6.3) [For22]. Dabei werden, ausgehend von der Baseline, die gewünschten Änderungen spezifiziert und in konkrete Flugzeugkonfigurationen überführt, die als sogenannte Manufacturer Serial Numbers (MSN) umgesetzt werden. Somit entstehen auf Basis der Baseline kundenbezogene Produktderivate [NM09]. Ziel ist es, zukünftig relevante Varianten frühzeitig im Entwicklungsprozess mitzudenken, um sowohl das Customizing zu beschleunigen als auch die Neuentwicklung effizienter zu gestalten.

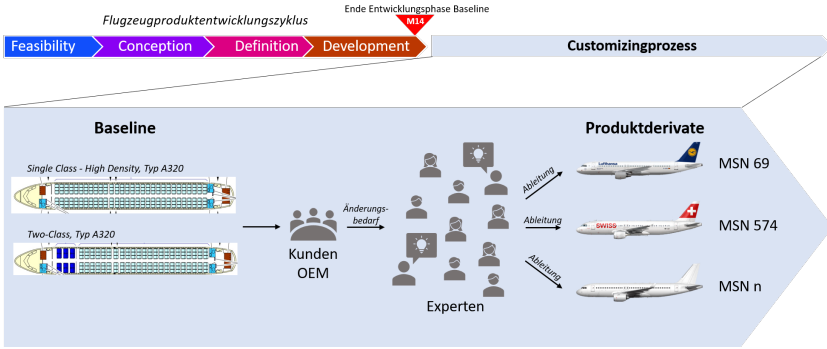



Abbildung 6.3: Nachgelagerter Customizingprozess in der Flugzeugentwicklung.

Wie in Kapitel 3 dargestellt, basieren die zum Einsatz kommenden heterogenen Modelle auf unterschiedlichen Abstraktionsebenen, die aufgrund ihrer domänenspezifischen Ausprägung keine durchgängige Interoperabilität und standardisierte Schnittstellen zwischen den Modellumgebungen gewährleisten. Abbildung 6.4 veranschaulicht beispielhaft die dabei eingesetzten Modellierungs- und Datenformate im Vorentwurfsprozess.

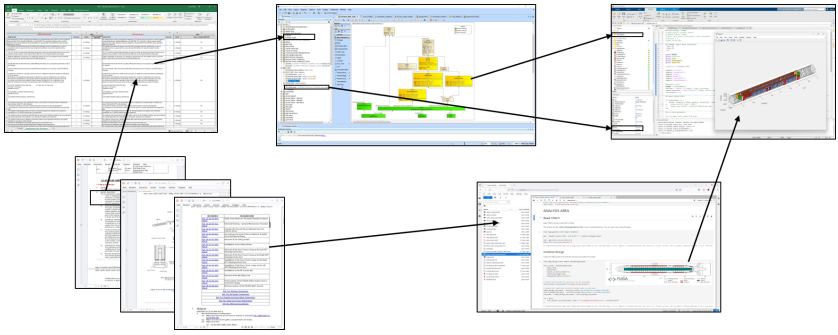


Abbildung 6.4: Darstellung verschiedener Modellierungsumgebungen und Datenformate zur systemübergreifenden Variantenauslegung im Flugzeugvorentwurfsprozess.

Der Customizingprozess umfasst daher viele manuelle Tätigkeiten, insbesondere im Hinblick auf den Informationsaustausch zwischen Fachexperten und deren domänenspezifischen Modellen, wodurch Änderungsprozesse häufig mehrere Monate in Anspruch nehmen [BBH+18]. Zusätzlich ist der Prozess wenig digitalisiert und stark sequentiell sowie in hohem Maße von implizitem Expertenwissen abhängig. Durch die fehlenden digitalen Schnittstellen sind die Wechselwirkungen zwischen Subsystemen nicht ausreichend abgebildet (Abbildung 6.5). In der Literatur finden sich Ansätze, um mehrere Experten kollaborativ zur Modellierung von Varianten zusammenzubringen. Das Ergebnis ist dabei häufig ein gemeinsames Einzelmodell, in dem alle Varianten abgebildet werden. In der praktischen Umsetzung zeigt sich allerdings, dass ein solches Modell schnell unübersichtlich und stark komplex wird. Dies schränkt sowohl die Flexibilität in der Variantenmodellierung als auch die Art der Kommunikation zwischen den Experten erheblich ein, da man an eine bestimmte Modellierungsmethode und einen festen Modelltyp gebunden ist. Die praktische Anwendbarkeit solcher Ansätze ist daher begrenzt.

Aus diesem Grund wird in dieser Arbeit ein neuer Lösungsansatz entwickelt, der eine verbesserte Kommunikation zwischen den Experten ermöglicht und die Entwicklung sowie Analyse von Varianten beschleunigt. Dabei stellt jeder Experte ein eigenes, interoperables Modell bereit, welches mit den Modellen der anderen Experten kommuniziert und Informationen austauscht mit dem Ziel, die erarbeiteten Anforderungen aus den Kapiteln 3 bis 5 zu adressieren.

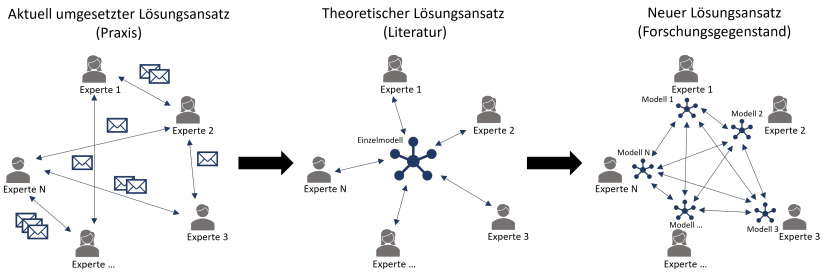



Abbildung 6.5: Darstellung der Ist-Situation in der Praxis sowie bestehender Ansätze aus der Literatur für die kollaborative Gesamtsystemmodellierung und Ableitung des entwickelten, neuen Lösungsansatzes.

**Lösungskonzept: Variantenbildung und Customizing mittels föderierter Partialmodelle**

Ziel dieser Forschungsarbeit ist daher die Entwicklung eines Lösungsansatzes, der verteiltes Arbeiten in komplexen Entwicklungskontexten ermöglicht und dabei auf industrielle Umsetzbarkeit sowie Anwendbarkeit ausgerichtet ist. Wie bereits ausführlich dargelegt, kann ein Gesamtsystem in verschiedene Subsysteme und Komponenten zerlegt werden. Für die Auslegung einer spezifischen Variante dieses Gesamtsystems werden die verschiedenen Fachexperten der jeweiligen Subsysteme interdisziplinär zusammengebracht. Abbildung 6.6 veranschaulicht exemplarisch die hier-

archische Struktur eines Gesamtsystems, das in Subsysteme und diese wiederum in einzelne Komponenten untergliedert ist.

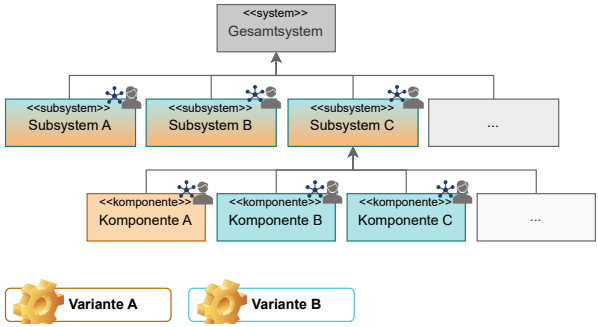



Abbildung 6.6: Zuordnung der Subsysteme zu Experten und deren Modellen sowie beispielhafte Auswahl der Subsysteme für Varianten des Gesamtsystems. Orange hinterlegte Kacheln zeigen die erforderlichen, miteinander zu koppelnden Subsysteme für die Auslegung der Variante A und blaue Kacheln für die Variante B.

Jedem Subsystem und jeder Komponente sind dabei spezifische Fachdisziplinen zugeordnet, die jeweils domänenspezifische Modelle bereitstellen, um die zugehörigen Teilsysteme adäquat abzubilden. Im Gegensatz zu bestehenden Modellierungsansätzen wird bei dem vorgestellten Lösungskonzept nicht in einem gemeinsamen, zentralen Gesamtmodell gearbeitet. Stattdessen verbleibt jedes Subsystem im Verantwortungsbereich der jeweiligen Experten. Dies ist insbesondere der Heterogenität der eingesetzten Modellierungsumgebungen geschuldet und entspricht den Anforderungen an eine praxisnahe Umsetzung im industriellen Kontext. Wie in Kapitel 5.1.4 erarbeitet, ist ein zentrales, einheitliches Modell in der Praxis aufgrund von Unternehmensgrenzen, lizenzrechtlichen Einschränkungen oder der starken fachdisziplinären Trennung häufig nicht realisierbar. Aus diesen Gründen verfolgt der hier vorgestellte Ansatz bewusst eine dezentrale Modellierung: Jedes Teilmodell bleibt eigenständig, wird aber über definierte Schnittstellen mit den anderen Modellen über einen Adapter gekoppelt.

Diese von den Experten bereitgestellten Modelle werden als Partialmodell bezeichnet. Ein Partialmodell beschreibt dabei jeweils ein konkretes System – sei es ein Subsystem innerhalb eines übergeordneten Zusammenhangs oder eine eigenständige Systemkomponente, die ihrerseits als unabhängiges System betrachtet werden kann. Die Kopplung dieser verteilten Partialmodelle bildet schließlich die Grundlage für die Variantenbildung im Gesamtsystems. Erst durch die gezielte Auswahl und Kombination bestimmter Komponenten und Subsysteme entsteht eine Variante des Gesamtsystems. Im Rahmen des entwickelten Konzepts werden auch die wechselseitigen Abhängigkeiten zwischen den Subsystemen und Komponenten berücksichtigt. Zwei grundlegende Beziehungsarten stehen dabei im Fokus. Zum einen sogenannte Wenn-dann-Beziehungen, bei denen die Auswahl eines bestimmten Subsystems zur Folge hat, dass ein weiteres Subsystem zwingend erforderlich wird. Zum anderen

werden Entweder-oder-Beziehungen berücksichtigt, um die Variation darzustellen, die entsteht, indem unterschiedliche Experten alternative Ausprägungen desselben Subsystems entwickeln und zur Verfügung stellen. Abbildung 6.7 veranschaulicht das Lösungskonzept dieser Arbeit.

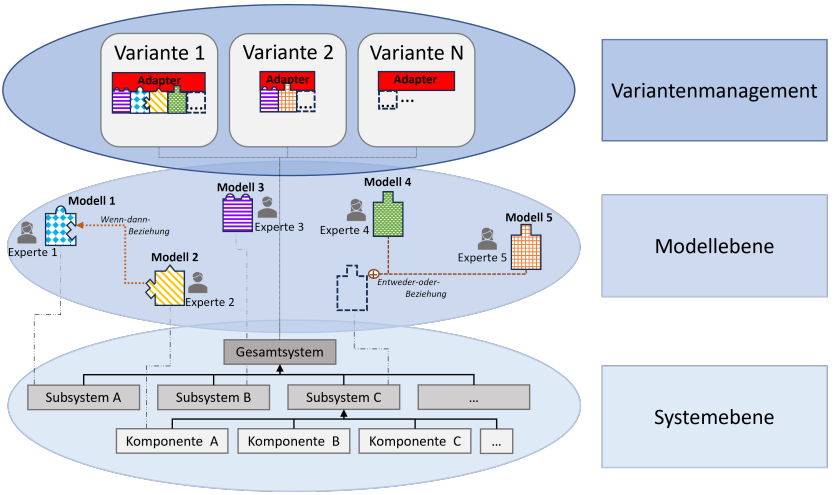



Abbildung 6.7: Lösungskonzept zur Auslegung einer Gesamtsystemvariante mit Partialmodellen.

Die im Lösungskonzept verwendeten methodischen Bausteine sowie die zugrunde liegenden Prinzipien, basierend auf den in der Descriptive Study I erarbeiteten Anforderungen, werden im Folgenden vorgestellt.

#### Lösungsbaustein: Dekomposition und Separation durch Partialmodelle

Damit Varianten frühzeitig entwickelt, analysiert und in den Gesamtkontext integriert werden können, müssen die eingesetzten Modelle nicht nur interoperabel, sondern auch variantenfähig sein. Methoden wie VAMOS oder das 150%-Modell bieten zwar eine theoretische Grundlage für das Variantenmanagement, stoßen jedoch in der industriellen Praxis auf mehrere Einschränkungen. Basierend auf diesen Erkenntnissen wird ein alternativer Lösungsansatz verfolgt, der die Vorteile verteilter, interoperabler Modellierung mit einem praxisgerechten Variantenmanagement kombiniert.

Wie bereits in Abschnitt 2.2 vorgestellt, besteht ein System aus beliebig vielen Subsystemen [HWFV12]. Eine schematische Dekomposition eines Systems ist in Abbildung 6.8 dargestellt. Dabei wird das Gesamtsystem (Ebene 1) in weitere Subsysteme (Ebene 2) unterteilt. Die Subsysteme können wiederum in weitere Subsysteme zerlegt werden, bis die Zerlegung mit den Komponenten auf der untersten Ebene (Ebene n) endet. Durch die Verknüpfung der Subsysteme bzw. Komponenten wird das Gesamtsystem gebildet und die geforderte Gesamtfunktion erfüllt.

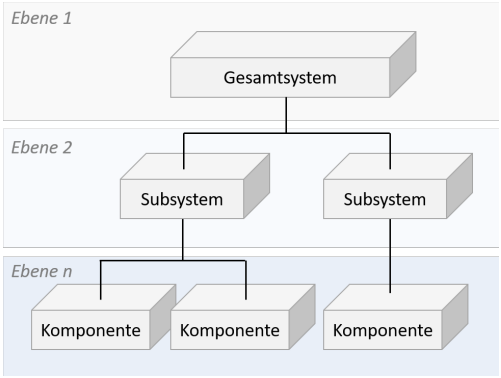



Abbildung 6.8: Schematische Darstellung der Dekomposition eines Systems.

Bei Varianten eines Gesamtsystems variieren die Subsysteme und deren Komponenten. Je nach Anforderung an die Funktionalität des Gesamtsystems werden verschiedene Subsysteme benötigt. Abbildung 6.9 zeigt schematisch die Dekomposition unter Berücksichtigung von Varianz. Das Gesamtsystem kann weiterhin in Subsysteme zerlegt werden. Allerdings befinden sich auf den Ebenen diesmal mehrere unterschiedliche Subsysteme und Komponenten. Für die Erzeugung von Varianten eines Gesamtsystems werden einzelne Subsysteme und deren Komponenten ausgetauscht oder weggelassen. Die farbigen Kugeln symbolisieren dabei einen Kopplungspunkt.

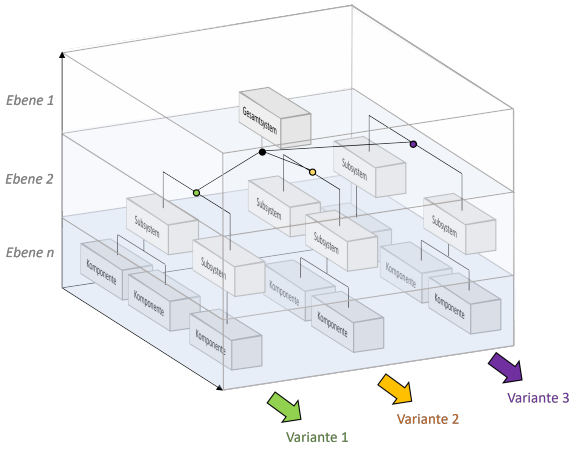



Abbildung 6.9: Schematische Darstellung der Dekomposition eines Systems unter Berücksichtigung von Varianz.

Das Ergebnis der Literaturanalyse hat gezeigt, dass die Variantenmodellierung und die Dekomposition eines Gesamtsystems mit Ansätzen wie VAMOS und dem 150% Modell häufig in einem Einzelmodell stattfindet. In der Praxis zeigt sich hierbei, dass

die induzierte Modellkomplexität durch die Abbildung von Produktvarianten in den immer größer werdenden Einzelmodellen zunimmt [MHS+21, DDBL11]. Bereitgestellte Lösungen der Softwarehersteller von Modellierungsumgebungen zur Auskopplung von Teilmodellen und zur Zusammenarbeit mit mehreren Experten erwiesen sich in der Praxis als nur bedingt hilfreich. Neben der Übertragung relevanter Modellelemente als Ausgangspunkt für die Subsystemmodellierung müssen weitere zahlreiche Elemente, die nicht unmittelbar mit dem zu modellierendem Subsystem zusammenhängen, ebenfalls ins neue Teilmodell kopiert werden. Ohne die erweiterten Modellelemente ist eine Nutzung der tatsächlich benötigten Modellelemente nicht möglich. Die interne Modellkomplexität der neuen Teilmodelle werden durch diese unterstützenden Elemente stark erhöht. Generell kann die strukturelle Komplexität eines Systems mit der strukturellen Komplexitätsmetrik (Gleichung 6.1) von Sinha und Suh berechnet werden [SS18].

$$C = \underbrace{\sum_{i=1}^{n} \alpha_i}_{\text{1-Komponenten}} + \underbrace{\left( \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_{ij} A_{ij} \right)}_{\text{2-Schnittstellen}} \underbrace{\frac{E(A)}{n}}_{\text{3-Architektur}} \tag{6.1}$$

Beitragende Elemente für die strukturelle Komplexität $C$ sind die Größe oder Anzahl der Teile eines Systems (1-Komponenten), die Wechselwirkungen zwischen diesen (2-Schnittstellen) und die topologische Anordnung der Systemschnittstellen (3-Architektur) [SS18]. Ausgehend von der Gleichung kann angenommen werden, dass eine Aufteilung eines Gesamtsystems in mehrere Partialmodelle die interne Komplexität in den Modellen durch die geringere Anzahl an darzustellender Elemente und Knoten im Vergleich zu einem Einzelmodell reduziert. Folglich sind die Modelle für den Menschen einfacher zu interpretieren und zu bearbeiten. Wie bereits die Analyse der Problemstellung im Rahmen der DS 1 gezeigt hat, stellt insbesondere die Modellierung von Varianten eine zentrale Herausforderung dar, da viele der in der Praxis angewandten Methoden häufig ein hohes Maß an induzierter Modellkomplexität mit sich bringen. Diese Komplexität wirkt sich wiederum unmittelbar auf die praktische Anwendbarkeit der Modelle im industriellen Kontext aus, besonders in Zusammenarbeit mit vielen (externen) Experten. Das daraus abgeleitete Subziel der Arbeit adressiert daher die Notwendigkeit, eine Methodik zu entwickeln, die einerseits die Variantenvielfalt systematisch abbildet und andererseits die Nutzbarkeit der Modelle durch gezielte Komplexitätsreduktion sicherstellt.

In dieser Arbeit wird daher der Ansatz verfolgt, die Dekomposition ebenfalls auf die Modellebene zu übertragen. Statt eines einzigen Gesamtmodells wird das System in modularen Teilsichten abgebildet. Dabei werden Subsysteme in separaten Partialmodellen modelliert und ausgelegt. Abbildung 6.10 zeigt konzeptionell die Abkapselung eines Subsystems mit dessen Komponenten als Partialmodell in blau dargestellt. Über eine definierte gemeinsame Schnittstelle als Kopplung (roter Punkt) werden die verschiedenen Partialmodelle zu einer Gesamtsystemlösung zusammengefügt. Eine Separation der Subsysteme und Subkomponenten ermöglicht eine unabhängige Entwicklung, erhöht die Wiederverwendbarkeit und verbessert die Wartbarkeit von Modellen.

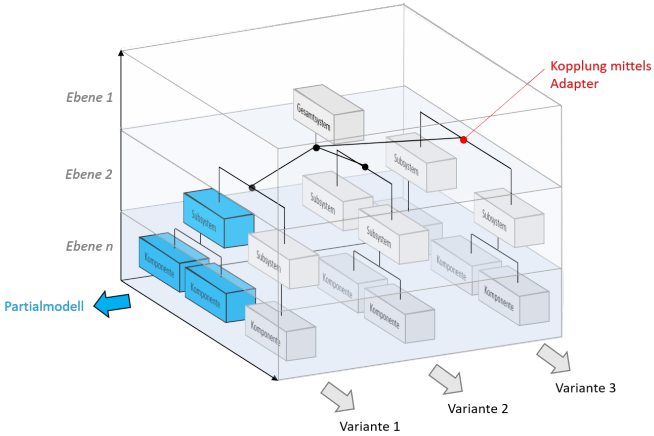



Abbildung 6.10: Beispielhafte Darstellung der Abkapslung eines Subsystems als Partialmodell.

Die jeweiligen Partialmodelle werden als Black-Box betrachtet, die über definierte Ein- und Ausgänge verfügen. Abbildung 6.11 zeigt den beispielhaften Aufbau eines Partialmodells als Black-Box. Der Umfang des Partialmodells wird dabei durch die Systemgrenze des abzubildenden Subsystems definiert.

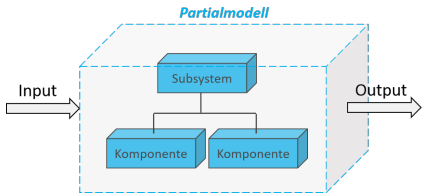



Abbildung 6.11: Partialmodell als Black-Box.

Abbildung 6.12 zeigt schematisch das Prinzip zur Kopplung von Partialmodellen verschiedener Unternehmen für die Auslegung eines Gesamtsystems (angelehnt an Abbildung 6.6). Durch die Aufteilung der Variantenmodellierung auf mehrere Partialmodelle wird eine Modularität erlaubt, die eine einfache Erweiterung und Modifikation der Modelle um neue Varianten durch die Integration neuer Technologiemodelle in das bestehende Gesamtsystemmodell erlaubt. Zugleich können Partialmodelle für Subsystemgruppen wiederverwendet werden. Die flexible Anbindung ermöglicht den Austausch sowie die Wiederverwendung von Partialmodellen verschiedener Experten für dasselbe Subsystem, etwa im Rahmen von Variantenbildungen in der Zusammenarbeit mit Zulieferern.

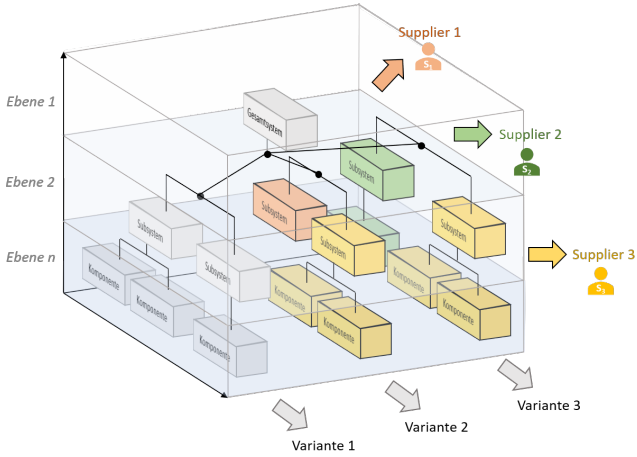



Abbildung 6.12: Kombination von Partialmodellen unterschiedlicher Experten/Unternehmen zu Systemvarianten.

**Lösungsbaustein: Interoperabilität und Multidisziplinarität durch XML-Adapter**

Bei der Zusammenarbeit mit externen Partnern und deren Partialmodellen müssen rechtliche und IT-bezogene Einschränkungen beachtet werden, was die digitale Kollaboration und den Datenaustausch erschwert. Gleichzeitig sind für die disziplinäre Vorhersage und Analyse unterschiedliche Methoden und Modelle notwendig. Die genutzten Tools bieten keine standardisierte Kopplung, sodass eine hohe Schnittstellenvielfalt vorherrscht. Das Resultat ist eine Steigerung der Komplexität des multidisziplinären Gesamtmodells des Systems. Mit dem Ansatz zur Nutzung föderierter Partialmodelle für die Variantenmodellierung entstehen weitere Schnittstellen, die die Kopplungskomplexität erhöhen.

Für die Umsetzung interoperabler Modelle ist daher eine Methodik erforderlich, die eine konsistente und einheitliche Schnittstelle gewährleistet, sodass der Implementierungsaufwand auf ein praktikables Maß beschränkt bleibt. Im Rahmen dieser Arbeit wird zur Umsetzung interoperabler Modelle ein Adapterkonzept verfolgt, das eine strukturierte Kopplung zwischen Teilmodellen ermöglicht. Vor dem Hintergrund der ergänzenden Forschungsfrage, die die Anwendbarkeit im industriellen Kontext adressiert, zeigen sich bestehende Tool-seitige Adapterlösungen, wie in Kapitel 5 dargelegt, als nur bedingt geeignet. Insbesondere ihre Abhängigkeit von proprietären Softwarelösungen sowie der damit verbundene Lizenzaufwand schränken ihre praktische Nutzbarkeit ein.

Zur Reduktion dieser Barrieren wird stattdessen in dieser Arbeit eine lizenzneutrale, universelle Schnittstelle auf Basis des XML-Formats implementiert. Diese Lösung gewährleistet eine flexible, standardisierte Kopplung der Partialmodelle und unterstützt damit ein niedrigschwelliges, systemübergreifendes Modellmanagement. Im Gegensatz zu anderen Schnittstellentechnologien ist der Aufwand zur Implementierung in

den Tools geringer. Die Wahl des Tools in ihren Partialmodellen ist den Partnern freigestellt, solange dieses XML unterstützt. Der Adapter fungiert als einzige Datenquelle und wird im Laufe der Auslegung durch die Daten der Partialmodelle ergänzt. Damit schafft dieser einen allgemeingültigen Datenbestand, indem alle Daten an einem fest definierten Ort hinterlegt werden. Mit dem Konzept der single source of truth basieren die Ergebnisse der Partialmodelle auf dem gleichen Datenset. Dadurch wird eine Redundanz von mehreren Datenquellen vermieden. Gleichzeitig wird die Weiterverarbeitung der Daten in den Partialmodellen unterstützt, indem durch die Verlinkung auf die Einträge in dem Adapter die Parameter automatisiert angepasst werden. Alle Partialmodelle dürfen die Datei lesen und befüllen. Ändert ein Partialmodell Parameter, müssen die weiteren Partialmodelle die Informationen aus dem Adapter erneut einlesen, um Auswirkungen auf deren Auslegung zu untersuchen. Ein weiterer Aspekt ist die Wahrung von Intellectual Property (IP). Bei der Kopplung wird über den Adapter nur ein limitierter Datensatz ausgetauscht, der für die jeweilige Systemteilauslegung benötigt ist. Dadurch verbleibt das erzeugte Wissen in den Modellen und das Know-how beim jeweiligen Experten. Zusätzlich kann der Datenaustausch automatisiert werden, indem der XML-Adapter mit den Austauschdaten an einem universell zugänglichen Ort (z.B. Server) für alle Experten bereitgestellt wird. Insgesamt steigen somit die Erfolgschancen für ein dezentrales, partnerübergreifendes, heterogenes Gesamtmodell.

Abbildung 6.13 stellt den Kopplungsansatz der Partialmodelle und die Detailtiefe vor. Die Detailtiefe eines Partialmodells ist offen und kann variabel gestaltet werden. Ein Partialmodell kann somit beispielsweise aus einem einzelnen SysML-Modell sowie aus mehreren heterogenen Modellen bestehen, die gemeinsam das betrachtete Subsystem beschreiben. Der entwickelte Ansatz setzt darauf, die Stärken der einzelnen Entwicklungsumgebungen zu nutzen und analytische oder geometrische Berechnungen außerhalb der SysML-Modelle zu belassen. Stattdessen werden die heterogenen Modelle über definierte Schnittstellen für einen konsistenten Parameter- und Datenaustausch innerhalb des Partialmodells miteinander verknüpft. Im kollaborativen Kontext kann die Nutzung bestehender Modellierungsumgebungen ohne zusätzliche Lizenzkosten erfolgen, was die Integration in bestehende Arbeitsprozesse erleichtern kann. Zusätzlich wird die Akzeptanz für eine Zusammenarbeit im Gesamtverbund gefördert, da präferierte und spezialisierte Modellierungsumgebungen der einzelnen Experten weiterverwendet werden können.

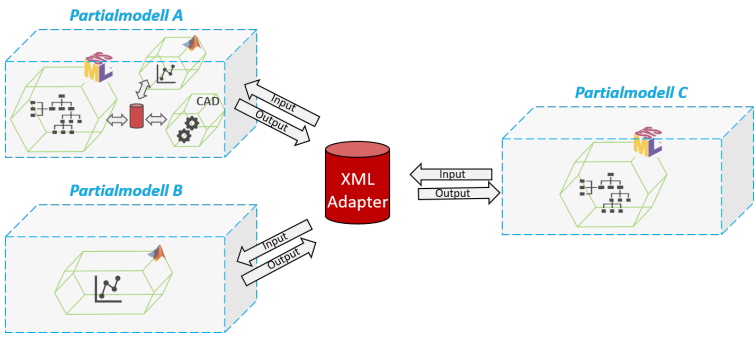



Abbildung 6.13: Detailtiefe eines Partialmodells und Kopplungsansatz.

**Lösungsbaustein: Interne Modellstruktur durch EVA-Prinzip**

Die Herstellung von Interoperabilität durch den Einsatz eines XML-Adapters stellt einen zentralen Lösungsbaustein dar. Gleichzeitig zeigen sich jedoch zwei weitere Anforderungen, die bereits ausführlich in Kapitel 5 erläutert wurden und im Lösungsansatz berücksichtigt werden müssen. Bisherige Ansätze zur Kopplung mehrerer SysML-Modelle erhöhen die interne Modellkomplexität, indem eine Vielzahl an unterstützenden Elementen im SysML-Modell, die nicht direkt dem realen System zugeordnet werden können, integriert werden. In dieser Arbeit wird eine textbasierte Schnittstelle zwischen den Tools verwendet, wodurch die Modellgröße nur bedingt beeinflusst wird. Über den Eingang wird dem Partialmodell ein Minimalset an Informationen übergeben, sodass dieses nur die Informationen für die eigene Subsystemauslegung erhält, ohne weitere Kenntnis über die Modellierung der anderen Subsysteme in deren Partialmodellen zu erhalten. Die Implementierung der Schnittstelle und das Ein-/Auslesen der Daten aus dem Adapter kann mit wenigen Modellelementen realisiert werden.

Zwar ist der Implementierungsaufwand für die XML-Datenschnittstelle minimal, dennoch führt die Integration solcher Schnittstellen zu zusätzlichen Strukturelementen innerhalb der Modelle. Diese wirken sich auf deren Komplexität, Verständlichkeit und Pflegeaufwand aus. Um dennoch eine klare Struktur und leichte Wartbarkeit zu gewährleisten, wird das EVA-Prinzip als weiterer Baustein für das Lösungskonzept herangezogen. Durch die explizite Trennung von Eingabe-, Verarbeitungs- und Ausgabegrößen entsteht ein transparenter Modellaufbau, der die manuelle Nachvollziehbarkeit und Anpassbarkeit deutlich verbessert. Dabei werden die innerhalb der Modelle benötigten zusätzlichen Elemente für die XML-Schnittstelle von den Elementen für die Systemmodellierung separiert (Abbildung 6.14). Damit ermöglicht die EVA-Struktur eine klare interne Gliederung der Modelle und die teilweise Speicherung von unterstützenden Elementen in Support-Ordnern, wodurch diese erkennbar sind.

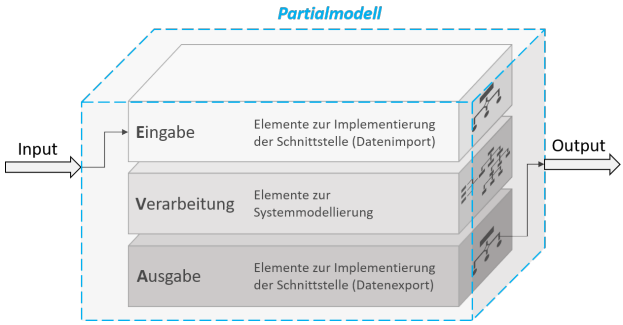



Abbildung 6.14: Interne Modellstruktur nach dem EVA-Prinzip.

Das zugrunde liegende Prinzip der Trennung nach dem EVA-Prinzip ist nicht exklusiv auf SysML-Modelle beschränkt. Vielmehr bietet es ein übertragbares Strukturkonzept, das in allen Modellen innerhalb eines Partialmodells anwendbar ist, beispielsweise in CAD- oder Simulationstools. Besonders in einer stark heterogenen Modelllandschaft wirkt sich eine wiederkehrende, einheitliche Struktur in den Modellen positiv auf den

Umgang mit der internen Komplexität auf. Zudem wird dadurch die Pflege der Modelle erleichtert. Diese kann durch jeden Experten für sein spezifisches Tool selbst durchgeführt werden. Hierbei müssen lediglich die Ein- und Ausgabeparameter mit den anderen Partialmodellen abgeglichen und die Modellelemente in den Supportordnern angepasst werden, ohne Änderungen innerhalb der Systemmodellierung an sich durchzuführen.

**Lösungsbaustein: Kontextsensitive Modellierung für das Customizing**

Wie bereits in Kapitel 4 ausführlich beschrieben, werden Varianten häufig innerhalb eines einzigen Gesamtmodells abgebildet. Durch Innovationen und wechselnde Kundenwünsche ist eine Flexibilität in der Modellierung und Untersuchung von Varianten notwendig. Bei den bisherigen Ansätzen ist eine nachträgliche Ergänzung neuer Varianten durch die starke Verschachtelung im Modell mit einem erheblichen Aufwand verbunden. Bei Änderungen an einzelnen Subsystemen ist es daher erforderlich, stets das gesamte Modell zu laden und in seiner Gesamtheit zu analysieren.

Im Gegensatz dazu werden bei einer kontextgesteuerten Sicht nur die relevanten Subsysteme betrachtet. Betrachtet man die Produktstruktur eines Flugzeugs, kann eine Änderung innerhalb der Produktstruktur unterschiedliche Auswirkungen auf die zugehörigen Subelemente haben (Abbildung 6.15). Abhängig von der Art der Änderung und den bestehenden Abhängigkeiten sind nicht zwangsläufig alle untergeordneten Komponenten betroffen. Narváez und Monroy heben hervor, dass durch kontextsensitive Modellierung im Customizing eine gezielte Reduktion der Modellkomplexität möglich ist, da bei Änderungen nur die jeweils betroffenen Subsysteme zur Bearbeitung freigegeben werden. [NM09].

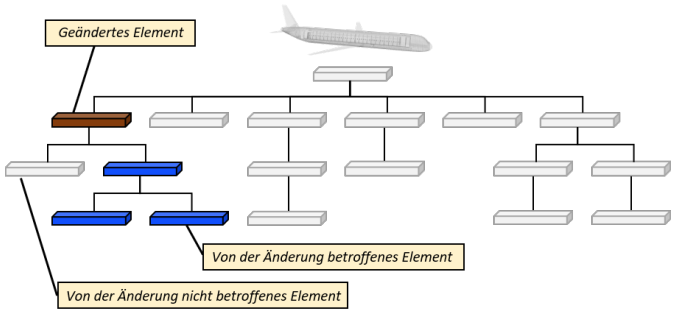



Abbildung 6.15: Visualisierung der Auswirkungen von Änderungen innerhalb der Produktstruktur eines Flugzeugs im Rahmen des Änderungsmanagements, modifiziert von [NM09].

Mit Hilfe des erarbeiteten Lösungskonzepts für föderierte Partialmodelle wird das Customizing um eine kontextgesteuerte Sicht erweitert. Dabei wird das Modell nicht mehr als statische Gesamtdarstellung verstanden, sondern als dynamisch nutzbare Repräsentation, die sich abhängig von Anwendungskontext (z.B. Variantenmodellierung einer neuen Flugzeugkonfiguration oder Customizing einer Baseline) gezielt anpassen lässt, indem nur bestimmte Modelle aktiviert und miteinander kombiniert werden.

Dies ist besonders in der Zusammenarbeit mit Zuliefern, bei der verschiedene Akteure dezentral an Teilmodellen arbeiten, erforderlich. Das Customizing setzt nach der Modellierung der Gesamtsystemvarianten an. Zu diesem Zeitpunkt wurden die übergeordneten Randbedingungen auf Systemebene bereits definiert und hinsichtlich ihrer Auswirkung analysiert. Anpassungswünsche, die auf nachträgliche, individuelle Kundenanforderungen zurückzuführen sind, beziehen sich daher ausschließlich auf ein bereits bestehendes Subsystem. Betroffen sind in diesem Zusammenhang lediglich die zugehörigen Partialmodelle sowie gegebenenfalls abhängige Subsysteme oder Komponenten. Die Anpassung erfolgt dann individuell in den betroffenen Partialmodellen.

# 7. Methodenentwicklung zur Kopplung von Partialmodellen für Varianten und das Customizing

In diesem Kapitel wird die technische Umsetzung des entwickelten Lösungsansatzes als Unterstützung für den Variantenentwurf im Detail vorgestellt, indem es auf einen realen Anwendungsfall im Luftfahrtkontext angewandt wird. Die gezeigten Beispiele beziehen sich für ein besseres Verständnis und für eine einfachere Darstellung auf den industriellen Anwendungsfall aus Kapitel 8. Bei diesem werden zwei verschiedene Varianten für die elektrische Versorgung der Flugzeugkabine ausgelegt und die Einflüsse auf die Flugzeugkabine untersucht. Die Implementierung erfolgt unter Verwendung des Cameo Systems Modeler. Im Fokus stehen dabei der Aufbau eines Partialmodells sowie dessen Integration in ein föderiertes Gesamtmodell. Zuerst wird das Lösungskonzept zur Variantenbildung auf den konkreten Anwendungsfall übertragen. Darauf aufbauend erfolgt eine detaillierte Darstellung des Aufbaus eines Partialmodells sowie der Kopplung heterogener Modelle innerhalb des Partialmodells. Anschließend wird der Aufbau des Adapters zur Kopplung mehrerer Partialmodelle beschrieben, einschließlich der internen Strukturierung nach dem EVA-Prinzip innerhalb der SysML-Modelle. Abschließend wird der automatisierte Prozess zur Generierung des Gesamtsystems erläutert. Damit leitet das Kapitel die Descriptive Study II ein.

## 7.1 Methodik zur Variantenbildung und für das Customizing

In diesem Abschnitt wird die Umsetzung des Lösungskonzepts für die Kopplung mehrerer Partialmodelle zu einem föderierten Modell für die Bildung von Varianten sowie für die nachträgliche Änderung einer Variante im Rahmen des Customizings aus Kapitelabschnitt 6.2 vorgestellt. Mit dem methodischen Lösungsansatz können sowohl Varianten auf Gesamtsystemebene kollaborativ abgebildet als auch das Customizing berücksichtigt werden. Während Varianten vor allem auf übergeordneter Ebene (Flugzeug) stattfinden und standardisierte Variationsmodule kombiniert [Sch02, S. 84], bezieht sich das Customizing auf Modifikationen auf Subsystemebene (z.B. Kabine, siehe Abbildung 7.1).

Abbildung 7.1 zeigt die Gesamtsystemdarstellung eines Flugzeugs als Klassendiagramm. Angelehnt an Abbildung 6.6 sind verschiedene Subsysteme und Komponen-

ten ausgewählt, um zwei beispielhafte Varianten zu realisieren. In beiden Varianten werden die Kabine und der Rumpf benötigt. Einzig bei dem Stromversorgungssystem unterscheiden sich die beiden Varianten. Das Subsystem *Stromversorgungssystem* stellt damit einen Variationspunkt dar. Je nach auszulegender Energievariante wird eine bestimmte Kombination der einzelnen Subsubsysteme/ Komponenten benötigt. Komponenten oder Subsysteme, die für die Variante nicht relevant sind, werden nicht betrachtet. Am Beispiel der Varianten A und B sind die entsprechenden Systemgruppen farblich eingefärbt, um das Konzept zu verdeutlichen.

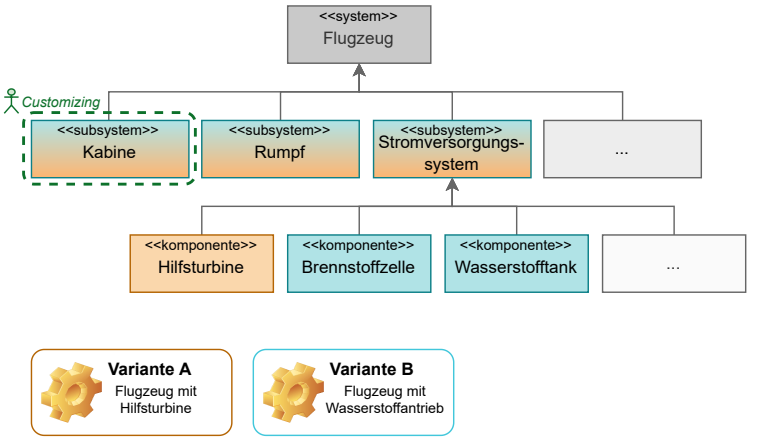



Abbildung 7.1: Beispielhafte Auswahl der Subsysteme für Flugzeugvarianten mit alternativen Stromversorgungssystemen. Orange hinterlegte Kacheln zeigen die erforderlichen, miteinander zu koppelnden Subsysteme für die Auslegung der Variante A und blaue Kacheln für die Variante B.

Jedem dieser Subsysteme oder Komponenten ist ein Partialmodell zugeordnet, das dessen Eigenschaften und Verhalten abbildet. Die Modellierung erfolgt dabei entweder auf Basis deskriptiver, analytischer Modelle oder einer Kombination beider Ansätze. Für die Auslegung von Varianten bzw. im Rahmen des Customizings variiert die Anzahl der angesteuerten Partialmodelle.

Für die Erstellung einer Gesamtsystemvariante (z.B. Flugzeug mit Brennstoffzelle) werden mehrere Partialmodelle miteinander über eine XML-Datei als Adapter gekoppelt und Parameter zwischen diesen ausgetauscht. Dabei wird bei der Erstauslegung der Flugzeugvariante im Entwurf eine Standard-Mission und -Kabinenausstattung gewählt sowie eine Einklassenbestuhlung untersucht, um die maximale Kapazität des Gesamtsystems für den neuen Flugzeugtyp zu analysieren.

Beim Customizing hingegen wird die Standardausstattung ausgetauscht, um alternative, individualisierte Kabinenkonfigurationen zu untersuchen. Als Ausgangslage der Neuauslegung dient der Parametersatz einer bereits ausgelegten Variante (z.B. Variante 1). Im Gegensatz zur Variantenbildung wird nur das Partialmodell der Kabine

angesteuert. In diesem werden dann Änderungen an der Subsystemausstattung vorgenommen und, sofern gewünscht, die Anzahl der Passagierklassen variiert. Abbildung 7.2 veranschaulicht die genannten Zusammenhänge.

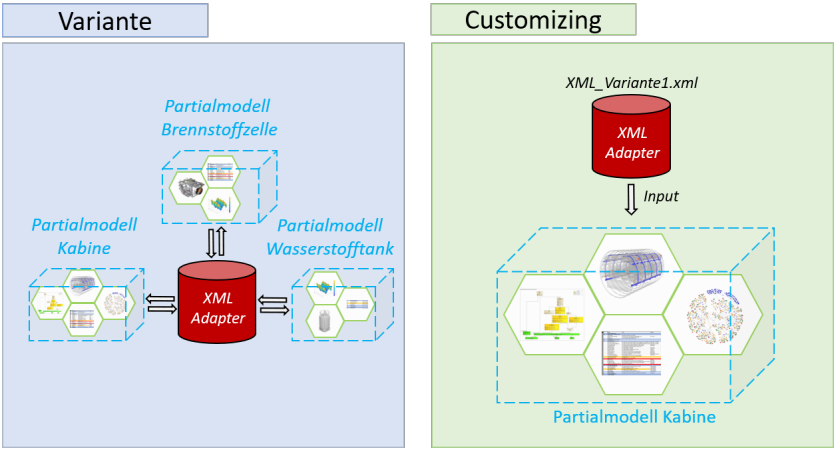



Abbildung 7.2: Allgemeine Darstellung der Fallunterscheidung Variantenbildung und Customizing. Bei der Variantenbildung werden mehrere Partialmodelle angesteuert, während beim Customizing nur ein Partialmodell verwendet wird. Ausgangslage für das Customizing bildet die XML-Datei einer spezifischen Variante.

Der folgende Abschnitt beschreibt zuerst den Prozess für die Kopplung der Partialmodelle für die Variantenbildung und für den Austausch von Daten. Anschließend wird der Prozess für das Customizing vorgestellt und ein Einblick in die Verarbeitung der Parameter in einem Partialmodell gegeben.

### 7.1.1 Methodisches Vorgehen und Prozessbeschreibung für die Variantenbildung

Die Ausführung der einzelnen Partialmodelle erfolgt sequenziell. Die Erzeugung einer Variante erfolgt durch das Zusammenschalten mehrerer Partialmodelle. Je nach Variante werden unterschiedliche Partialmodelle angesteuert, die dann als Summe das Gesamtsystem abbilden und eine Analyse ermöglichen.

Abbildung 7.3 zeigt den zugehörigen methodischen Ablauf für die Erzeugung einer Systemvariante. Gestartet wird der Prozess mit der Eingabe der Startparameter. Dazu gehören die Top-Level Flugzeuganforderungen und Variablen wie Reichweite und Passagieranzahl für die auszulegende Variante. Im nächsten Schritt wird festgehalten, welche Partialmodelle ausgewählt und angesteuert werden müssen, um die Variante zu erzeugen. Anschließend erfolgt die nacheinander stattfindende Ausführung der Partialmodelle für die jeweilige Auslegung eines Subsystems. Dabei wird zuerst das deskriptive SysML-Modell angesteuert. Dieses importiert relevante Anforderungen aus der Exceltabelle sowie Parameter aus der XML-Datei. Basierend auf den

importierten Parametern wird das System ausgelegt. Die Modellierung der Systemarchitektur sowie überschlägige Berechnungen finden in dem SysML-Modell statt. Sind für die Auslegung des Systems weiterführende Analysen (z.B. Geometrieauslegung) notwendig, werden die analytischen Modelle (z.B. Matlab/Simulink) aus dem SysML-Modell heraus angesteuert und ausgeführt. Im Anschluss werden die generierten Parameter aus den Analysemodellen an das SysML-Modell zurückgeführt und ausgewählte Parameter zusammen mit den Anforderungen in die XML-Datei exportiert. Darüber hinaus können die Ergebnisse der Anforderungsüberprüfung ebenfalls in die Exceltabelle überführt werden. Die Ergebnisse der Anforderungsüberprüfung können dann entweder der XML-Datei oder der Exceltabelle entnommen werden.

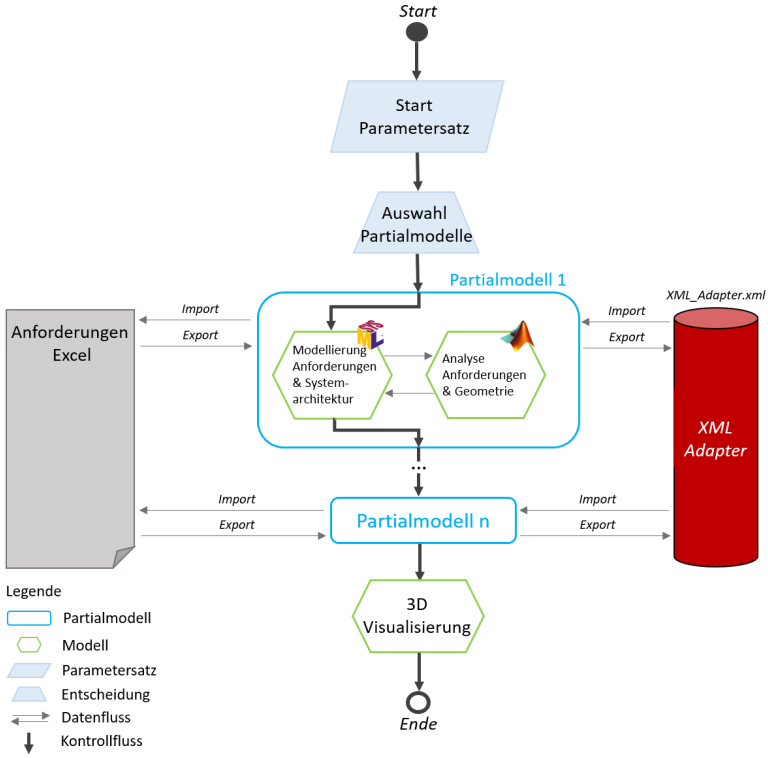



Abbildung 7.3: Methodisches Vorgehen für die Auslegung einer Flugzeugvariante.

Eine Abfrage überprüft nach Beendigung der Ausführung eines Partialmodells, ob die Variante bereits vollständig ausgelegt wurde oder ob weitere Partialmodelle angesteuert werden müssen. Diese Schleife wird wiederholt durchlaufen, bis alle notwendigen Subsysteme mit n Partialmodellen ausgelegt wurden. Abschließend wird die Variante mit einem 3D Modell visualisiert und der Prozess beendet.

Während der Variantenauslegung wird eine XML-Datei mit dem universellen Namen XML_Adapter.xml verwendet. Dadurch ist eine Anpassung des Dateinamens des Ad-

apters in den Partialmodellen nicht notwendig, da sowohl der Speicherort als auch der Name der Adapterdatei immer gleich bleiben. Nachdem eine Variante vollständig ausgelegt und der gesamte Prozess durchlaufen ist, kann die generierte und mit Parametern befüllte XML-Datei unter einem spezifischen Namen (z.B. XML_V1_Hydrogen.xml) an einem Ort der Wahl abgespeichert werden. Dadurch ist für den Menschen eine Zuordnung der XML-Datei zu der jeweiligen Variante möglich. Allgemein fungiert die XML-Datei als Beschreibung der Flugzeugvariante und dient als zentrale Datenablage, da sie sowohl alle Flugzeugkomponenten als auch Ergebnisse von Anforderungsüberprüfungen enthält.

Der Prozess zur Auslegung einer Variante erfolgt teilautomatisiert. Der Import und Export der Parameter aus den Anforderungen der Exceltabelle und der XML-Datei sowie Ausführung des Partialmodells, bestehend aus mehreren heterogenen Modellen, erfolgen automatisiert. Lediglich die Auswahl der Partialmodelle und Ansteuerung dieser als auch die Operation zur Überprüfung der Vollständigkeit der Variante erfolgen manuell. Ein Ansatz für eine vollständige Automatisierung des Prozesses wird in 7.5 vorgestellt.

Für die Untersuchung einer Flugzeugvariante wird standardmäßig eine maximale Passagierauslastung in einer Einklassenbestuhlung untersucht. Zudem wird eine Standardausstattung der Kabinenkomponenten gewählt. Dadurch ist eine erste Machbarkeitsanalyse der Variante sowie eine Überprüfung von Anforderungen an die Variante auf Gesamtsystemebene möglich. Dennoch ist es möglich, für die Auslegung eine andere Ausstattung oder Passagierbelegung als Standard festzulegen. Allerdings wird bei der Variantenauslegung auf Flugzeuggesamtsystemebene nur eine Konfiguration untersucht. Weitere Designänderungen oder Untersuchungen von verschiedenen Kabinenkonfigurationen erfolgen erst im Customizing.

### 7.1.2 Methodisches Vorgehen und Prozessbeschreibung für das Customizing

Das Customizing eines Flugzeugs findet ausschließlich im Subsystem der Kabine statt. Dabei wird eine bereits auslegte Flugzeugvariante als Basiskonfiguration verwendet. Subsysteme, wie das Antriebssystem oder das Stromversorgungssystem, bleiben gleich. Änderungen werden nur an den Kabinenkomponenten vorgenommen. Dadurch wird für das Customizing nur ein Partialmodell angesteuert. Die anderen Partialmodelle werden für die Untersuchung nicht benötigt.

Das kann einerseits eine veränderte Sitzkonfiguration sein. Im Rahmen der Variantenauslegung wird eine Einklassenbestuhlung, nur Economyklasse (Economy Class, EC) untersucht, um die maximale Passagierauslastung zu betrachten. Im Rahmen des Customizings können Airlines auch eine Zwei- oder Dreiklassenbestuhlung mit Businessklasse (Business Class, BC) und Erste Klasse (First Class, FC) wählen. Die Änderung der Sitzklassen hat nicht nur Auswirkungen auf das Gewicht und die Anzahl der Passagiere, sondern auch auf die benötigten Stauräume der Küchen oder die Anzahl der Waschräume. Besonders in der BC und FC teilen sich weniger Passagiere einen Waschraum und eine größere Auswahl an Essen und Getränken wird vorgehalten. Andererseits können ganze Baugruppen oder Subsubsysteme geändert werden.

Die Zulieferer bieten eine Vielfalt an Varianten für Gepäckablagen, Sitze und Monumente an. Zusätzlich können Airlines entscheiden, welchen Service sie Passagieren anbieten möchten. Je nachdem, werden im Flugzeug Komponenten wie Bildschirme für In-Flight Entertainment oder Monumente wie eine Bar eingebaut. Änderungen an den Baugruppen haben zusätzlich zu den Auswirkungen bei Gewicht und geforderter Leistung Einfluss auf den Passagierkomfort.

Das methodische Vorgehen für das Customizing einer bereits definierten Flugzeugvariante ist in Abbildung 7.4 dargestellt. Für das Customizing wird nur das Partialmodell Kabine angesteuert. Da es in diesem Beispiel ausschließlich aus einem Matlab-Modell besteht, erfolgt auch lediglich dessen Ausführung.

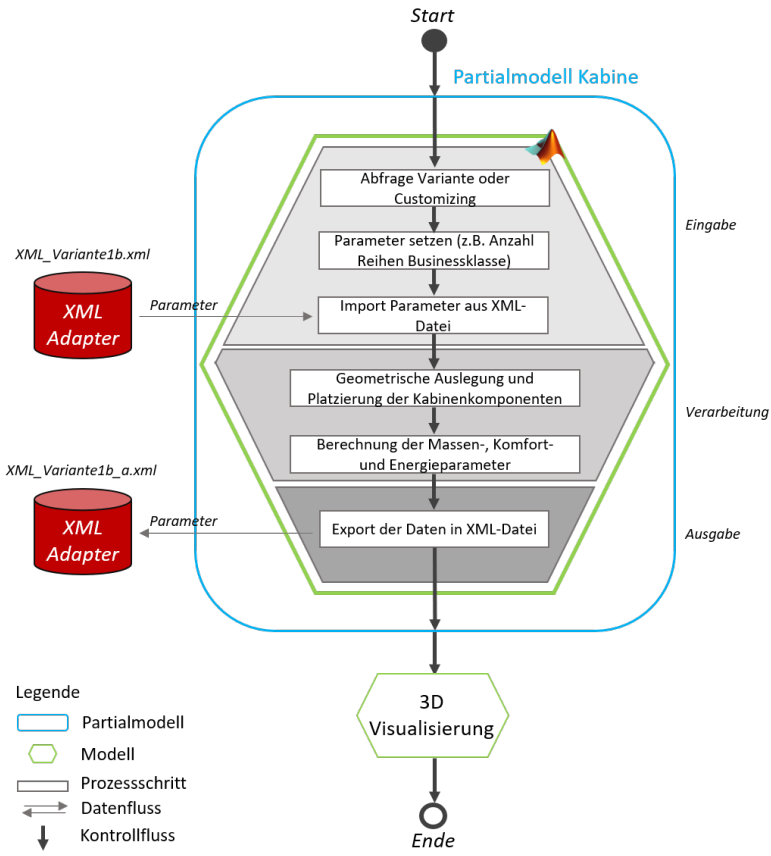



Abbildung 7.4: Methodisches Vorgehen für das Customizing in der Flugzeugkabine.

Das Matlab-Modell ist ebenfalls nach dem EVA-Prinzip aufgebaut. In der Eingabe überprüft das Modell, ob eine Variante ausgelegt oder das Customizing gestartet werden soll. Da in diesem Fall das Customizing gestartet wird, erfolgt im nächsten

Schritt das Setzen von individuell anpassbaren Parametern (z.B. Anzahl Reihen mit Businessklassesitzen). Im Anschluss werden weitere Parameter aus der XML-Datei der Basiskonfiguration (XML_Variante1b.xml) importiert, die ansonsten vom SysML-Modell bereitgestellt worden wären. Dazu gehören beispielsweise die Kabinenlänge und der Rumpfdurchmesser. Im nächsten Abschnitt, der Verarbeitung, wird der gleiche Auslegungsprozess durchlaufen wie bei der Variantenbildung. Lediglich die Ausgangsparameter unterscheiden sich durch die manuelle Eingabe der neuen individuell angepassten Parameter. Dabei werden zuerst die Kabinenkomponenten geometrisch ausgelegt und platziert. Anschließend werden die Bewertungskriterien wie Gewicht, Passagierkomfort und Energiebedarf berechnet, um eine Bewertung der ausgelegten Kabinenkonfiguration im Vergleich zu der Basiskonfiguration zu ermöglichen. Im letzten Abschnitt der Ausgabe werden die neu ermittelten Parameter in die XML-Datei exportiert und der Name der Datei umbenannt (XML_Variante1b_a.xml). Dabei wird nur der Kabinen-Baumknoten in der XML der Basisdatei mit den neuen Werten überschrieben. Die Baumknoten der anderen Subsysteme werden nicht verändert. Der zugehörige Matlab-Code ist in Anhang A.3.3 dargestellt. Im Anschluss an die Ausführung des Kabinenpartialmodells wird die neu konfigurierte Kabine in einem 3D Modell visualisiert. Die Darstellung erfolgt in einer höheren Detailtiefe als beim Variantenentwurf, da das Customizing weitere Aspekte wie Design und Optik in der Bewertung berücksichtigt. Damit ist der Prozess beendet.

## 7.2 Aufbau und Modellbeschreibung eines Partialmodells

Die Detailtiefe eines Partialmodells ist offen. Ein Partialmodell kann aus mehreren, heterogenen Modellen bestehen. Je nach System werden mehrere Modelle unterschiedlicher Fachdisziplinen benötigt, um das System zu beschreiben. Dazu gehören deskriptive Modelle wie SysML-Modelle für die Beschreibung der Systemarchitektur oder für Anforderungsanalysen und analytische Modelle wie Matlab oder Catia für die Geometrieauslegung, funktionale Analyse oder Visualisierung des Systems. Die folgenden Abschnitte geben Einblicke in den Modellaufbau der einzelnen Disziplinen am Beispiel der Kabinenmodellierung und stellen die relevanten Elemente und Diagramme der Modellierungsumgebungen vor.

### 7.2.1 Architekturmodellierung in der SysML

In dieser Arbeit wird die SysML v1.7 verwendet. Wie bereits in 3.2.3 beschrieben, wird ein System in der SysML mit Diagrammen definiert und visualisiert. Dabei wird mit dem Diagramm nur ein bestimmter Aspekt des Systems veranschaulicht [Abu12]. Dafür werden in den verschiedenen Diagrammen spezifische Knoten- und Verbindungselemente verwendet. Eine Einführung in diese und in die für die Arbeit angewandten Diagramme, generiert mit dem Cameo System Modeler, ist in Anhang A.2 gegeben. Der folgende Abschnitt stellt die Systemarchitekturmodellierung in der hier erarbeiteten Methode am Beispiel der Kabine mit der SysML vor.

#### 7.2.1.1 Modellierung des Systems Kabine

Das SysML-Modell wird zur Erstellung der Systemarchitektur und für die Verfolgung der Anforderungen verwendet. Ausgehend von den Anforderungen an die Kabinenkonfiguration und den Randbedingungen (z.B. verfügbarer Bauraum, Energiesystem), wird die zugehörige Kabinensystemarchitektur erstellt. Dafür werden zwei Blockdefinitonsdiagramme genutzt.

Mit dem ersten werden die Systemkomponenten beschrieben. Abbildung 7.5 zeigt das Blockdefinitionsdiagramm für die Generalizierung (Pfeil mit offener Spitze). Die Kabine besteht aus mehreren Systemkomponenten. Jede Systemkomponente wird als Block (hellbraun) repräsentiert und durch Eigenschaften beschrieben. Alle Systemkomponenten besitzen dabei gleiche Eigenschaftsvariablen, die sich nur in ihren Werten unterscheiden. Diese Eigenschaftsvariablen können mit Hilfe der Vererbung weitergegeben werden, ohne das eine erneute Initialisierung notwendig ist. Dafür wird eine generelle Beschreibung einer Systemkomponente, genannt Component (grün), erzeugt. Diese besitzt Eigenschaften wie eine Masse, ID oder Position. Spezifische Systemkomponenten wie die Küche, der Sitz oder die Waschräume erben von der generellen Systemkomponente alle Eigenschaften. Je nach Typ werden die Eigenschaften mit unterschiedlichen Werten ausgefüllt. Feste Werte, wie für den Typ oder die Zuordnung des ATA-Kapitels, können über die *redefine*-Funktion hinterlegt werden. Dadurch wird der Wert der geerbten Eigenschaft überschrieben. Zusätzlich können den Systemkomponenten Eigenschaften hinzugefügt werden, die andere Komponenten nicht haben (z.B. Seatpitch für den Sitz). Das Auffüllen der weiteren Werte erfolgt erst bei Ausführung des Modells während des Auslegungsprozesses. Für alle Werte werden die SI-Einheiten verwendet.

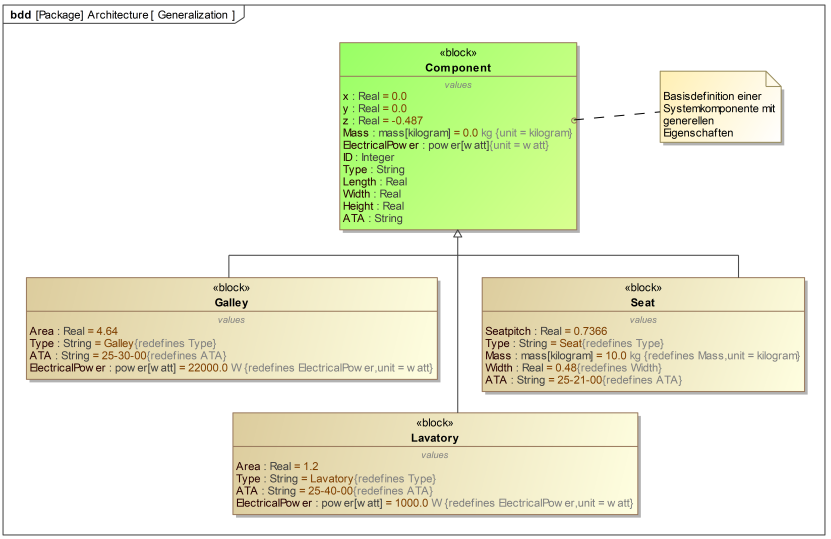



Abbildung 7.5: Blockdefinitionsdiagramm für die Vererbung von generellen Eigenschaften für die Systemkomponenten, generiert mit dem Cameo Systems Modeler.

Mit dem zweiten Blockdefinitonsdiagramm wird die Systemarchitektur beschrieben (Abbildung 7.6). Mit Hilfe der Komposition (Pfeil mit geschlossener Raute) wird die Architektur aufgebaut. Dabei gibt die Zahl neben dem Pfeil die Multiplizität an, wodurch bei Ausführung des SysML-Modells mehrere oder eine genau definierte Anzahl an

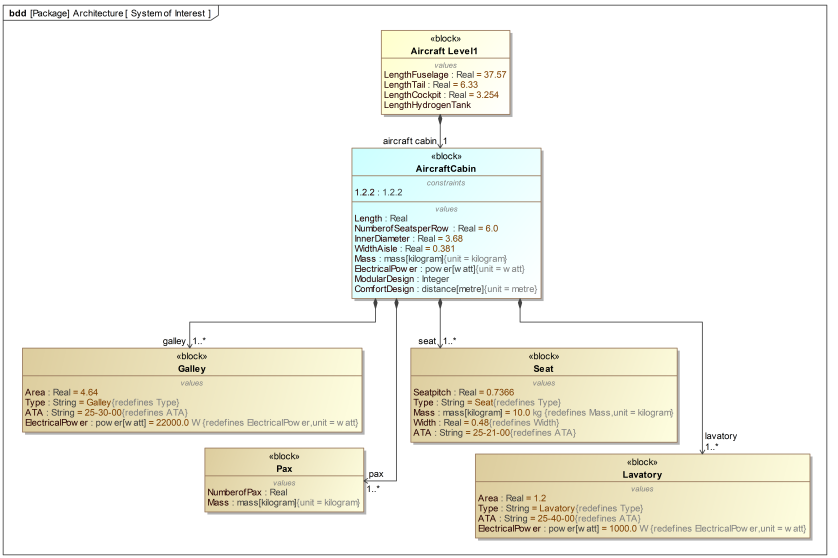



Abbildung 7.6: Blockdefinitionsdiagramm für den Aufbau der Systemarchitektur mit Hilfe der Komposition, generiert mit dem Cameo Systems Modeler.

Teilinstanzen automatisch erzeugt werden. Im gezeigten Beispiel besitzt das Flugzeug (Aircraft Level 1) bei Initialisierung genau eine Kabine (AircraftCabin). Die Kabine wiederum hat jeweils mindestens eine Küche, einen Passagier, einen Sitz und einen Waschraum. Die genaue Anzahl der Komponenten ist allerdings zum Zeitpunkt der Modellierung noch offen und kann während des Auslegungsprozesses eine beliebige Zahl >1 annehmen, symbolisiert durch das *. Die Anzahl der Waschräume, Sitze und Küchen ist abhängig von der Anzahl der zu befördernden Fluggäste, die wiederum durch den verfügbaren Baumraum der Kabine bestimmt wird. Je nach Anzahl befördernder Fluggäste variiert die Anzahl der Komponenten und damit die instanziierte Systemarchitektur.

#### 7.2.1.2 Anforderungsmodellierung und -überprüfung mit der SysML

Neben der Systemarchitektur werden im SysML-Modell die Anforderungen des Systems modelliert und überprüft. In der Luftfahrtindustrie liegen die Anforderungen überwiegend in Exceltabellen vor. Abbildung 7.7 zeigt den Ausschnitt einer Exceltabelle mit Anforderungen an das Subsystem Flugzeugkabine. Daher werden zuerst die Anforderungen automatisch über die Excel-Import/-Export-Funktion des Cameo Systems Modeler ausgelesen. Zeitgleich erzeugt der Cameo Systems Modeler Anforderungsstereotypen mit den entsprechenden Werten aus der Exceltabelle. Mit einem

Anforderungsdiagramm können die Anforderungen in Beziehung gesetzt werden. Abbildung 7.8 zeigt alle eingelesen Anforderungen und ihre Beziehung zueinander. Dabei stellt die Anforderung *Passenger Transport* eine übergeordnete qualitative Ausgangsanforderung an das Gesamtsystem Kabine dar, von der sich wiederum weitere quantifizierbare Anforderungen ableiten lassen. Zudem können die Anforderungen in einer Anforderungstabelle im CSM eingesehen und verwaltet werden.

Für die automatische Überprüfung der Anforderungen müssen diese mit den Eigenschaften der Systemkomponenten in Beziehung gesetzt werden. Abbildung 7.9 zeigt ein Anforderungsdiagramm mit der Zuordnung der Eigenschaften für die Überprüfung der Anforderungen. Mit Hilfe der satisfy-Beziehung wird zugeordnet, welches Modellelement die Anforderung erfüllt. Der Cameo Systems Modeler erkennt in dem Text der Anforderungen anhand fest definierter Schlagworte vor Zahlenwerten automatisch Bedingungen, die erfüllt werden müssen. Diese werden als quantifizierbare mathematische Regeln hinterlegt. Für die Anforderung mit der ID 1.1.1 (Anzahl der Passagiere) gilt die Bedingung, dass genau der Wert 180 erreicht werden soll. Den Vergleichswert erhält die Anforderung von der Klasse Pax. Der Wert der Blockeigenschaft NumberofPax wird für die Überprüfung mit dem Sollwert von 180 verwendet.

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | **ID** | **Name** | **Text** | **Owner** | **Verified by** | **Value** | **Unit** |
| 2 | 1.0.0 | Passenger Transport | The cabin shall provide a comfortable area for transporting people and provide services. | Requirements | | | - |
| 3 | 1.1.1 | Max. Number of passengers (Single Class) | The number of passengers shall be 180. | Requirements | | | Real |
| 4 | 1.2.1 | Design Payload | The Design payload shall be less than 25000 kg. | Requirements | | | kg |
| 5 | 1.2.2 | Hatrack Modularity | The cabin design shall enable a modular assembly of the hatracks. | Requirements | | | - |
| 6 | 1.3.1 | Total Cabin Energy Demand | The total energy of the cabin and furnishings, provided by the power system, shall be less than 75000W. | Requirements | | | W |
| 7 | 1.4.1 | Passenger Comfort | The accessibilty of the passenger service functions shall be less than 0.715 m. | Requirements | | | m |

Abbildung 7.7: Ausschnitt einer Exceltabelle mit Anforderungen an das Kabinensystem.

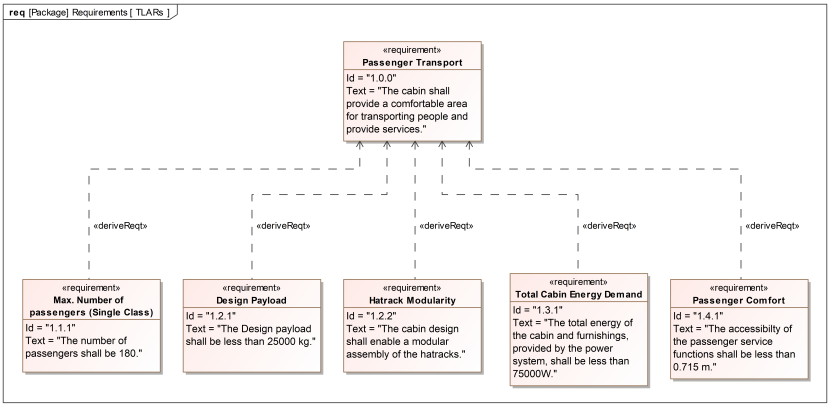



Abbildung 7.8: Abgeleitete weiterführende Anforderungen von der übergeordneten Systemanforderung *Passenger Transport* an das System Flugzeugkabine, generiert mit dem Cameo Systems Modeler.

Wenn aus dem Text einer Anforderung die quantifizierbare Bedingung nicht direkt ableitbar ist, wird diese manuell über einen Bedingungsblock (constraint) erzeugt. Das Vorgehen wird am Beispiel der Anforderung 1.2.2 veranschaulicht. Für die Erfüllung der Anforderung einer modularen Gepäckablage müssen alle zugehörigen Subkomponenten des Moduls innerhalb der Abmessungen der Gepäckablage angeordnet werden. Die Überprüfung dieser geometrischen Randbedingung findet während des Auslegungprozesses im Matlab-Modell statt. Im Anschluss an den geometrischen Entwurf übergibt Matlab einen einzigen Wert an das SysML-Modell zurück. Dieser wird im Objekt AircraftCabin unter der Eigenschaft modularDesign abgespeichert. Damit der Wert mit der Anforderung in Beziehung gesetzt werden kann, wird eine Bedingung (constraint 1.2.2) erzeugt, die einerseits ein Teil der Klasse AircraftCabin ist und andererseits für die Erfüllung der Anforderung HatrackModularity herangezogen wird. Die Bedingung zur Überprüfung ist im Bedingungsblock unter *contraints* hinterlegt. Die Bedingung ist erfüllt, wenn der Parameter ModularValue den Wert 1 hat. Im zusätzlich eingefügten Parametrikdiagramm der Klasse AircraftCabin ist die Zuordnung der Eigenschaft ModularDesign zu dem Parameter ModularValue des Bedingungsblocks dargestellt. Dadurch ist der Eigenschaftswert ModularDesign über die Bedingung an die Anforderung gekoppelt.

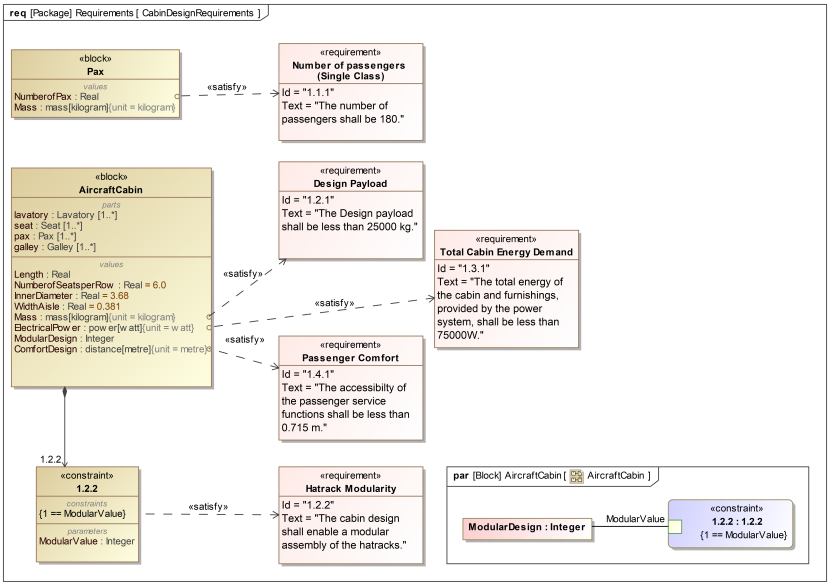



Abbildung 7.9: Anforderungsdiagramm für die Zuordnung der Kabinendesignanforderungen zu den Eigenschaften der Systemkomponenten, generiert mit dem Cameo Systems Modeler.

Die Ergebnisse der Anforderungsüberprüfung können im Anschluss an den Auslegungsprozess in einer Instanztabelle eingesehen werden. Abbildung 7.10 zeigt die Instanztabelle einer Kabinenkonfiguration. Im Tabellenkopf sind die Modellelemente aufgelistet, die mit Anforderungen verknüpft sind. In der zweiten Spalte sind die in-

stanziierten Objekte aufgeführt, zu denen die Modellelemente gehören. Jeder Eintrag zeigt den errechneten Wert des Modellelements. Wenn eine Anforderung erfüllt ist, wird das Feld des Eintrags grün eingefärbt. Ist eine Anforderung nicht erfüllt, wird das Feld des Eintrags rot eingefärbt. Bewegt man die Maus über den Eintrag, wird die zugeordnete Anforderung eingeblendet. Insgesamt kann dadurch ein schneller Überblick über die Erfüllung der Anforderungen einer Kabinenkonfiguration geschaffen werden.

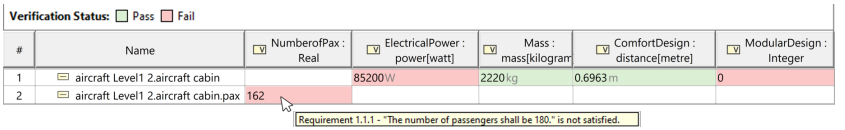



Abbildung 7.10: Instanztabelle der Kabine mit Ergebnissen für die Überprüfung der Anforderungen, generiert mit dem Cameo Systems Modeler und Anzeige der zugehörigen Anforderung. Grün hinterlegte Einträge symbolisieren eine erfüllte Anforderungen, rot hinterlegte Einträge nicht erfüllte Anforderungen.

Die Werte für die Objekteigenschaften werden zum Großteil in Matlab erzeugt. Einige Parameter berechnet das SysML-Modell selber und schreibt diese in die instanziierten Objekte. Der Prozess zum Import der Parameter aus dem Matlab-Modell wird in Abschnitt 7.3 vorgestellt.

### 7.2.2 Methodik für die Analyse und geometrische Auslegung in Matlab

Zusätzlich zu der Modellierung der Architektur in der SysML findet die Analyse und geometrische Auslegung der Komponenten in Matlab statt. In der in dieser Arbeit entwickelten Methode werden mit Matlab die vorliegenden Auslegungsdaten aus der Architekturmodellierung verarbeitet und vervollständigt. Im folgenden Abschnitt finden sich der entwickelte Ansatz, die Daten- und Modellstruktur sowie die Funktionalitäten innerhalb des Matlabmodells. Am Beispiel einer Flugzeugkabinenauslegung werden die Elemente erklärt. Die Beschreibung der Anbindung und Übergabe von Daten an dem Cameo Systems Modeler erfolgt in Abschnitt 7.3.

#### 7.2.2.1 Aufbau der Klassenstruktur mit dem objektorientierten Ansatz in der numerischen Rechenumgebung Matlab

Der Algorithmus in der Modellierungsumgebung Matlab für die konzeptionelle Auslegung der Systemkomponenten basiert auf dem Konzept der objektorientierten Programmierung (OOP). Grundidee des Konzepts ist die Orientierung der Programmstruktur an einem Modell der Wirklichkeit [Bec20]. Das Hauptelement bildet dabei ein Objekt und wird auf Grundlage einer Klassendefinition instanziiert [Bec20]. Die Entwicklung von Objekten erfolgt in der OOP unter anderem auf der Grundlage des Konzepts der Datenkapselung. Dabei werden die Implementierungsdetails verborgen, sodass auf die interne Datenstruktur nicht direkt zugegriffen werden kann. Dadurch

können Objekte nicht den internen Zustand anderer Objekte verändern oder lesen. Der Zugriff auf die interne Datenstruktur ist nur über definierte Schnittstellen möglich.

Ein Objekt kann Funktionen, Methoden oder Attribute besitzen, die in der Klasse definiert werden. Methoden beschreiben das Objektverhalten in einem vorgegebenen Programmkontext, während Attribute, oder auch Eigenschaften genannt, die Datenstruktur des Objekts festlegen [Bec20]. Analog zur Generalisierung in der SysML können hier ebenfalls abgeleitete Klassen die Methoden und Attribute der Basisklasse über das Konzept der Vererbung besitzen. Abbildung 7.11 zeigt einen Ausschnitt der Klassenstruktur am Beispiel der Kabinenkomponenten für die Auslegung in Matlab. Die Klasse cabinObject dient als Basisklasse, von der die meisten anderen Klassen erben. Sie enthält nur ein Attribut und eine Methode. Das Attribut ID ist ein Integer und ermöglicht für jedes instanziierte Objekt (ob Kabinenkomponente, Anforderung oder Link) die eindeutige Zuordnung und Identifikation. Mit der Methode generateID() wird eine ID automatisch bei Instanziierung des Objektes generiert. Die Oberklasse erbt wiederum von der von Matlab bereitgestellten, abstrakten Klasse *handle*, um der Handle-Semantik zu folgen und nützliche Methoden wie die Implementierung von get- und set-Methoden zu erben. Die weiteren Klassen cabinLink, Component und Requirement erben diese Features.

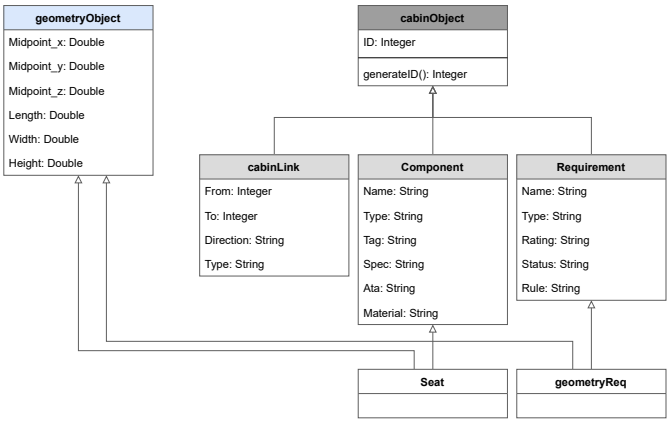



Abbildung 7.11: Ausschnitt der Klassenstruktur am Beispiel der Kabinenkomponenten.

Die Klasse Component beschreibt allgemein eine Komponente des abzubildenden Systems, von der alle Klassen für die Beschreibung der Kabinensystemkomponenten erben (z.B. Seat) [Bec20]. Sie enthält Attribute, die für alle Komponenten identisch sind. Dazu gehören der Name, der Typ und ein Material. Die Klasse Requirement beschreibt eine Anforderung. Diese besteht aus einem Namen, einem Typ, einer Bewertung, einem Status und einer Regel. Objektklassen, wie die geometryReq für geometrische Anforderungen, verfeinern die Anforderungsklasse und definieren spezifische Attribute und Methoden einer Anforderung. Die Beziehung zwischen zwei Kabinenkomponenten oder zwischen einer Kabinenkomponenten und einer Anforderung wird als Verknüpfung (Link) abgespeichert. Die Klasse cabinLink speichert die Beziehung

ab. In den Attributen werden die beiden verbundenen Elemente (From, To) sowie der Typ der Beziehung hinterlegt. Zusätzlich wird mit der Helferklasse geometryObject die Geometrie eines Objektes beschrieben und enthält die entsprechenden Attribute und Methoden [Bec20]. Für die Darstellung in Matlab werden einfache, geometrische Formen wie Rechtecke, Zylinder oder Kugeln verwendet. Diese sind für eine Evaluierung des Bauraums hinreichend genau und werden im weiteren Verlauf der Auslegung durch detaillierte Modelle ersetzt (siehe Abschnitt 7.2.3). Objekte mit einer definierten Geometrie, wie die Sitzkomponente, und die geometrische Anforderung, erben von dieser Klasse [Bec20].

In Abbildung 7.11 sind einige Attribute dargestellt. Bei der Speicherung der Daten werden unterschiedliche Datentypen verwendet. Die drei wesentlichen Datentypen sind String, Integer und Double. Ein String beschreibt eine Zeichenkette aus alphanumerischen Zeichen (z.B. „ATA-31“), ein Integer speichert ganzzahlige Werte (z.B. 54321) und mit Double werden Fließkommazahlen dargestellt (z.B 1,234) [Bec20]. Gesetzt werden die Werte in den Attributen unter anderem direkt durch die Objektmethoden. Zu den Objektmethoden gehören Visualisierungsfunktionen für die geometrische Darstellung eines Objektes und Instanziierungfunktionen für die Erzeugung einer Objektinstanz. Allgemein werden Klassennamen großgeschrieben, während bei Variablen- und Methodennamen Kleinbuchstaben verwendet werden. Der Aufbau der Objektklassen bzw. die Ontologie erfolgt in gleicher Weise zu der Struktur in den SysML-Modellen.

#### 7.2.2.2 Interne Modell- und Ordnerstruktur der numerischen Rechenumgebung Matlab

Die geometrische Auslegung und analytische Überprüfung eines Systems erfolgt in Matlab mit der definierten Klassenstruktur in Kombination mit Funktionsskripten. Damit die Auslegung der Systemkomponenten ebenfalls variabel stattfinden und leicht erweitert werden kann, wird eine modulare interne Modellstruktur in Anlehnung an die Modellstruktur im SysML-Modell gewählt. Abbildung 7.12 zeigt die interne Modellstruktur in Matlab. Das Hauptskript main.m startet den Auslegungsprozess. In diesem findet zum einen die Kopplung zur Modellierungsumgebung des Cameo Systems Modeler statt (siehe Abschnitt 7.3) und zum anderen die Einbindung weiterer Matlab-Ordner inklusive deren Skripte oder Funktionen für die Systemauslegung. Mit dem Befehl addpath() werden Ordner aus dem gleichen Speicherverzeichnis zum Suchpfad hinzugefügt, sodass auf deren untergeordnete Skripte o.ä. zugegriffen werden kann. Insgesamt werden drei Ordner eingebunden. Der erste Ordner beinhaltet alle definierten Objektklassen, basierend auf dem Konzept aus Abschnitt 7.2.2.1. Eine Klasse wird in Matlab mit einem orangefarbenen Quadrat symbolisiert. Ein Beispiel für eine Objektklasse in Matlab ist in Anhang A.3.1 dargestellt.

Der zweite Ordner enthält alle Funktionsskripte. Für eine bessere Übersicht, eine einfachere Pflege der Modelle und einen modularen Austausch von Systemgruppen, wird für jede Systemgruppe ein eigenes Funktionsskript definiert. In diesem werden die leeren Objekte aus der SysML-Architekturmodellierung weiter verarbeitet und z.B. geometrisch in dem vorhandenen Bauraum und im Bezug zu weiteren Kabinenkomponenten mit Auslegungsregeln platziert. Dabei werden entweder die bereits vorhandenen Instanzen verwendet oder neue erzeugt. Mit Hilfe der Funktionskapselung sind

zudem individuelle Anpassungen an den Subsystemen im Rahmen des Customizings möglich. Für eine Systemkomponente (Beispiel Gepäckablage) werden unterschiedliche Varianten hinterlegt und die Dimensionierung der Abmaße parametrisiert. Bei Auslegung der Komponente werden dadurch die Eingabeparameter eingelesen und die finale geometrische Modellierung angepasst. Ein Funktionsskript deklariert eine Funktion mit bestimmten Eingaben und Ausgaben. Ein Beispiel für den Aufbau eines Funktionsskript ist in Anhang A.3.2 dargestellt. Das Funktionsskript wird in Matlab durch ein $f_x$ symbolisiert.

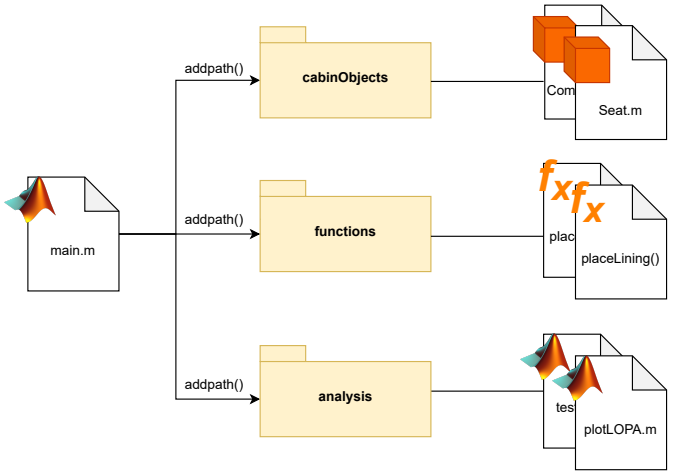



Abbildung 7.12: Schematische Darstellung der Modellstruktur in Matlab.

Der dritte Ordner enthält alle Skripte für die Analyse. Ein Skript ist durch das Matlab-Logo gekennzeichnet. Dazu gehören die Überprüfung der Anforderungen nach erfolgter Platzierung und Auslegung aller Systemkomponenten oder die Berechnung von Simulationen. Zusätzlich kann mit Visualisierungsfunktionen das Gesamtergebnis betrachtet werden. Damit kann entweder das System als 2D LOPA[12] und als 3D Layout dargestellt oder einzelne Aspekte aus den Anforderungsüberprüfungen und Simulationen visualisiert werden.

Während der Ausführung der Funktionen werden alle erzeugten Objektinstanzen bei Instanziierung in einem zentralen Speicher (Collection) abgelegt (Abbildung 7.13). Auf diesen Speicher kann von jedem Objekt und jedem Skript aus zugegriffen werden. Die Objektinstanzen bleiben in diesem zentralen Speicher über deren eindeutige ID weiterhin identifizierbar und referenzierbar [Bec20]. Dadurch kann auf die Objekte und deren Werte universal zugegriffen werden und diese für die Erstellung von Verbindungen, die Überprüfung von Anforderungen oder die Platzierung in Abhängigkeit einer anderen Komponente genutzt werden. Die Befehle für die Erzeugung eines Speichers und das Auslesen sind im folgenden Matlab-Code dargestellt.

[12]Layout of Passenger Accommodations: Bezeichnung der Konfiguration und Anordnung von Sitzen, Bordküchen, Toiletten und anderen Passagiereinrichtungen im Flugzeug.

```
1 global collection
2 collection = containers.Map;
3 test_a = collection.values;  % alle Objekte
4 test_b = collection.keys;   % alle IDs
5 collection(num2str(seats(1).ID))   % ein spezifisches Objekt
```

Abbildung 7.13: Matlab-Code für die Bildung eines zentralen Speichers und für den Zugriff auf Objektinstanzen.

Allgemein ist ein Map-Objekt eine Datenstruktur, mit der Werte (Value) über einen Schlüssel (Key) abgerufen werden können [The23]. Schlüssel können Zeichenvektoren oder reelle Zahlen sein und Werte können skalare oder nicht skalare Arrays sein [The23]. Als Schlüssel werden hier die IDs und als Werte die Objektinstanzen verwendet. Mit dem Befehl in Zeile 3 werden alle Werte, in diesem Fall alle Objektinstanzen, der Collection ausgegeben. Der Befehl in Zeile 4 liest alle Schlüssel und damit alle IDs aus. Mit dem Befehl in Zeile 5 kann auf ein spezifisches Objekt und dessen Eigenschaften zugegriffen werden. In diesem Fall wird das erste Sitzobjekt ausgelesen.

### 7.2.3 Visualisierung und Eigenschaftsabsicherung in 3D

Im Customizing-Prozess sind neben der geometrischen Platzierung der Komponenten in der Kabine weitere Aspekte relevant. Dazu gehören das räumliche Empfinden, die Optik, der Komfort und die Bedienbarkeit der Komponenten durch den Nutzer. Diese Kriterien können nicht ausreichend quantitativ mit Bewertungsformeln in SysML- oder Matlabmodellen beurteilt werden. Stattdessen müssen weitere Fachdisziplinen angeschlossen und im Rahmen einer Eigenschaftsabsicherung zusätzliche Faktoren der Systemauslegung untersucht werden, um die Bedürfnisse aller Stakeholder zu erfüllen. Eine Möglichkeit ist die frühe Anbindung und Präsentation der Analyseergebnisse in einer 3D-Darstellung oder in einer virtuellen Umgebung. In letzterer kann der Mensch mit Hilfe von virtueller Realität das Design des Gesamtsystems erleben und die Analyseergebnisse interpretieren und bewerten. Für die Visualisierung wird daher eine weitere Entwicklungsumgebung an die Auslegungskette angeschlossen. In dem folgenden Abschnitt wird ein Einblick in den Modellaufbau mit der Entwicklungsumgebung Blender gegeben.

#### 3D-Modellierung mit der 3D Grafiksoftware Blender

Für eine möglichst realistische, immersive Umgebung müssen in einem ersten Schritt detaillierte 3D Modelle für jede Systemkomponente erzeugt werden. Grundlage für die 3D Modellierung bilden die Ergebnisse aus der Matlab-Platzierung. Mit Hilfe der open-source 3D Grafiksoftware Blender werden die einfachen Geometriemodelle aus Matlab gegen hochauflösende 3D Geometriemodelle ausgetauscht [FGBN23]. Abbildung 7.14 zeigt im oberen Bild das 2D Layout einer Flugzeugkabine, in dem durch einfache geometrische Formen die Systeme und in grün farblich hervorgehoben ein Sitz dargestellt sind. Im unteren Bild ist das mit Blender generierte hochauflösende

3D Modell mit dem ausgetauschten Sitz dargestellt. Dafür werden die Objektinstanzen mit ihren Eigenschaftswerten aus Matlab exportiert und über eine XML-Datei an Blender übergeben. Ein Einblick in die Kopplung und den Datentransfer wird in Abschnitt 7.3 gegeben.

Bei der automatischen Modellgenerierung können entweder ebenfalls einfache geometrische Formen verwendet oder auf extern erstellte 3D Modelldateien zurückgegriffen werden. Für letzteres werden in einem Ordner für jede Systemkomponente separate Einzelmodelle gesammelt. Die Einzelmodelle können unter anderem von externen Partnern bereitgestellt, eigens entworfen oder mittels 3D Laserscanverfahren von Realgeometrien abgeleitet digitalisiert worden sein.

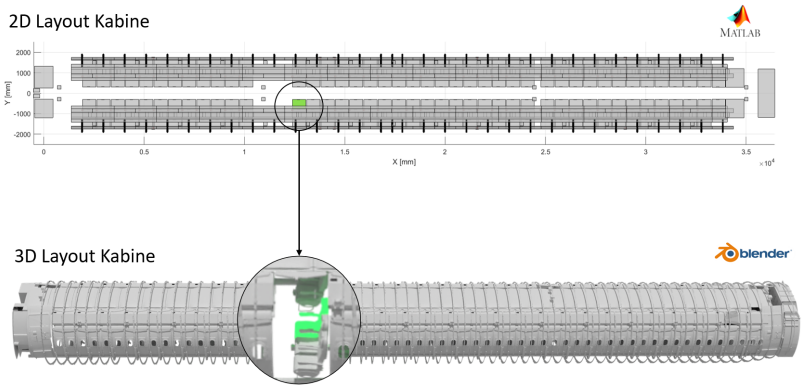



Abbildung 7.14: 3D Modellgenerierung in Blender auf den Objektdaten von Matlab.

Blender verfügt über eine Python-Schnittstelle, über die auf die Benutzeroberfläche zugegriffen werden kann [Bec20]. Insgesamt werden zwei Skripte für die Modellgenerierung mit Blender benötigt. Die mit der Programmiersprache Python verfasste Hauptdatei steuert die automatisierte Ausführung von Funktionen in Blender und speichert das Ergebnis anschließend in einem Zielordner für die Weiterverwendung mit der virtuellen Realität. Das gesamte Steuerungsskript ist in Anhang A.4.1 dargestellt. Zuerst liest das Pythonskript die Daten aus der XML-Datei aus und sucht alle Elemente mit dem Tag *component* [Bec20]. Anschließend wird überprüft, ob für jedes Element eine 3D Modellkonfiguration vorliegt. Dafür wird zusätzlich eine JSON-Konfigurationsdatei für die Modellgenerierung erstellt. JSON (JavaScript Objekt Notation) ist ein Datenformat und wird unter anderem bei der Entwicklung von Webanwendungen eingesetzt [Bec20]. Mit dieser Datei werden anhand der Typ-Eigenschaft jeder Systemkomponente die 3D Einzelmodelle zugeordnet. Abbildung 7.15 zeigt einen verkürzten beispielhaften Aufbau für die Konfigurationsdatei. Im Anhang A.4.2 ist der gesamte Quellcode für die Generierung eines Verkehrsflugzeugs aufgeführt.

Nach der Formatdefinition von JSON beginnt und endet ein Objekt mit geschweiften Klammern. Jedes Objekt kann durch Kommata getrennte Eigenschaften enthalten. Die Eigenschaften bestehen aus einem Schlüssel (keys) und einem Wert (values), voneinander getrennt durch einen Doppelpunkt. In der Konfigurationsdatei kann für

```
1  {
2    "Seat": {
3      "default": {
4      "meshType": "fbx",
5      "meshSource": "fbx/seat.fbx",
6      "transform": null
7      }
8    },
9    "Ohsc": {
10     "check": [
11     {
12       "condition": "Spec=RH",
13       "meshType": "fbx",
14       "meshSource": "fbx/hatrack.fbx",
15       "transform": [
16       {
17         "transformType": "rotation",
18         "rotationAxis": "Z",
19         "rotationAngle": 180
20       },
21       {
22       "transformType": "translation",
23       "translationX": 0,
24       "translationY": 307.1,
25       "translationZ": 30
26       }
27       ]
28     }
29     ],
30     "default": {
31       "meshType": "fbx",
32       "meshSource": "fbx/hatrack.fbx",
33       "transform": null
34     }
35   }
36 }
```

Abbildung 7.15: Beispiel JSON-Code für die Konfigurationsdatei und Zuordnung der 3D Einzelmodelle, modifiziert von [Bec20].

jeden Systemkomponententyp ein Eintrag als JSON-Objekt hinterlegt werden. Dieser besteht zuerst aus der Bezeichnung des Objekttyps, z.B. “Seat“ in Zeile 2, der dem Typ-Attribut aus den Matlab Objektinstanzen entspricht. Wenn in der Konfigurationsdatei für den Typ ein Eintrag hinterlegt ist, erfolgt die Zuweisung. Dabei kann eine weitere Überprüfung von Eigenschaften stattfinden. Anhand des Beispiels für den Sitz ist die Zuweisung einer Standardkonfiguration dargestellt. Bei dieser wird keine Translation oder Rotation vorgenommen. Stattdessen wird ein Einzelmodell mit dem Dateispeicherpfad fbx/seat.fbx und dem Datenformat fbx (filmbox) zugeordnet. Am Beispiel der Gepäckablage (Overhead Storage Compartment, OHSC) ist die Zuweisung von Einzelmodellen mit weiteren Bedingungen dargestellt. Im Bedingungsblock, eingeleitet mit dem Schlüssel “check“, können weitere Eigenschaften der Objekte abgefragt werden. In Zeile 12 wird die zusätzliche Bedingung abgefragt, auf welcher Flugzeugseite die Gepäckablage positioniert ist. Eingeleitet wird die Bedingung mit dem Schlüssel “condition“. In diesem Fall ist die Bedingung erfüllt, wenn das Objekt

auf der rechten Seite positioniert ist (right hand, RH). Trifft die Bedingung zu, werden in den Zeilen 17 bis 25 eine Translation und Rotation des Objektes vorgenommen. Ist die Bedingung nicht erfüllt, wird die Standardkonfiguration der Gepäckablage zugeordnet. Diese wird mit dem Schlüssel “default“ eingeleitet. Für den Fall, dass kein Eintrag in der Konfigurationsdatei für eine Systemkomponente gefunden wird, wird auf die Basiskonfiguration mit einem 3D Quader zurückgegriffen. Die Größe des Quaders ergibt sich aus den in der XML-Datei definierten Element-Eigenschaften, sodass der Quader entsprechend skaliert werden kann [Bec20].

Im Anschluss an die Zuordnung eines 3D Einzelmodells oder eines Quaders wird für die Systemkomponente die Position im dreidimensionalen Raum aus der XML-Datei ausgelesen. Die Werte werden an Blender übergeben und das 3D Einzelmodell an der entsprechenden Stelle platziert sowie die Transformationen auf das 3D Modell angewendet [Bec20]. Die Komponente erhält im letzten Schritt als Bezeichnung ihre ID-Nummer, um im weiteren Verlauf eine eindeutige Zuordnung zwischen den Objekteigenschaften und dem geometrischen Modell zu gewährleisten. Anschließend kann das 3D Gesamtmodell exportiert und für die Verwendung in weiteren Entwicklungsumgebungen (z.B. Unity für VR) bereitgestellt oder für die Veranschaulichung der generierten Variantenkonfiguration genutzt werden. Besonders im Rahmen des Customizings ermöglicht die 3D Darstellung der Einbausituation einer gewählten Konfiguration eine frühzeitige Beurteilung subjektiver Eigenschaften (z.B. Design).

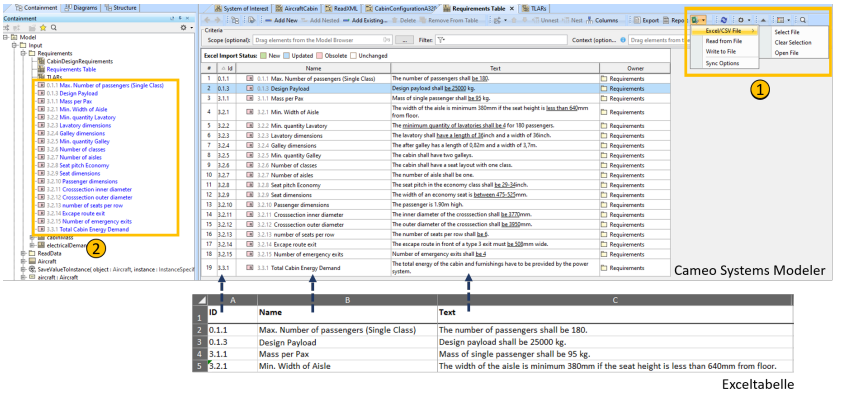
## 7.3 Interne Kopplung der heterogenen Modelle

Innerhalb eines Partialmodells werden die heterogenen Modelle miteinander gekoppelt. Dabei werden unterschiedliche Schnittstellentechnologien verwendet. Für das Beispiel der Kabinenauslegung müssen vier Tools miteinander gekoppelt werden. Dies umfasst jeweils die Kopplung zwischen Exceltabellen und Matlab mit dem Cameo Systems Modeler sowie die Anbindung von Blender und Matlab. Der folgende Abschnitt stellt den entwickelten Lösungsansatz zur Kopplung der heterogenen Modelle vor.

#### Kopplung Excel und Cameo Systems Modeler

Die erste interne Kopplung beschreibt den Austausch von Daten zwischen Exceltabellen und dem Cameo Systems Modeler. In der Luftfahrtindustrie liegen die Anforderungen häufig in Exceltabellen vor. Für die Modellierung der Anforderungen werden die dokumentierten Anforderungen daher aus einer Exceltabelle in die Modellierungsumgebung des Cameo Systems Modeler eingeladen. Dafür stellt CSM eine integrierte Import- und Exportschnittstelle bereit (siehe Abschnitt 5.2.2). Abbildung 7.16 zeigt eine Anforderungstabelle in der Modellierungsumgebung. Über den Excel-Button in der oberen Toolleiste kann eine beliebige Exceldatei ausgewählt und verknüpft werden. Mit dem Befehl *Read from File* werden die Zeilen und Spalten der verknüpften Exceltabelle ausgelesen und in die Anforderungstabelle überführt. Die Zuordnung der Spalten aus Excel zu den Spalten der Anforderungstabelle erfolgt einmalig manuell bei der Dateiauswahl. Parallel zum Ausfüllen der Anforderungstabelle im Cameo Systems Modeler werden für jede Anforderung ein Anforderungselement erzeugt. Diese

erscheinen dann im Containment-Baum. Änderungen an den Anforderungen oder die Ergebnisse nach der Anforderungsüberprüfung können über den Befehl *Write to File* wieder in die hinterlegte Exceltabelle zurückgeschrieben werden.




| | A | B | C |
|---|---|---|---|
| 1 | ID | Name | Text |
| 2 | 0.1.1 | Max. Number of passengers (Single Class) | The number of passengers shall be 180. |
| 3 | 0.1.3 | Design Payload | Design payload shall be 25000 kg. |
| 4 | 3.1.1 | Mass per Pax | Mass of single passenger shall be 95 kg. |
| 5 | 3.2.1 | Min. Width of Aisle | The width of the aisle is minimum 380mm if the seat height is less than 640mm from floor. |

Abbildung 7.16: Integrierte Anbindung von Daten aus Exceltabellen (1) und automatische Erzeugung von Anforderungselementen (2) im Cameo Systems Modeler.

### Kopplung Matlab und Cameo Systems Modeler

Die zweite Kopplungsart beschreibt den Datenaustausch zwischen Matlab und dem Cameo Systems Modeler. Hier wird ebenfalls eine von der Modellierungsumgebung bereitgestellte integrierte Schnittstelle genutzt. Die Anbindung zwischen den zwei Modellierungsumgebungen erfolgt mit der Opaque Aktion. Bei diesem Element kann als Sprache Matlab ausgewählt werden. Abbildung 7.17 zeigt ein Minimalbeispiel für den Datenaustausch und den Aufruf zum Ausführen des Matlabskipts mit dem Namen mainCabin. Zuerst wird mit der SysML eine Architektur erstellt und Instanzen (seats) erzeugt. Anschließend werden Werte an die zweite Opaque Aktion übergeben. Diese können entweder Variablen oder Objekte sein. In dem gezeigten Beispiel wird ein Wert für die Anzahl der Sitze (numberOfSeats) übergeben. Dieser Wert ist dadurch als Variable im Workspace von Matlab bekannt und kann für die Auslegung weiter Verwendung finden. Ausführen des Matlabskripts erfolgt mit dem Aufrauf run mainCabin.m in der Opaque Aktion. Abschließend werden die generierten Daten aus dem Workspace in Matlab an die SysML zurückgegeben.

Die Übergabe und Zuordnung der Werte zu den Attributen in den SysML Objektinstanzen erfolgt mit der dritten Opaque Aktion reimportMatlabData (Abbildung 7.17). Der Import der Daten aus dem Matlabworkspace und die Zuordnung zu den Attributen der SysML-Instanzen erfolgt in drei Schritten. Zuerst werden die zur Laufzeit der Simulation bestehenden Instanzen im Workspace des Cameo Systems Modeler mit der readSelf Aktion ausgelesen. Anschließend werden alle Sitzparameter aus Matlab als Sammlung in ein Array geschrieben oder direkt als Parameter übergeben. Zuletzt werden die Parameter den Attributen in den SysML Instanzen mit den ALH (Action Language Helper) Programmierschnittstellen-Befehlen zugeordnet.

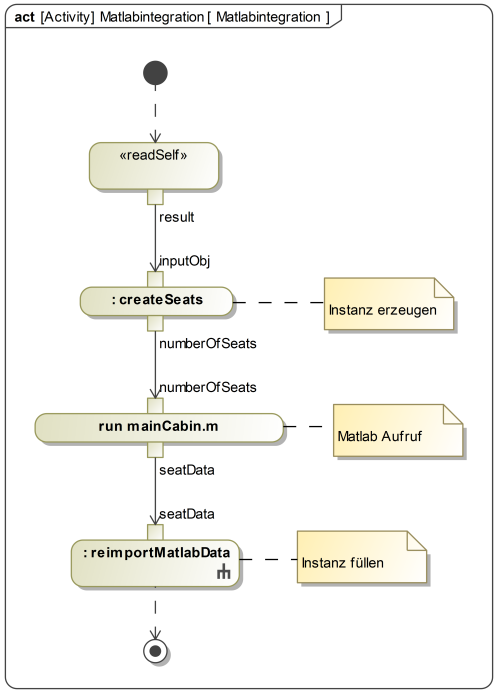



Abbildung 7.17: Minimalbeispiel für das Aufrufen eines Matlabskripts im Aktivitätsdiagramm mit dem Aktionselement Opaque Action.

Zusätzlich zur Anbindung auf der CSM-Seite muss das aufzurufende Matlabskript selbst mit dem Cameo Systems Modeler gekoppelt werden. Dies erfolgt entweder über den Matlabbefehl in Zeile 2 in Abbildung 7.18 in einer laufenden Session oder über das Hinzufügen der Matlabdateien zum selben Speicherverzeichnis. Das Skript mainCabin.m fungiert als Hauptsimulationsprogramm und steuert alle weiteren Funktionsskripte und Objekte an. Damit dieses die weiteren Skripte und Elemente findet, werden dessen Speicherpfade über die beispielhaft gezeigten Befehle in den Zeilen 4 und 5 eingebunden.

```
1 % CAMEO anbinden
2 matlab.engine.shareEngine
3 % Ordner hinzufügen
4 addpath('cabinObjects')
5 addpath('functions')
```

Abbildung 7.18: Matlab-Code für die Anbindung an den Cameo Systems Modeler und für das Hinzufügen weiterer Ordner.

Das Ergebnis der Datenrückführung ist für das Minimalbeispiel in Abbildung 7.19 dargestellt. Auf der linken Seite sind die vorerst leeren SysML Instanzen der Sitze dar-

gestellt. Auf der rechten Seite ist das Resultat nach der Auslegung in Matlab abgebildet. Die Daten wurden zurückgeführt und den Attributen der CSM Instanzen zugeordnet. Eine eindeutige Identifikation und Zuordnung der Daten aus den Matlab-Objektinstanzen zu den Cameo Systems Modeler Instanzen ist durch die ID möglich.

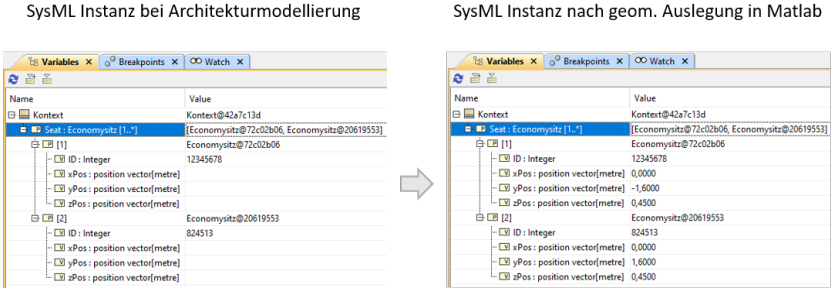



Abbildung 7.19: Minimalbeispiel für das Auffüllen von SysML-Objektinstanzen mit Matlabparametern.

### Kopplung Blender und Matlab

Mit der Verbindung zwischen dem SysML- und dem Matlab-Modell ist die Modellierung der Architektur und die geometrische Platzierung der Systemkomponenten abgedeckt. Für die Darstellung oder die Überprüfung weiterer Bewertungskriterien eines Systems kann zusätzlich die 3D Grafiksoftware Blender angekoppelt werden. Die Kopplung erfolgt ebenfalls über eine XML-Datei. Abbildung 7.20 zeigt die Kopplung für den Datenaustausch zwischen den Fachdisziplinen am Beispiel der Flugzeugkabine. Der gesamte Kopplungsprozess erfolgt teilautomatisiert.

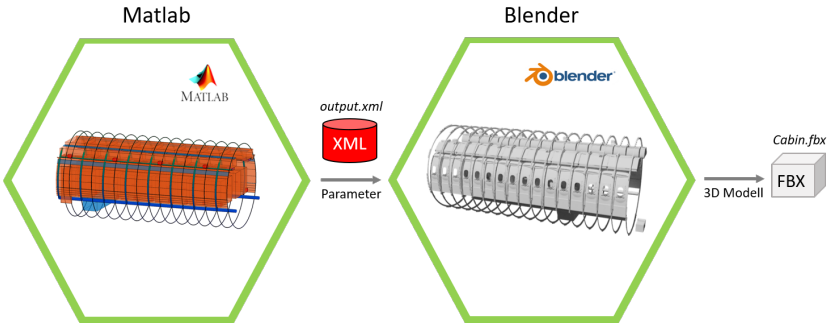



Abbildung 7.20: Visualisierung der Datenkopplung zwischen den Modellierungsumgebungen Matlab und Blender.

Der Prozess für die detaillierte Darstellung einer Kabinenkonfiguration startet im Anschluss an die Auslegung in Matlab. Die detailreiche Visualisierung der Kabinenkonfiguration benötigt neben den Basisinformationen, die im XML-Adapter hinterlegt sind,

weitere spezielle Eigenschaftswerte für die Visualisierung. Damit der XML-Adapter für die Kopplung der Partialmodelle nicht mit diesen ansonsten nicht notwendigen Parametern überfüllt wird, wird eine zusätzliche XML-Datei mit dem Namen *output* erzeugt und an Blender übergeben. In diese werden Parameter von vier Objektkategorien hinterlegt. In gleicher Weise wie bei dem XML-Adapter werden die Parameter, die Kabinenkomponenten und die Ergebnisse der Anforderungen gesammelt. Zusätzlich werden die Objektverbindungen zwischen den Kabinenkomponenten und zwischen Kabinenkomponenten und Anforderungen exportiert. Der Aufbau der XML-Datei orientiert sich an dem XML-Adapter zur Kopplung der Partialmodelle. Einblicke in den Aufbau einer Objektverbindung (Link) oder einer Anforderung gibt Abschnitt 7.2.2.1. Der Aufbau der XML-Datei ist Anhang A.4.3 zu entnehmen.

Die Ansteuerung des 3D Modellierungsprozesses in Blender auf Basis der Parameter aus Matlab erfolgt manuell über die Windows-Eingabeaufforderung. Dabei werden die Informationen über die Position, die Baumaße und des Typs jeder Komponente aus der XML-Datei eingelesen. Anschließend erfolgt die Zuordnung zu einem hochauflösenden 3D Modell. Dafür sind in einem Katalog-Ordner mehrere 3D Modelle für jeden Systemtyp hinterlegt. Die Zuordnung erfolgt über den Typ der Komponente und einer JSON-Konfigurationsdatei (s. Abschnitt 7.2.3). Das 3D Modell wird entsprechend der Baumaße skaliert und an der hinterlegten Position im dreidimensionalen Raum platziert. Die fertiggestellte Kabine wird im Filmbox-Datenformat unter dem Namen *Cabin.fbx* automatisch abgespeichert. Das 3D Modell kann entweder als Visualisierung einer Konfiguration für die Stakeholder verwendet werden oder in ein anderes Datenformat konvertiert und zusammen mit der XML-Datei für weiterführende Analysen in immersiven (VR) oder analytischen (Ansys) Umgebungen verwendet werden.

## 7.4 Kopplung mehrerer Partialmodelle miteinander

Bei der in dieser Arbeit entwickelten Methode erfolgt die Kopplung mehrerer Partialmodelle miteinander zu einem föderierten Modell über einen Adapter. Als Adapter fungiert eine XML-Datei, die dafür genutzt wird, Parameter zwischen den Partialmodellen auszutauschen. Dadurch werden die Modellierungsumgebungen miteinander in Beziehung gesetzt und die Ergebnisse der verschiedenen Systemdisziplinen können über die Modellierungsgrenze hinaus weiterverwendet werden. Dabei muss ein Partialmodell nicht immer aus einem SysML-Modell bestehen. Ein Partialmodell kann aus einem Modell einer einzigen Modellierungsumgebung (z.B. Matlab/Simulink) bestehen, solange dieses eine Schnittstelle zum Adapter aufweisen kann. In den folgenden Abschnitten werden der Aufbau der XML-Datei und die Strukturierung der SysML-Modelle für die Anbindung an die XML-Datei vorgestellt.

### 7.4.1 Aufbau des Adapters zur Kopplung partieller Modelle

Aufgabe des Adapters ist es, sowohl die Vernetzung der Partialmodelle miteinander als auch die Kopplung zwischen heterogenen Modellen (Abbildung 7.21) zu ermöglichen. Als Schnittstellentechnologie wird in dieser Arbeit eine XML-Datei verwendet. Die Auszeichnungssprache XML ist als Adapter gut geeignet, da diese von sehr vielen Programmen und Modellierungstools verarbeitet werden kann. Der Aufbau einer

XML-Datei ist standardisiert und kann immer nach dem gleichen Schema ausgelesen werden [Wös15, S. 31]. Zudem ist die Speicherung von Daten kaum begrenzt und bietet genügend Freiheiten bei der Bezeichnung der internen Strukturen und Datennamen. Für die Variantenmodellierung und das Customizing wird daher eine XML-Datei verwendet, um die Parameter zwischen den Partialmodellen auszutauschen und eine Gesamtauslegung eines Systems zu ermöglichen.

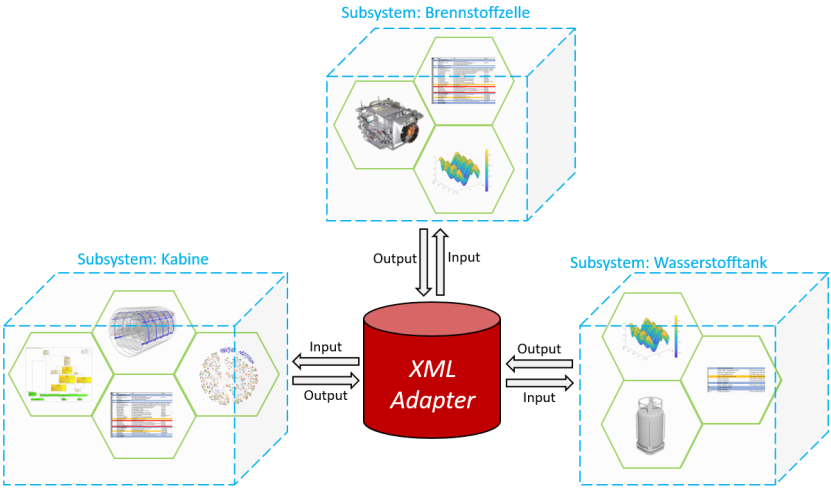
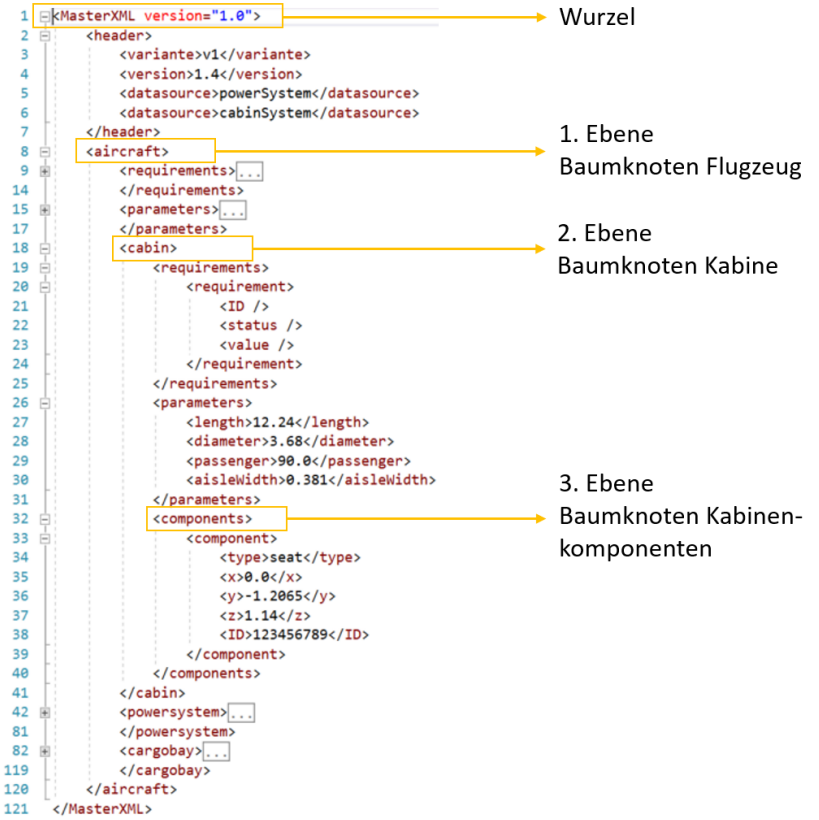



Abbildung 7.21: XML-Datei als Adapter zur Kopplung mehrerer Partialmodelle.

#### 7.4.1.1 Interne Struktur der XML-Datei

Der Aufbau der XML-Adapterdatei ist in Abbildung 7.22 dargestellt. In der ersten Zeile der Datei befindet sich die XML-Deklaration. Dazu gehört die erforderliche Angabe der Versionsnummer, auf der die XML-Spezifikation basiert. Mit der Deklaration können zudem weitere Informationen, wie die Zeichenkodierung, spezifiziert werden. Der nachfolgende Inhalt wird als logischer Aufbau beschrieben und entspricht einer hierarchisch, organisierten Baumstruktur. In dieser Struktur gibt es Baumknoten. Im Rahmen dieser Arbeit werden nur Elemente als Baumknoten verwendet. Ein Element kann entweder weitere Elemente oder eine Zeichenkette beinhalten. Für die Darstellung eines Elements werden Tags verwendet. Der Beginn eines Elements wird durch ein Starttag (<Name>) und das Ende durch ein Endtag (</Name>) gekennzeichnet. Ist der Inhalt eines Elements leer, wird das Element mit einem Leertag (<Name/>) dargestellt.




```
<MasterXML version="1.0">
    <header>
        <variante>v1</variante>
        <version>1.4</version>
        <datasource>powerSystem</datasource>
        <datasource>cabinSystem</datasource>
    </header>
    <aircraft>
        <requirements>...
        </requirements>
        <parameters>...
        </parameters>
        <cabin>
            <requirements>
                <requirement>
                    <ID />
                    <status />
                    <value />
                </requirement>
            </requirements>
            <parameters>
                <length>12.24</length>
                <diameter>3.68</diameter>
                <passenger>90.0</passenger>
                <aisleWidth>0.381</aisleWidth>
            </parameters>
            <components>
                <component>
                    <type>seat</type>
                    <x>0.0</x>
                    <y>-1.2065</y>
                    <z>1.14</z>
                    <ID>123456789</ID>
                </component>
            </components>
        </cabin>
        <powersystem>...
        </powersystem>
        <cargobay>...
        </cargobay>
    </aircraft>
</MasterXML>
```

Abbildung 7.22: Interne Struktur der XML-Adapterdatei.

Das Element auf oberster Ebene und damit die Wurzel der Struktur bildet die Bezeichnung der XML-Datei (MasterXML). Anschließend folgen auf gleicher Ebene jeweils ein Elementsatz für den Header und für das Gesamtsystem (hier Flugzeug). Im Header werden der Name der zu untersuchenden Variante, die Versionsnummer der Datei und die Namen der angesteuerten Partialmodelle gespeichert. Die Versionsnummer wird im Rahmen der Versionierung verwendet und ändert sich entsprechend durchgeführter Iterationen, wodurch eine eindeutige Zuordnung der Variante gewährleistet ist. Der Baumknoten des Flugzeugs (aircraft) unterteilt sich wiederum in Anforderungen, Parameter und die Subsysteme (Kabine, Energiesystem, Frachtraum). Sowohl die Anforderungen als auch die Parameter beziehen sich auf das Gesamtsystem Flugzeug. Die Subsysteme verfügen über eigene Unterbaumknoten zu den beiden Kategorien.

Ein Subsystem kann ebenfalls in weitere Subsysteme unterteilt werden. Der Aufbau aller Systeme ist gleich und unterteilt sich in drei Kategorien. Zuerst werden Daten für die Anforderungen hinterlegt, dann Parameter und anschließend die Systemkomponenten. Am Beispiel der Kabine ist der interne Aufbau der drei Kategorien dargestellt.

In einem Partialmodell können bereits systemspezifische Anforderungen überprüft werden. Die Ergebnisse können ebenfalls in die XML-Datei überführt werden. Für jede Anforderung wird ein Anforderungselement <requirement> erzeugt. Dieses beinhaltet weitere Elemente für die ID, für den Status und für einen Wert. Im Statuselement wird hinterlegt, ob eine Anforderung erfüllt oder nicht erfüllt wurde. Im Wertelement wird zusätzlich das Ergebnis aus der Anforderungsüberprüfung (z.B. Gewicht) eingetragen. Der Wert kann sowohl quantitativ (Zahlenwert) als auch qualitativ (boolean true/false) sein. In der zweiten Kategorie im Baumknoten <parameters> werden die allgemeingültigen Parameter für das verknüpfte Subsystem hinterlegt. Für das Beispiel der Kabine sind dies unter anderem die Kabinenlänge, der Kabinendurchmesser, die Anzahl der Passagiere und die Gangbreite. Die Systemkomponenten werden unter dem Baumknoten der dritten Kategorie als <component> abgespeichert. Eine Komponente ist äquivalent zu den instanziierten Objekten in den Modellen. Ein Objekt wird beschrieben durch eine geometrische Repräsentation (x,y,z), eine ID, Eigenschaftswerte und weitere Attribute (z.B. type). Sowohl für die Anforderungen als auch für die Komponenten ist eine eindeutige Identifikation der Objekte mit der universal gültigen Identifikationsnummer möglich. Dadurch ist eine Zuordnung der Daten über die Grenzen der Partialmodelle hinaus sichergestellt.

#### 7.4.1.2 Grundlegende Eigenschaften der XML-Datei

XML-Dateien bringen spezielle Eigenschaften mit sich, die für die in dieser Arbeit erarbeitete Methode die Kopplung von Partialmodellen durch einen Parametertransfer umzusetzen. Die Auszeichnungssprache XML ermöglicht die Darstellung hierarchisch strukturierter Daten in einer flexiblen Textdatei. Mit einer XML-Datei wird eine für den Menschen und die Maschine offene, verständliche Informationslandschaft geschaffen [Wor16]. Dennoch liegt der Fokus bei dem Einsatz der XML-Datei auf der Datenverarbeitung durch einen Computer. Parameter- bzw. Elementbezeichnungen müssen unmissverständlich von den unterschiedlichen Modellierungstools und deren Programmiersprachen interpretiert und ausgelesen werden. Die Tools stellen dafür entweder einen integrierten Parser bereit oder können über weitere vorgeschaltete Schnittstellen die Daten lesen und verarbeiten. In der XML-Datei wird eine einheitliche Bezeichnung verwendet. Die Elementbezeichnungen in der Baumstruktur beginnen mit einem Kleinbuchstaben und werden ohne Sonderzeichen oder Leerzeichen geschrieben. Reicht ein Wort für die Bezeichnung nicht aus, wird das nachfolgende Wort mit einem Großbuchstaben ohne Leerzeichen in der Zeichenkette ergänzt. Die Namen müssen aussagekräftig und unterscheidbar sein, um fehleranfällige Konstrukte und Doppelungen zu vermeiden.

Zu Beginn der Auslegung einer Variante wird eine leere MasterXML-Datei aufgesetzt. Diese enthält nur die Baumknoten Header und Aircraft. Im Header werden bei Initialisierung jeweils ein Eintrag für die Variante und für die Version hinterlegt. Für den Knoten Aircraft werden nur die weiteren Elementknoten für die Anforderungen und Parameter hinterlegt. Die Subsysteme werden erst mit der Ausführung des Partialmodells angehängt. Das Partialmodell füllt die Elemente und deren Werte mit den generierten Daten aus. Dabei werden weitere Elemente in der für das Subsystem vorgegebenen Ebene angehängt. Im Laufe des Prozesses werden keine Baumknoten oder Elemente gelöscht. Ausschließlich im Rahmen einer Iteration können bereits

bestehende Elementeinträge überschrieben werden. Bei der Übertragung der Daten in die XML-Datei müssen nicht alle Informationen aus dem Partialmodell über die Modellgrenzen hinaus geteilt werden. Modelldisziplinspezifische Daten, die im Rahmen des Subsystemkontext für die Analyse und Auslegung notwendig sind, aber keine Abhängigkeit zu weiteren externen Subsystemen haben und für die weitere Betrachtung des Gesamtsystems nicht notwendig sind, werden nicht exportiert.

XML-Dateien zeichnen sich durch ihre Flexibilität aus, unterschiedlichste Datentypen im Vergleich zu anderen Formaten speichern zu können [BGG+18]. Bei der Speicherung der Daten als Elemente werden allerdings aufgrund der internen Baumstruktur viele redundante Tags verwendet [Mae12, BGG+18]. In der Studie von Breje et al. wurden Datenübermittlungsmethoden für XML- und JSON-Modelle verglichen. Anhand der zwei Kriterien Antwortgeschwindigkeit (in Sekunden, s) und Größe der empfangenen Daten (in Kilobyte, KB) wurden die beiden Formate analysiert. Bei der Verarbeitung von 1000 Elementen ergab die Untersuchung bei dem XML-Format eine Antwortzeit von 1,03 s und eine Dateigröße von 337 KB [BGG+18]. Damit ist das Auslesen von Daten aus einer XML-Datei langsamer im Vergleich zu einer JSON-Datei (0,49 s). Für die Kopplung der Partialmodelle wird eine XML-Datei im Kilobyte- bzw. im unteren einstelligen Megabyte-Bereich verwendet. Zusätzlich wird die XML-Datei bei der Bearbeitung der Daten einmalig eingelesen und die Elemente gebündelt zurück geschrieben, um die Verarbeitung auch bei größeren Datenmengen durch den Arbeitsspeicher performant zu halten. Zudem werden in jedem Partialmodell wenige hundert Elemente verarbeitet, sodass sich die Bearbeitungszeit im Sekundenbereich bewegt. Die geringfügig schnellere Verarbeitung der Daten in anderen Formaten ist vernachlässigbar, da die Vorteile einer flexiblen Struktur, der Speicherung unterschiedlicher Datentypen und die Anbindung durch Schnittstellen an viele Modellierungsumgebungen überwiegen. Die in dieser Arbeit entwickelte Methodik ist dadurch auf größere Datenmengen skalierbar, sodass weitere Partialmodelle gekoppelt und die Daten verarbeitet werden können. Zusammenfassend bietet die Verwendung einer XML-Datei leistungsfähige Verfahren für die Speicherung und Verarbeitung von Datenmengen.

### 7.4.2 Strukturierung der SysML-Modelle nach dem EVA-Prinzip

Für die Strukturierung der SysML-Modelle wird das EVA-Prinzip angewendet, welches bereits bei der PAKo-Methode (Abschnitt 4.4.2) im Konzeptentwicklungsprozess Anwendung findet [Tec08, S. 6]. Analog zur Ordnerstruktur bei der PAKo-Methode in der Konstruktion, wird das EVA-Prinzip bei der Modellierung mit der SysML ebenfalls angewendet. Ziel dieser Strukturierung ist der einheitliche Aufbau jedes SysML-Modells. Dies fördert das Systemverständnis und unterstützt eine einheitliche Modellierung der jeweiligen Subsysteme. Zusätzlich schafft diese Struktur eine klare Trennung zwischen den Eingabeparametern und der Modellierung. Die Abhängigkeiten und weitere Verarbeitung der Eingangsparameter ist klar definiert und in jedem Modell gleich integriert. Anpassungen an den Schnittstellen können somit einfacher durchgeführt werden, ohne in der verschachtelten Modellierung zu suchen. Eine Beschreibung der Inhalte erfolgt in den folgenden drei Abschnitten am Beispiel des SysML-Modells einer Flugzeugkabine.

#### 7.4.2.1 *Eingabe*: Import externer Daten aus XML- und Excel-Dateien

Im Ordner Input befinden sich die Elemente für den Import externer Parameter. Abbildung 7.23 zeigt die Ordnerstruktur für die Eingabe in einem SysML-Modell. Als Modellierungstool wird der Cameo Systems Modeler verwendet. Innerhalb des Ordners *Input* befinden sich weitere Unterordner, auch Packages genannt. Diese sind jeweils ein Ordner für die Anforderungen (*Requirements*) und zum Auslesen der Parameter aus der XML-Datei (*ReadData*). Daneben gibt es einen Block *Aircraft* und eine Instanz *aircraft* vom Typ Aircraft.

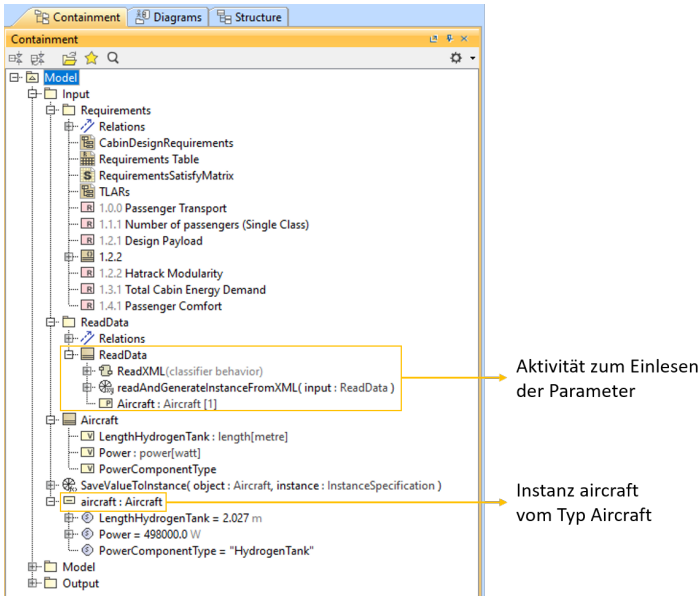



Abbildung 7.23: Ordnerstruktur für die Eingabe (Input) der SysML-Modelle mit dem Cameo Systems Modeler.

Im Requirements-Package sind alle Elemente, Diagramme und Tabellen für die Erstellung von Anforderungen hinterlegt. Dazu gehören die Anforderungstabelle, das Anforderungsdiagramm und die Elemente Anforderung und Bedingung. Die Anforderungen werden extern über eine Exceltabelle eingeladen. Bei Bedarf können die Anforderungen auch manuell erzeugt werden. Bei dem Import von Anforderungen über die Anforderungstabelle aus einer Exceltabelle werden automatisch Anforderungselemente erzeugt. Diese erscheinen dann im Containment-Baum und können unter anderem mit dem Anforderungsdiagramm den Systemelementen zugeordnet werden. Dadurch können Anforderungen auf ihre Erfüllung hin überprüft werden. Der Cameo Systems Modeler bietet die Möglichkeit einer direkten Verknüpfung von Anforderungen mit Attributen der Objektblöcke. Dadurch können quantitative Bedingungen, wie die maximale Passagieranzahl eines Verkehrsflugzeugs, direkt überprüft werden. Zusätzlich besteht die Option mit dem Element der Bedingung weitere Regeln und Formeln zu erzeugen, mit denen die Erfüllung einer Anforderung untersucht werden kann.

Wird beispielsweise als Anforderung vorgegeben, dass das Gewicht des Gesamtsystems 150 kg nicht überschreiten darf, müssen die Gewichte aller Subkomponenten aufsummiert und das Ergebnis an die Anforderung übertragen werden. Mit Fertigstellung der Systemauslegung lassen sich die Ergebnisse der Anforderungstabelle in die verknüpfte Exceltabelle exportieren.

Für das Auslesen von Parametern aus der XML-Datei werden der ReadData-Ordner sowie der Block *Aircraft* und die Instanz *aircraft* benötigt. Der Block Aircraft beschreibt das Objekt Flugzeug. Es besitzt drei für die Kabinenauslegung relevante Werte (values). Diese sind die Länge des Wasserstofftanks, die Leistung und der Komponententyp des Stromversorgungssystems. Im ReadData Ordner definiert der Block *ReadData* den Kontext. In dem Kontext befinden sich das Aktivitätsdiagramm *ReadXML*, die Opaque Aktion *readAndGenerateInstanceFromXML* und die Teileigenschaft (Part Property) *Aircraft*. Der Import der Daten aus der XML-Datei findet mit dem Aktivitätsdiagramm *ReadXML* statt. Abbildung 7.24 zeigt das Diagramm.

Die Aktivität startet mit der Read-Self-Aktion. Bei dieser werden alle Objekte zurückgegeben, welche im Kontext der Aktivität dargestellt sind. In diesem Fall ist es das Objekt Aircraft vom Typ Aircraft. Durch die Beziehung zum vorher definierten Objekt Aircraft, enthält die Teileigenschaft automatisch die definierten Values LengthHydrogenTank, Power und PowerComponentType. Somit existiert bei Ausführung des Diagramms ein instanziiertes Objekt vom Typ Aircraft mit drei leeren Values. Anschließend wird das leere Objekt als input an die nächste Aktivität übergeben.

Bei der readAndGenerateInstanceFromXML Aktivität handelt es sich um eine Opaque Aktion. Der Programmcode der Opaque Aktion ist in Anhang A.5.1 dargestellt. Geschrieben ist der Code in der Programmiersprache Python. Zuerst wird mit dem Code die XML-Datei als Baumstruktur eingelesen. Innerhalb der Struktur werden Elemente/ Parameter als einzelne Knoten dargestellt. Die benötigten Parameter können dadurch gesucht und an die Instanz in Cameo übergeben werden. Die beiden vorerst leeren Values werden mit den Parameterwerten aufgefüllt. Abschließend wird die Instanz automatisch im Containment-Baum abgespeichert, damit sie im weiteren Verlauf der Modellierung verwendet werden kann (siehe Abbildung 7.23 aircraft:Aircraft).

Die automatische Speicherung der Instanz und deren Eigenschaften erfolgt mit einer Call Behavior Action. Zuerst wird mit der «readSelf» Aktion das generierte Objekt im Kontext der Aktivität ausgelesen. Anschließend erfolgt in einem parallelen Schritt die Übergabe des Objekts (1) und die Übergabe der Instanz für die Speicherung (2). Mit dem fUML-Befehl saveValueToInstance(object,instance) werden dann die Laufzeitwerte des angegebenen Objekts in den Slots der angegeben Instanz aircraft gespeichert. In dem gezeigten Beispiel aus Abbildung 7.23 ergeben sich für die Instanz aircraft die Werte HydrogenTank (PowerComponentType), 2,027 m (LengthHydrogenTank) und 498000 W (Power).

Die abgespeicherte Instanz aircraft mit den ausgefüllten Eigenschaftswerten kann im Anschluss für die weitere Auslegung der Kabine genutzt werden. Bei Änderungen der Werte in der XML-Datei werden die Werte der Instanz neu überschrieben. Dadurch kann die Instanz als festes Element in der Modellierung weiterverwendet werden, ohne dass eine manuelle Änderung notwendig ist.

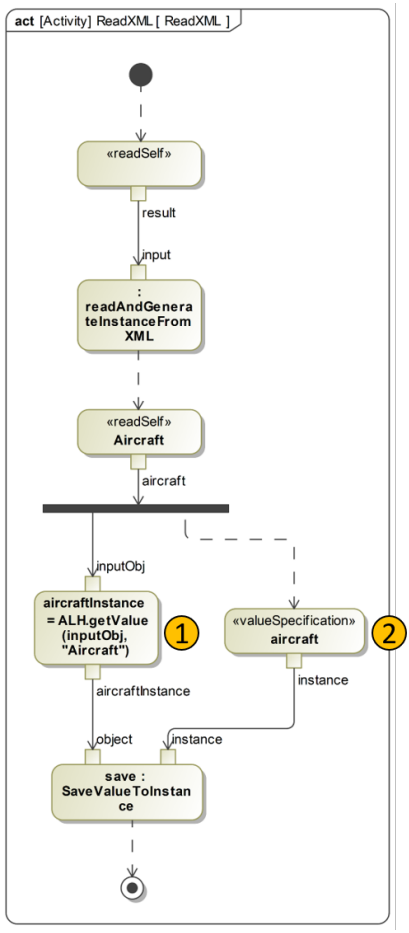



Abbildung 7.24: Aktivitätsdiagramm zum Auslesen der Parameter aus der XML-Datei. Die automatische Speicherung der gelesenen Werte erfolgt durch das Auslesen der neu generierten Instanz aircraftInstance (1) und Zuordnung zur Speicherinstanz aircraft (2).

#### 7.4.2.2 *Verarbeitung*: Architekturmodellierung, funktionale Auslegung und Instanziierung von Objekten

Im Ordner Model befinden sich alle Berechnungen, Elemente, Diagramme und Programmiercodes, die für die Modellierung des untersuchenden Systems erstellt werden. Am Beispiel der Flugzeugkabine werden einige Diagramme und Elemente vorgestellt. Abbildung 7.25 zeigt die Ordnerstruktur für die Verarbeitung (Model). Der Ordner Structure beinhaltet alle Diagramme und Elemente, mit denen das zu untersuchende System abgebildet wird. Eine Einteilung innerhalb verfeinert die abzubil-

denden Eigenschaften. Im Ordner System Context befinden sich alle Elemente für die Beschreibung des Systemkontext. Dazu gehören Anwendungsfalldiagramme. Im Architecture Ordner wird die Systemarchitektur beschrieben. Mit Hilfe von Blockdefinitionsdiagrammen werden das zu designende und analysierende System (System of Interest, SoI) und dessen Vererbungen modelliert. Zudem werden mit dem Architekturelement Block die einzelnen Subkomponenten (z.B. Seat) des Systems Kabine und deren Eigenschaften sowie die Beziehungen untereinander modelliert. Darüber hinaus werden Parametrikdiagramme und Bedingungen (Constraints) für die Verknüpfung von Systemeigenschaften mit Anforderungen in dem Ordner abgelegt.

Die dynamische Auslegung der Kabine erfolgt mit einem Aktivitätsdiagramm und dem Element Opaque Aktion. Die für die Simulation benötigten Diagramme und Elemente befinden sich im Ordner Execution. Gleichartig zum Vorgehen beim Import der Daten aus der XML wird ein Kontext (A320) definiert, in dem die Simulation stattfindet und Instanzen generiert werden. Die instanziierte Systemarchitektur sowie die Eigenschaftswerte in den einzelnen Instanzen basieren auf der Modellierung und den Diagrammen aus dem Ordner Structure. Ausgangspunkt für die Modellierung bildet die generierte Instanz mit den importierten Parametern aus der XML-Datei aus dem Input-Ordner. Deren Parameter werden zu Beginn wieder mit der Lese-Aktion value Specification ausgelesen und im Rahmen der Simulation als Parameter für weitere Kalkulationen (z.B. für den Abgleich der Leistung) übergeben.

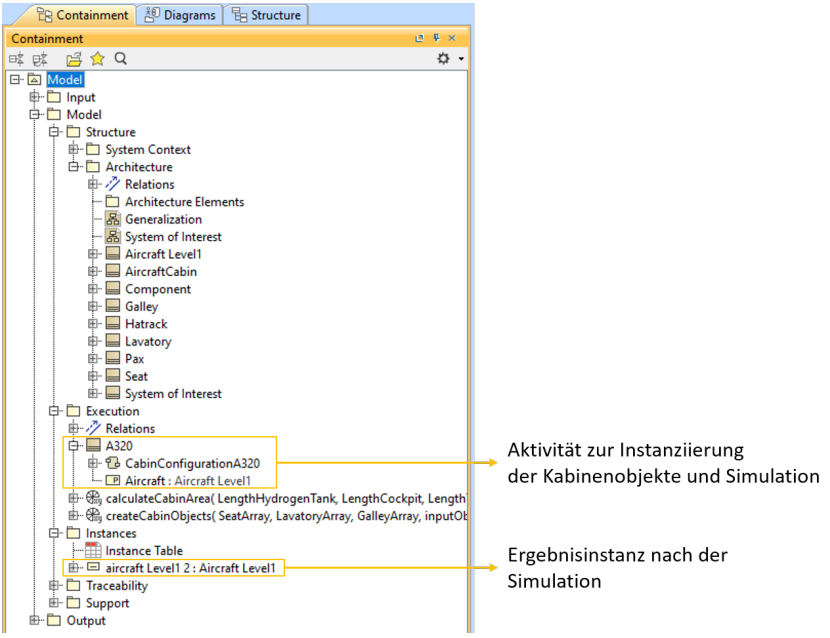



Abbildung 7.25: Ordnerstruktur für die Verarbeitung (Model) der SysML-Modelle mit dem Cameo Systems Modeler.

Bei der Simulation des SoI wird dieses als Objekt zur Laufzeit instanziiert. Damit die Ergebnisse der Simulation noch nach Ende der Simulation einsehbar und für die XML-Datei exportierbar sind, werden die erzeugten Instanzen am Ende der Ausführung automatisch im Ordner Instances abgespeichert. Die automatisierte Speicherung der Instanzen erfolgt entweder mit dem Befehl saveValueToInstance() oder mit der Simulationskonfiguration und ist in Abschnitt 7.4.2.1 erklärt, oder kann manuell dem Speicherort zugewiesen und abgespeichert werden. Zusätzlich ist in dem Ordner die Instanztabelle mit einer Übersicht der Ergebnisse hinterlegt.

Im Ordner Traceability sind Rückverfolgbarkeitsmatrizen abgelegt. Mit diesen lassen sich zum Beispiel Beziehungen zwischen Anforderungen und anderen Entwurfselementen analysieren oder ändern. Zudem können Funktionsblöcke mit entsprechenden Aktivitäten verknüpft und zurückverfolgt werden. Diese Matrizen oder Tabellen ermöglichen eine Durchführung von Anforderungslücken- und Vollständigkeitsanalysen [No 22a].

Im letzten Ordner Support finden sich Elemente wieder, die keinen direkten Beitrag zu der Modellierung des Systems leisten. Dazu gehört unter anderem das Element Simulationskonfiguration. Mit diesem können Aktivitätsdiagramme automatisiert ausgeführt und Instanzen an festgelegten Orten im Containment-Baum nach Ausführung der Simulation abgespeichert werden.

#### 7.4.2.3 *Ausgabe*: Export der Daten in die XML-Datei

Im Ordner Ausgabe sind alle Modellelemente für den Export der Daten in die XML-Datei abgelegt. Der Export der Daten erfolgt mit einem Aktivitätsdiagramm. Der Programmcode für das Erzeugen neuer Einträge oder das Überschreiben bereits vorhandener Einträge in der XML wird mit dem Modellelement Opaque Aktion erzeugt. Abbildung 7.26 zeigt die Ordnerstruktur für die Ausgabe im Cameo Systems Modeler mit dem Aktivitätsdiagramm und den drei Opaque Aktionen.

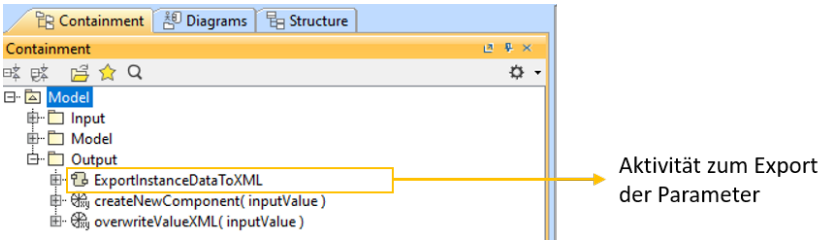



Abbildung 7.26: Ordnerstruktur für die Ausgabe (Output) der SysML-Modelle mit dem Cameo Systems Modeler.

Die Abbildung 7.27 zeigt das Aktivitätsdiagramm für den Export der Daten in die XML-Datei. Begonnen wird die Aktivität mit dem Auslesen der generierten Instanz aus der Modellarchitekturauslegung. Anschließend erfolgt das Lesen der Strukturmerkmale mit dem zweiten Aktionselement. Dadurch können Parts einer Instanz ausgelesen werden. Bei der Architekturmodellierung werden in einer Instanz mehrere Subparts,

stellvertretend für die Subkomponenten, erzeugt. Dadurch besitzt beispielsweise eine Kabineninstanz mehrere Parts vom Typ Sitz und vom Typ Galley. Mit dem Aktionselement können beliebig verschachtelte Instanzen ausgelesen und bestimmte Parts angesteuert werden. Sobald die gewünschte Ebene der Subkomponente erreicht wird, erfolgt der Export der Daten in die XML-Datei. Dafür wird der generische Programmcode createNewComponent und die Programmiersprache Python verwendet. Mit diesem wird zuerst die XML-Datei als Baumelement eingeladen. Im nächsten Schritt wird der entsprechende lokale Pfad in der XML-Datei für die Platzierung der Komponente je nach Partialmodell gesucht (z.B. //cabin//components für das Kabinenmodell). Anschließend werden alle Parts der Cameo-Instanz bzw. dessen Parameter ausgelesen und als component in die XML-Datei geschrieben. Die erzeugten, neuen Elemente werden in der XML-Datei hintereinander angehängt. Ein detaillierter Einblick in den Aufbau der XML-Datei ist in Abschnitt 7.4.1 gegeben. Dieser Vorgang wird für alle Parts wiederholt und findet nacheinander statt.

Im nächsten Schritt des Exports lassen sich Daten in der XML-Datei überschreiben. Mit dem Programmcode overwriteValueXML kann jeder beliebige Baumknoten in der XML-Datei gesucht und anschließend das anhängende Element mit einem neuen Wert überschrieben werden. Dies wird beispielsweise genutzt, um bei den übergeordneten Parametern der Kabine die neu berechnete Anzahl an Passagieren und die innere Kabinenlänge einzutragen. Zusätzlich wird im Header der XML-Datei als Datenquelle der Name des ausgeführten Partialmodells hinterlegt, um für die weitere Verwendung der Daten kenntlich zu machen, welche Partialmodelle die bislang hinterlegten Daten generiert haben. In Anhang A.5.2 sind die beiden Programmcodes im Detail aufgeführt.

Der Export der Anforderungen und deren Ergebnisse kann auf unterschiedliche Weise erfolgen. Insgesamt gibt es drei Möglichkeiten. Die erste sieht den direkten Export der Ergebnisse aus dem Cameo Systems Modeler zurück in die Exceltabelle vor. Dafür wird die in Abschnitt 7.3 beschriebene integrierte Schnittstelle des Cameo Systems Modeler genutzt. Mit dieser werden die ermittelten Werte (z.B. Gesamtmasse) und der Status über die Erfüllung der Anforderungen als (nicht) bestanden in der Exceltabelle ergänzt. Die Exceltabelle kann dann wiederum in weiteren Partialmodellen ausgelesen oder für die Dokumentation der ausgelegten Variante eingebunden werden. Die zweite Möglichkeit ist die Überprüfung der Anforderungen in einer externen Modellierungsumgebung. Dabei werden z.B. in Matlab/Simulink die Anforderungen überprüft und die Werte berechnet. Anschließend können die Ergebnisse entweder als ein Datenarray zurück an den Cameo Systems Modeler übergeben oder direkt in die XML-Datei geschrieben werden. Die dritte Möglichkeit beschreibt den Export der Anforderungswerte aus dem Cameo Systems Modeler durch eine Opaque Aktion. In dieser Aktion werden in einem Python-Code die Anforderungen auf ihre Erfüllung hin überprüft und anschließend als neues Element in die Baumstruktur der XML-Datei eingefügt. Dabei wird nur der berechnete, zu prüfende Eigenschaftswert automatisch aus den Instanzen ausgelesen. Die ID der Anforderung und die Grenzwerte müssen manuell für eine erneute Prüfung hinterlegt werden. In dieser Arbeit wird die erste Variante für den Export der Anforderungen in die XML-Datei verwendet und die Ergebnisse über den internen Export zurück in die Exceltabelle geschrieben.

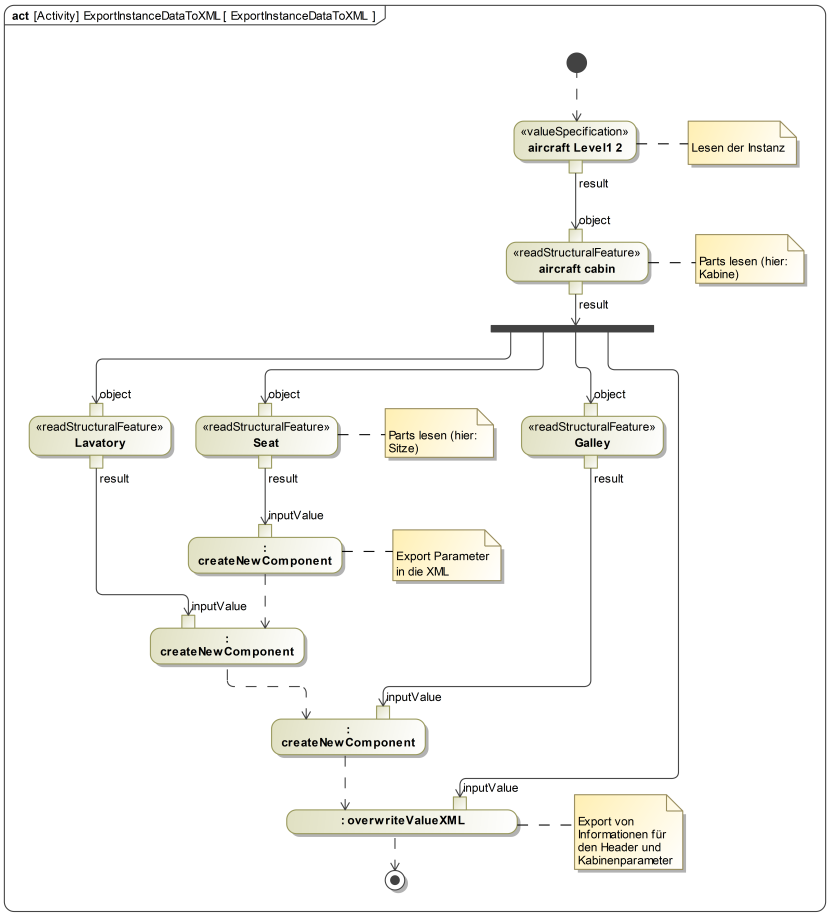



Abbildung 7.27: Aktivitätsdiagramm zum Auslesen der Parameter aus den Instanzen und für die Erzeugung neuer Einträge in der XML-Datei am Beispiel der Flugzeugkabine.

## 7.5 Ansatz für eine automatisierte, externe Ansteuerung der Partialmodelle

Der gesamte Prozess für die Auslegung einer Variante ist automatisierbar. Sind am Auslegungsprozess externe Partner beteiligt, erfolgt die Ansteuerung der Partialmodelle jeweils manuell bei den Partnern. Die Kopplung wiederum kann automatisch erfolgen, indem die XML-Adapterdatei auf einem, für alle am Prozess beteiligten Partner, zugänglichen Server hinterlegt wird. Die Partialmodelle werden dadurch als Black-Box betrachtet, sodass das Expertenwissen beim jeweiligen Partner verbleibt.

Die Partner müssen dafür die Schnittstellen eindeutig definieren und die interne Richtigkeit der Modelle gewährleisten.
Demgegenüber kann der gesamte Auslegungsprozess automatisiert werden, wenn alle Modelle geteilt oder auf demselben Server hinterlegt werden dürfen. In diesem Abschnitt wird dafür ein Ansatz vorgestellt. Die Kopplung und Ausführung der miteinander verknüpften heterogenen Modelle innerhalb eines Partialmodells erfolgt bereits automatisiert. Das SysML-Modell ruft durch die integrierte Toolanbindung über das Aktivitätsdiagramm Simulationsumgebungen wie Matlab auf und tauscht die Parameter zwischen den beiden Entwicklungsumgebungen hin und her. Für einen gesamten, automatisierten Prozess müssen daher die folgenden drei manuell ausgeführten Schritte ersetzt werden:

1. Auswahl und Öffnen der Partialmodelle,
2. Ausführung der SysML-Aktivitätsdiagramme,
3. Erzeugen einer leeren Start-XML-Datei und Umbenennung der XML-Datei am Ende des Durchlaufs.

Ein Lösungsansatz für die Automatisierung ist eine Python-Schnittstelle in Kombination mit einer grafischen Nutzeroberfläche. Der erste manuelle Schritt umfasst die Auswahl und das Öffnen der Partialmodelle. Mit Hilfe einer externen grafischen Nutzeroberfläche (z.B. generiert mit Python) kann ein Interface geschaffen werden, welches automatisiert nacheinander die gewünschten Partialmodelle einer Variante sucht und die Entwicklungsumgebungen öffnet. Dafür müssen alle Modelle an einem fest definierten Speicherort abgelegt sein. Über die GUI wählt der Nutzer aus einem Menü alle Komponenten oder Subsysteme aus, die für die auszulegende Variante verwendet werden sollen. Dabei ist für jede Komponente oder jedes Subsystem der Speicherort und der Dateiname des anzusteuernden Modells fest hinterlegt. Damit können nacheinander die Partialmodelle gesucht und geöffnet werden.

Der zweite manuelle Schritt beinhaltet die Ausführung der Aktivitätsdiagramme im Cameo Systems Modeler. Für die Auslegung eines Subsystems müssen im SysML-Modell drei Aktivitätsdiagramme nacheinander ausgeführt werden. Zuerst das Diagramm für den Import von Parametern aus der XML-Datei, anschließend das Diagramm für die Architekturinitialisierung und Subsystemauslegung und zuletzt das Diagramm für den Export der Parameter in die XML-Datei (siehe Abschnitt 7.4.2). Damit diese Aktivitätsdiagramme nicht manuell nacheinander ausgeführt werden müssen, können diese mit Hilfe einer modellbasierten Ausführungskonfiguration angesteuert werden. Die Simulationskonfiguration (SimulationConfig) ist ein vom Cameo Systems Modeler bereitgestelltes Element, mit dem vordefinierte Optionen ausgeführt werden können. Das Element enthält verschiedene Stereotypen, die individuell angepasst und verändert werden können. Für diese Arbeit sind besonders die beiden Stereotypen executionTarget und resultLocation (siehe (2) in Abbildung 7.28) relevant. Das executionTarget beschreibt ein Element, von dem die Simulation ausgehen soll [No 22a]. In dem Beispiel in Abbildung 7.28 ist der Kontextblock ausgewählt worden, unter dem sich das Aktivitätsdiagramm und weitere Elemente für den Import von Daten aus der XML-Datei befinden. In der resultLocation wird ein Kontextobjekt (z.B. Instanz) nach der Simulation gespeichert [No 22a]. In dem gezeigten Beispiel ist als

Speicherort die Instanz hinterlegt, die im Rahmen des Imports erzeugt wird und die importierten Parameter aus der XML-Datei enthält.

Die Simulationskonfiguration kann entweder über das Kontextmenü oder über die Symbolleiste der Simulationssteuerung (siehe (1) in Abbildung 7.28) ausgeführt werden. Zusätzlich kann die Ausführung des Elements von außerhalb erfolgen. Über eine externe Nutzeroberfläche oder über die Teamworks Cloud kann auf die Simulationskonfiguration zugegriffen werden. Mit einer grafischen Nutzeroberfläche außerhalb der Entwicklungsumgebung des Cameo Systems Modeler kann dann eine externe Ansteuerung der einzelnen SysML-Modelle stattfinden. Für die automatisierte Variantenauslegung eines Systems können dadurch nacheinander die relevanten Partialmodelle angesteuert und ausgeführt werden. Aufgrund der internen, automatischen Kopplung der SysML-Modelle zu externen Entwicklungsumgebungen (z.B. Matlab) wird der Datenaustausch zwischen diesen direkt mitgestartet, sobald die Konfigurations-Ansteuerung über die GUI ausgeführt wird. Sobald der Prozessablauf eines Partialmodells vollständig durchlaufen ist, meldet CMS dies an die Nutzeroberfläche zurück und die Ausführung des nächsten Partialmodells startet. Zusammenfassend ermöglicht die Ausführung der Simulationskonfiguration über einen externen Zugriff die Umsetzung eines sequenziellen Prozess und die Reihenschaltung von Partialmodellen für die holistische Auslegung einer Systemvariante. Je nach Anforderung an das System werden dann nur die entsprechend relevanten Partialmodelle der Subsysteme angesteuert.

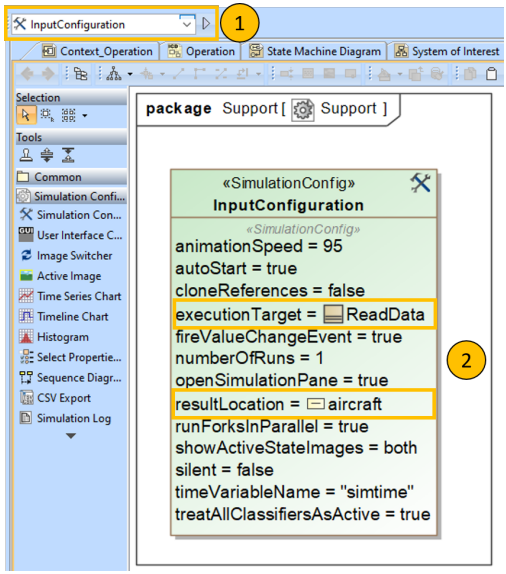



Abbildung 7.28: Element Simulationskonfiguration des Cameo Systems Modeler für die automatisierte Ausführung und Speicherung von Diagrammen oder Elementen.

Der dritte manuelle Schritt umfasst die Erstellung einer leeren XML-Datei zum Start des Prozesses und die Umbenennung der XML-Datei nach erfolgter Auslegung. In

dem Programmcode der Python-Schnittstelle wird daher eine Basisdefinition einer XML-Datei hinterlegt, die jedes Mal zu Beginn der Variantenbildung erzeugt wird. Diese erhält den Namen XML_Adapter.xml. Nachdem alle Partialmodelle angesteuert und ausgeführt wurden, wird die mittlerweile beschriebene XML-Datei automatisch durch die GUI-Schnittstelle umbenannt und an einem fest definierten Speicherort abgelegt. Die Wahl der Bezeichnung kann durch einen Nutzer in der GUI zu Beginn der Auslegung festgelegt werden. Vertiefende Einblicke in den Aufbau der GUI, die Speicherung der XML-Datei und die externe Ansteuerung der SysML-Modelle sind der Arbeit von Hoyos [Hoy24] zu entnehmen.

## 7.6 Kapitelfazit

In den Abschnitten dieses Kapitel wurde der Lösungsansatz von föderierten Partialmodellen für die Variantenmodellierung und das Customizing im Luftfahrtkontext umgesetzt. Dabei kann das Gesamtsystem Flugzeug in mehrere Subsysteme unterteilt werden. Die Klassifizierung nach den ATA-Kapiteln ermöglicht einen Anhaltspunkt für die Aufteilung in Partialmodelle. Innerhalb der Partialmodelle werden für die Auslegung und Analyse sowohl analytische als auch deskriptive Modelle verwendet. Daher wurde in diesem Kapitel der interne Modellaufbau dieser heterogenen Modelle im Detail erläutert. Der Fokus lag dabei auf den jeweilig verwendeten Modellelementen und Modellstrukturen, um den Lösungsansatz aus Abschnitt 6.2 zu realisieren.

Die Kopplung der Partialmodelle zu einem Gesamtsystem wird durch den Austausch von Parametern über die Systemgrenzen eines Partialmodells hinaus mittels eines XML-Adapters realisiert. Die Kopplung der heterogenen Modelle innerhalb eines Partialmodells findet über die integrierten Schnittstellentechnologien der Modellierungstools statt. Ist diese nicht gegeben, kann hier ebenfalls das Adaptermodell zum Einsatz kommen. Allgemein wird nur ein einziges Adaptermodell verwendet, welches je nach Produktvariante neu mit Daten gefüllt wird. Mit dieser einzigen Datenquelle wird die Konsistenz der Daten gesichert, indem jedes Partialmodell auf dieses referenziert und die Anforderung an einen Parameteraustausch erfüllt.

Schließlich wurde im Detail vorgestellt, wie die Partialmodelle und deren Schnittstellen entsprechend der Produktvariantenauslegung in einem übergeordneten Prozess mittels einer GUI automatisch angesteuert und ausgeführt werden können. Die Ansteuerung und Ausführung der heterogenen Modelle innerhalb eines Partialmodells erfolgt bereits automatisch. Damit bietet die Methode Ansätze für eine ganzheitliche Automatisierung, um bei einer Vielzahl auszuführender Partialmodelle die Prozesszeiten manueller Schritte auf ein Minimum zu reduzieren.

# 8. Anwendung gekoppelter Partialmodelle zur Variantenbildung und Customizing im industriellen Kontext

Im vorangegangenen Kapitel 7 wurden die Modellelemente und internen Strukturen der heterogenen Modelle für die Umsetzung des Lösungsansatzes für die Analyse von Flugzeugvarianten und Kabinenkonfigurationen vorgestellt. Ziel der Methode ist die Unterstützung bei der kollaborativen Modellierung von Produktvarianten unter Berücksichtigung einer einfachen Modellanpassung. Die Umsetzung und Anwendbarkeit der Methode wird in diesem Kapitel im industriellen Kontext mit mehreren Partnern anhand der Auslegung eines Flugzeugs untersucht und in zwei Fallstudien demonstriert. Damit stellt das Kapitel die Descriptive Study II der DRM zur Evaluation der entwickelten Unterstützung dar.
Die vorliegenden Studien zielen darauf ab, gemäß der DRM von Blessing und Chakrabarti, in einer initialen DS II die Anwendbarkeit der Unterstützung im Sinne einer Anwendungsevaluation anhand der Schlüsselfaktoren zu bewerten [BC09] und die gestellte Forschungsfrage zu beantworten. Dabei werden mit der Methode in der ersten Fallstudie in Abschnitt 8.2 zwei Flugzeugvarianten erzeugt. Diese beiden Varianten unterscheiden sich in der Art ihrer Energieerzeugung für die elektrische Versorgung der Flugzeugkabinensysteme. Bei der ersten Variante wird als Referenzkonfiguration das Hilfstriebwerk verwendet, während bei der zweiten Variante dieses durch ein Wasserstoff-Brennstoffzellensystem ersetzt wird. Die Kopplung verschiedener Partialmodelle durch das Schnittstellenformat XML wird hier ebenfalls diskutiert.
Schließlich wird in Abschnitt 8.3 mit der entwickelten Methode in der zweiten Fallstudie das Customizing innerhalb eines Partialmodells (Kabine) untersucht. Dabei wird eine der Flugzeugvarianten ausgewählt und ein Subsystem (Gepäckablage) der Kabine im Rahmen des Customizings verändert sowie die neue Architekturauslegung entsprechender Kundenkriterien bewertet.

## 8.1 Versuchsaufbau und Industriekontext

Die nachfolgenden Abschnitte geben einen methodischen Überblick zur Evaluation der angewandten Methode, stellen den Industriekontext vor und geben einen Einblick in die getroffenen Annahmen und Anforderungen des Versuchsaufbaus. Hierzu geht

Abschnitt 8.1.1 auf die Art der Evaluationsmethodik ein und Abschnitt 8.1.2 stellt den Industriekontext vor. Abschnitt 8.1.3 stellt die im Rahmen dieser Arbeit getroffenen Annahmen und Anforderungen dar.

### 8.1.1 Evaluationsmethodik

Grundsätzlich gibt es für die Durchführung der Evaluation diverse Methoden: Methoden zur Datenerhebung in Echtzeit wie beobachtende Fallstudien oder Computersimulationen und retrospektive Methoden wie Experteninterviews und Fragebögen [BC09]. Zudem gibt es weitere Methoden wie statistische Experimente oder Analysen [JP14]. Bei den evaluierenden Methoden wie Analyse oder statistische Experimente handelt es sich um komplexe Vorgehensmodelle, die genaue Daten anderer bereits existierender Methoden für einen Vergleich benötigen. In den meisten Publikationen im Bereich des (Design-)Vorentwurfs von Flugzeugen sind allerdings kaum nutzbare Daten (z.B. Entwicklungszeiten, Prozesszeiten) angegeben, um eine Validierung der hier entwickelten Methode vorzunehmen. Zudem wurden in Kapitel 6 die Herausforderungen vorgestellt, die mit der in dieser Arbeit entwickelten Methode adressiert werden sollen und die sich überwiegend im industriellen Kontext wiederfinden. Daher werden durch die mittels Literaturrecherche erarbeiteten Herausforderungen bei der Entwicklung neuer Flugzeugsystemkonzepte in der Industrie und den fehlenden Referenzdaten anderer Methoden, in dieser Arbeit Fallstudien im industriellen Kontext als geeignete Evaluationsmethodik angesehen und eingesetzt.

Blessing und Chakrabarti verstehen die Analyse realer Entwurfsprozesse in einer praktischen Umgebung (z.B. Fallstudien) als eine Form der Feldforschung [BC09]. Der Hauptvorteil liegt darin, dass die Ergebnisse auf der Realität basieren und eine praxisnahe Ausrichtung haben [BC09]. Im Gegensatz zu Laborbedingungen ermöglichen Fallstudien mit Praxisbezug die Untersuchung der Problemstellung in der Tiefe und die Konfrontation der entwickelten Artefakte mit nicht zu stark vereinfachten Annahmen [JP14]. Artefakte werden in dieser Arbeit nach Johannesson und Perjons definiert als durch Menschen hergestellte Objekte (z.B. Modelle, Richtlinien oder Prototypen), die zur Lösung eines Problems verwendet werden [JP14].

Im Rahmen der Evaluation dieser Arbeit werden zwei Fallstudien im industriellen Kontext angewandt mit dem Ziel, die Interaktion der Methode mit ihrem realen Kontext zu untersuchen. Der Vorteil liegt darin, dass von dem realen Projekt die Anforderungen berücksichtigt werden und die Lösung reale Herausforderungen fokussiert. Die Erkenntnisse, die sich aus der Anwendung der Fallstudien ergeben, sind daher von besonderer Relevanz und können für die Erreichung des Forschungsziel herangezogen werden [Wie14]. Darüber hinaus wird in der ersten Fallstudie als Referenz eine heutige Standard-Flugzeugkonfiguration ausgelegt, um die Methode zu validieren und die erzielten Ergebnisse mit den Daten anderer Vorentwurfsmethoden und dem realen Produkt zu vergleichen. Da diese Arbeit sich mit der Auslegung von Systemen beschäftigt, die unter anderem noch nicht existieren und mit der entworfenen Methode bestehende Prozesse modifiziert werden sollen, um bessere Ergebnisse zu erzielen, kann die Art der Forschung nach Dresch et al. als „Design Science“ beschrieben werden [DLA15]. Damit kann die entwickelte Methode mit einer Fallstudie evaluiert werden. Die jeweilige Strukturierung der Durchführung der beiden Fallstudien besteht daher in Anlehnung an [DLA15] aus folgenden Schritten:

1. Definition des Fallbeispiels: In diesem Schritt wird das Fallbeispiel der Studie definiert. Dies umfasst die Spezifikation der Problemstellung und die gewünschte Zielstellung, die mit der Methode erreicht werden soll.

2. Planung und Vorgehen der Fallstudie: Anschließend werden die Artefakte der Methode für die Zielerreichung zusammengetragen und das Vorgehen (Prozess) für die Verknüpfung dieser erarbeitet. Da beide Fallstudien unterschiedliche Ziele adressieren und Herausforderungen angeben, werden in den Beispielen unterschiedliche Fokusse gesetzt und Artefakte angewendet.

3. Anwendung der Methode und deren Artefakte: Die Methode wird im Rahmen der Anwendung erprobt und durch die Autorin sowie die Projektpartner umgesetzt. Dafür wird ein detaillierter Einblick in die Artefakte gegeben.

4. Analyse der Ergebnisse: In diesem Schritt werden die erzielten Ergebnisse dahingehend analysiert, inwieweit die Anforderungen an die Methode erfüllt und Lösungen für die Herausforderungen geschaffen werden konnten.

5. Dokumentation: Zum Abschluss werden die Ergebnisse der Anwendung dokumentiert, sodass die Ergebnisse nachvollziehbar und reproduzierbar sind.

Die nächsten beiden Abschnitte stellen zum einen den Industriekontext und das Industrieprojekt für die Fallstudien vor. Zum anderen werden die Annahmen und Anforderungen erläutert, die dem Entwurf der Flugzeugkonfiguration zugrunde gelegt werden. Eine erste Analyse der Ergebnisse in Bezug auf das Forschungsziel der vorliegenden Arbeit findet jeweils am Ende des Abschnitts des Fallbeispiels statt und wird in Kapitel 9 vertieft.

### 8.1.2 Beschreibung des Industriekontexts

Die in dieser Arbeit entwickelte Methode wird in einem industriellen Kooperationsprojekt als realistisches Anwendungsszenario erprobt. Grundlage des Industriekontexts bildet das durch die Hamburger Förderrichtlinie GATE (Green Aviation Technologies) geförderte Projekt MIWa (**M**BSE-basierte **I**ntegration & Variantenbildung von **Wa**sserstoffkryodrucktanksystemen zukünftiger Flugzeugkonfigurationen). Ziel des Projekts ist die Durchführung ganzheitlicher Analysen für verschiedene Flugzeugvarianten im frühen Stadium der Flugzeugentwicklung. Der Fokus liegt dabei auf der Untersuchung alternativer Antriebssysteme und der Integration von Wasserstoffsystemen in Flugzeugstrukturen. Mit Hilfe des modellbasierten und multidisziplinären Systementwicklungsansatzes können unterschiedliche Gesamtsystemvarianten erzeugt und verglichen werden. Damit im Projekt Varianten von Wasserstoffflugzeugen erzeugt und ganzheitlich bewertet werden können, stellt die in dieser Arbeit entwickelte Methode einen Ansatz für den Einsatz des Cameo Systems Modeler als Entwicklungsumgebung für die Systemarchitekturmodellierung sowie für die Kopplung der Schnittstellen zu anderen Modellierungsumgebungen im industriellen Kontext bereit.
Neben der Projektbeteiligung des DLRs sind die HAW Hamburg als weitere Forschungseinrichtung und die KMU Centerline Design als industrieller Kooperationspartner beteiligt. In den folgenden Studien liegt der Fokus auf der Anwendung der Methode bei dem in dieser Arbeit neu aufgebauten Kabinenmodell. Zudem wird die

Kopplung der Partialmodelle im industriellen Kontext untersucht und die partnerübergreifende Integration fremder Partialmodelle fokussiert. Dafür liefern die beiden Projektpartner zwei Modelle, in denen jeweils ihr Expertenwissen hinterlegt ist. Für die Validierung der Modelle sind die Projektpartner verantwortlich. Dies betrifft das Wasserstofftanksystemmodell von Centerline Design [FAF+23] und das Brennstoffzellensystemmodell von Meier [Mei23] von der HAW Hamburg. Lediglich der Aufbau der externen Partialmodelle (Wasserstofftank, Brennstoffzelle) erfolgt nach den gleichen internen Strukturen, wie in dieser Arbeit gezeigt. Nähere Einblicke in die Modelle sind [FAF+23] zu entnehmen.

Einen Überblick über die beiden Fallstudien gibt Abbildung 8.1. In der ersten Fallstudie werden mit der in dieser Arbeit entworfenen Methode zwei Flugzeugvarianten ausgelegt. Dabei wird nur eine Iteration (Version 1) untersucht. Die Varianten unterscheiden sich in der Art ihrer Energieerzeugung für den Betrieb der Kabine und die Versorgung einiger Hauptdienste im Notfall. Bei der ersten Variante (1a) wird ein Flugzeug mit einer Hilfsturbine (Auxiliary Power Unit, APU) im Heck ausgestattet. Bei der zweiten Variante (1b) wird die Hilfsturbine durch ein Brennstoffzellensystem in Kombination mit einem Wasserstofftanksystem ausgetauscht. Bei beiden Varianten wird die gleiche Kabineninnenausstattung (nur Economy, reguläres Gepäckablagefach) verwendet. In der zweiten Fallstudie wird das Customizing untersucht. Dabei wird die Kabine der zweiten Variante (Brennstoffzellensystem) entsprechend beispielhafter Kundenwünsche verändert. Bei der ersten Customizingvariante (1b-1) wird die reguläre Gepäckablage durch eine größere Variante ausgetauscht. Bei der zweiten Customizingsvariante (1b-2) wird anstelle einer Einklassenbestuhlung eine Zweiklassenbestuhlung (Businessklasse und Economyklasse) untersucht und eine neuartige, modulare Gepäckablage verwendet.

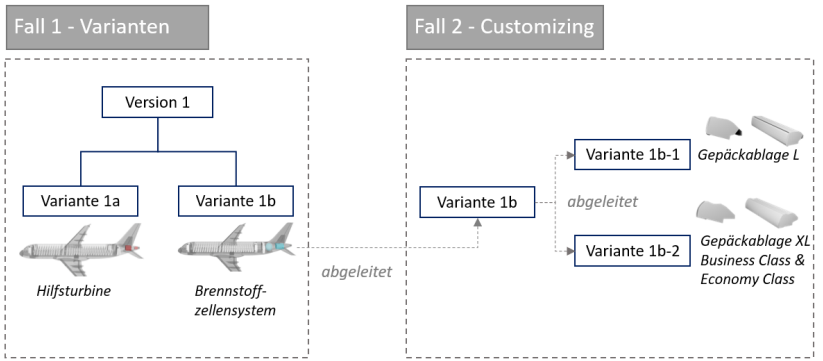



Abbildung 8.1: Übersicht der beiden untersuchten Fallstudien für die Generierung von Flugzeugvarianten (Fall 1) und für die Anwendung des Customizings bei einer Variante (Fall 2).

Bei der Durchführung der beiden Studien werden unterschiedliche Tools und Programmiersprachen eingesetzt. Tabelle 8.1 zeigt eine Übersicht über die angewandten Tools und Programmiersprachen für die Versuchsreihe. Die Durchführung erfolgte mit einem Dell-Laptop mit einem Arbeitsspeicher von 32GB, mit Windows 10, einem 12th

Gen Intel(R) Core(TM) i7-1270P Prozessor und mit den Grafikkarten Intel(R) Iris(r) Xe sowie NVIDIA GeForce MX550.

Im Projekt werden für alle Partialmodelle der Cameo Systems Modeler für den Aufbau der SysML-Modelle verwendet. Beim Partialmodell der Brennstoffzelle und der Kabine wird Matlab bzw. Matlab/Simulink für die Analyse und Auslegung der Systemkomponenten angewendet. Die 3D-Visualisierung der Flugzeugkonfigurationen inklusive der Kabine erfolgt mit der Grafiksoftware Blender. Als Adapter zwischen den Modellen wird eine XML-Datei verwendet. Anforderungen hingegen werden in einer Exceltabelle gesammelt. Die automatisierte Ansteuerung der Modelle sowie die erweiterte Programmierung innerhalb des Cameo Systems Modeler und die Dimensionierung des Wasserstofftanks erfolgen mit Python.

Tabelle 8.1: Übersicht über die angewandten Tools und Programmiersprachen.

| Tool/ Sprache | Version |
|---|---|
| Matlab | R2022b |
| Cameo Systems Modeler | 2022x |
| Blender | 3.3.1 |
| XML | 1.0 |
| Excel | 2019 |
| Python | 3.8 |

### 8.1.3 Annahmen und Anforderungen

Die Studien zur Variantenbildung und für das Customizing werden anhand eines Mittelstreckenflugzeugs durchgeführt. Ausgangslage bildet ein Mittelstreckenflugzeug vom Typ Airbus A320neo. Für die hier untersuchten Flugzeugkonfigurationen wird die Rumpfstruktur des Flugzeugs nicht verändert. Stattdessen wird für alle ausgelegten Varianten dieselbe Rumpfstruktur verwendet. Grundlage für die Rumpfstruktur bildet die Referenzkonfiguration aus einem DLR-internen Projekt mit einem generischen Mittelstreckenflugzeug mit 180 Passagieren mit einer Einklassenbestuhlung (single-class, Economy) [FAF+23]. Tabelle 8.2 zeigt die Auslegungsparameter für das Flugzeugmodell.

Zudem wird bei der Betrachtung der Kabine der Fokus auf eine Subsystembaugruppe (Gepäckablage) gelegt, welche wiederum im Detail ausgearbeitet wird. Die anderen Subsysteme der Kabine werden im Ansatz dargestellt. Die Bewertung der in dieser Arbeit generierten Kabinenvarianten erfolgt auf Basis einfacher Kriterien. Dazu gehören die Gesamtmasse der Kabine, die benötigte Leistung zur Versorgung der Kabine, der Passagierkomfort und der Grad der Kompaktbauweise der Subsystembaugruppe. Für die Massen der Kabinenmonumente wurden basierend auf den Datenblättern von [Win23] überschlägige Annahmen getroffen. Nach dem Stand der Forschung ergibt sich für die Betrachtung des Passagiers ein Gesamtgewicht von 95 kg inklusive 14 kg Gepäck [FAF+23]. Für die Galley wird nach der Studie von Doering und Thielecke

[DT18] eine Leistung von 22 kW angenommen. Die Versorgung des Passagiers mit Licht und Strom wird nach [Sch15a, S. 33] konservativ mit einem Wert von 0,1 kW abgeschätzt [FAF+23]. Tabelle 8.3 zeigt die konservativ getroffenen Annahmen der Massen und Leistungsparameter für die Kabinenmonumente.

Tabelle 8.2: Referenz-Auslegungsparameter für ein Mittelstreckenpassagierflugzeug von [FAF+23].

| Bezeichnung | Wert |
|---|---|
| Max. Anzahl Passagiere | 180 |
| Referenzmission | 2500 nm |
| Max. Betriebshöhe | 40.000 ft |
| Kabinenlänge (innen) | 26,90 m |
| Kabinendurchmesser | 3,69 m |

Für die Bewertung des Passagierkomforts werden die Erreichbarkeiten der Bedienelemente der Passagierservicefunktionen für jeden Sitzplatz ermittelt. Anschließend werden alle ermittelten Werte addiert und ein Mittelwert gebildet. Je kürzer die Entfernung zwischen den Bedienelementen und der Sitzposition ist, desto komfortabler ist der Sitzplatz. Zuletzt wird der Grad der Kompaktbauweise ermittelt. Dafür müssen alle Subsystemkomponenten in einer Baugruppe angeordnet werden können und dürfen ein fest definiertes geometrisches Abmaß nicht überschreiten.

Tabelle 8.3: Massen- und Leistungsparameter für die Kabinenkomponenten von [FAF+23].

| Bezeichnung | Gewicht [kg] | Leistung [kW] |
|---|---|---|
| Passagier inkl. Gepäck | 95 | 0,1 |
| Lavatory | 50 | 1,0 |
| Galley | 150 | 22,0 |
| 3er Sitzreihe | 30 | - |

Für die Flugzeugvarianten wird eine allgemeingültige Anforderungsliste verwendet. Einen Einblick in die Anforderungsliste gibt Anhang A.6.1. Die Liste enthält Vorgaben an das Design, Sicherheitsregularien und Vorschriften aus der Bauvorschrift für Großflugzeuge CS-25 von der EASA [Eur21]. Designanforderungen umfassen einerseits Angaben zu Bemaßungen einzelner Kabinenkomponenten und anderseits Vorgaben zu allgemeinen geometrischen Abmaßen der Kabine. Zusätzlich werden Massenangaben und Leistungsparameter berücksichtigt. Die Sicherheitsregularien und Bauvorschriften beinhalten Anforderungen an die geometrische Platzierung der Kabinenkomponenten. Dazu gehören Freibereiche an den vorderen und hinteren Notausgangstüren sowie größere Sitzabstände bei den mittleren Notausgängen über

den Flügeln. Zusätzlich werden Anforderungen an die Erreichbarkeit von Sicherheitsequipment (z.B. Sauerstoffmasken) betrachtet. Weitere Anforderungen wie die modulare Anordnung von Subsystemkomponenten für eine kompakte Bauweise, unterschiedliche Sitzklassen (Business- oder Economyklasse) oder variierende Sitzabstände werden ebenfalls berücksichtigt.

## 8.2 Fallstudie I: Flugzeugvariantenbildung

In der ersten Fallstudie werden mit der in dieser Arbeit entwickelten Methode zwei Flugzeugvarianten modelliert und ausgelegt. Dafür werden verschiedene Partialmodelle miteinander gekoppelt und Parameter über die definierten Schnittstellen ausgetauscht. Die Partialmodelle für das Wasserstofftanksystem und das Brennstoffzellensystem werden von den Projektpartnern bereitgestellt und im folgenden Verlauf als Black-Box betrachtet.

### 8.2.1 Beschreibung des Anwendungsfalls und des Auslegungsprozesses

Die Durchführung der Fallstudie wird nach den Schritten aus Abschnitt 8.1.1 strukturiert. Im Folgenden werden die drei Schritte Definition, Vorgehen und Anwendung der Methode und dessen Artefakte vorgestellt. Anschließend werden die Ergebnisse dokumentiert und analysiert.

#### Definition

In diesem Anwendungsfall werden mit Hilfe der Partialmodelle zwei Flugzeugvarianten generiert. Beide Varianten verwenden die gleiche Rumpfstruktur. Zudem wird bei beiden die gleiche Standardkabinenauslegung angenommen. Diese zeichnet sich durch eine Einklassenbestuhlung aus. Darüber hinaus wird eine Standard Gepäckablagenarchitektur verwendet. Unterschieden werden die Varianten in der Art der Energieerzeugung für die Bereitstellung von Strom für die Versorgung und den Betrieb der Kabinensysteme. Ziel der Studie ist es, die Kopplung unterschiedlicher Partialmodelle partnerübergreifend zu demonstrieren und die Inputs/Outputs zu testen. Insgesamt symbolisiert die Studie die Sicht des OEMs wie Airbus auf die Produktauslegung und Untersuchung von Variantenkonzepten für die Erzeugung von Produktfamilien.

#### Vorgehen

Die erste Variante stellt eine Referenzkonfiguration dar, bei der sich im Heck des Flugzeugs eine Hilfsturbine befindet. Der Auslegungsprozess für Flugzeugvariante 1a mit Hilfsturbine ist in Abbildung 8.2 dargestellt. Der Prozess startet mit einen Parametersatz, in dem die Anfangswerte wie Reichweite und das Missionsprofil hinterlegt sind. Bei der Hilfsturbine handelt es sich um eine bereits dimensionierte und zertifizierte

Systemkomponente. Deren Daten werden von externen Partnern für die Auslegung der Kabine zur Verfügung gestellt und liegen als Parametersatz vor. Ausgehend von den Parametern wird die Kabine dimensioniert. Dabei importiert, verarbeitet und exportiert das Partialmodell der Kabine die Parameter aus der XML-Datei nach dem EVA-Prinzip. Gleichzeitig werden die Anforderungen überprüft und die Ergebnisse in die Anforderungstabelle in Excel übertragen. Im letzten Schritt wird die berechnete Flugzeugkonfiguration in einem 3D Modell visualisiert.

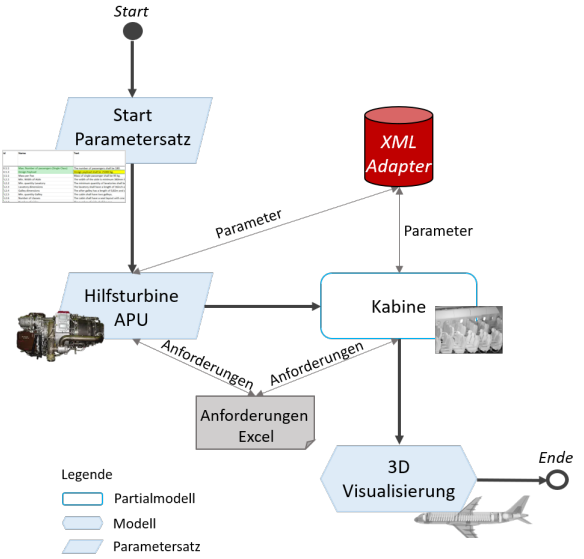



Abbildung 8.2: Prozess für die Auslegung der Variante 1a mit Hilfsturbine[13].

Die zweite Variante betrachtet eine Flugzeugkonfiguration, bei der anstelle der Hilfsturbine ein Brennstoffzellensystem zusammen mit einem Wasserstofftanksystem verwendet wird. Im Gegensatz zu Variante 1a wird eine bereits zertifizierte und dimensionierte Systemkomponente (APU) durch zwei neue Systeme ausgetauscht. Damit die neue Flugzeugkonfiguration ermittelt werden kann, werden zwei Partialmodelle angeschlossen, mit denen eine Berechnung und Dimensionierung der neu hinzukommenden System erfolgen kann. Abbildung 8.3 zeigt den Prozess für die Auslegung und Architekturgenerierung für die Variante 1b. Der Prozess startet ebenfalls mit einen Startparametersatz, in dem die Ausgangsinformationen, wie Reichweite, zusammengetragen sind. Anschließend werden im Gegensatz zur ersten Variante drei Partialmodelle nacheinander angesteuert und ausgeführt. Für Variante 1b startet der Durchlauf mit dem Brennstoffzellensystem, führt dann mit dem Tankmodell fort und endet mit der Ausführung des Kabinen-Partialmodells. Während der Ausführung der Partialmodelle werden entsprechend dem EVA-Prinzip jeweils zuerst die Parameter aus der XML-Datei ausgelesen und importiert. Anschließend findet die Verarbeitung der

[13]Beispielhafte Darstellung einer APU von https://www.aeroexpo.online/prod/aerosila/product-173981-67266.html.

Daten statt und das entsprechende Subsystem wird modelliert und ausgelegt. Zum Schluss werden die neu generierten Parameter aus dem Partialmodell zurück in die XML-Datei geschrieben. Parallel dazu werden die Anforderungen aus der Exceltabelle in die Partialmodelle importiert und die Ergebnisse aus der Anforderungsüberprüfung in diese wieder zurück geschrieben.

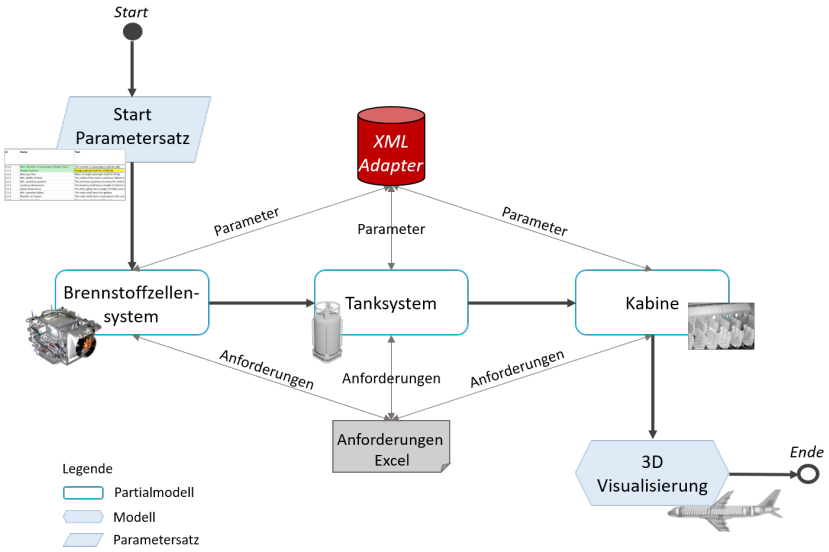



Abbildung 8.3: Prozess für die Auslegung der Variante 1b mit Wasserstofftank-[14]und Brennstoffzellensystem.

Im letzten Schritt wird die generierte Flugzeugvariante mit Blender in einer 3D Darstellung visualisiert. Das Partialmodell der Kabine erzeugt zusätzlich einen 2D Plot der Kabinenkonfiguration. Insgesamt wird für jede Variante eine eigene XML-Datei zur Kopplung und für den Austausch der Parameter verwendet.

#### Anwendung der Methode und dessen Artefakte

Die Artefakte (Modelle und deren Elemente) für das Partialmodell der Kabine wurden bereits zu einem großen Teil in Kapitel 7 vorgestellt. In diesem Abschnitt werden nochmal zwei Einblicke in den Aufbau des Partialmodells der Kabine gegeben, um die Entstehung der Werte im Ergebnisteil zu erklären und nachzuvollziehen. Der erste Einblick umfasst die Simulation im Cameo Systems Modeler für die Erstellung der Systemarchitektur und den Datentransfer zum Matlab-Modell. Das Aktivitätsdiagramm für die Simulation im Cameo Systems Modeler ist in Abbildung 8.4 dargestellt. Das Diagramm befindet sich im Verabeitungsordner des SysML-Modells. Der Import

[14]Bei der beispielhaften Abbildung des Tanks handelt es sich um einen Wasserstofftank der Firma Cryotherm.

der Parameter aus dem XML-Datei ist bereits erfolgt. Die Ausführung des Aktivitätsdiagramm startet mit dem Einlesen der Instanz aircraft, die drei Eigenschaften mit der Bezeichnung LengthHydrogenTank, PowerComponentType und Power enthält (vgl. Abschnitt 7.4.2.1). In einem nächsten parallelen Schritt werden innerhalb der Instanz aircraft nach der Länge des Tanks und der Powerkomponente gefiltert. Die Länge des Wasserstofftank wird für eine erste Berechnung der zur Verfügung stehenden Kabinenlänge und der Sitzreihen genutzt. Mit der Opaque Funktion calculateCabinArea werden die Werte berechnet. Die Formel für die Berechnung der Kabinenlänge ist in Gleichung 8.1 dargestellt und ergibt sich aus der Rumpflänge ($l_{Cabin}$) abzüglich der Cockpitlänge ($l_{Cockpit}$), der Hecklänge ($l_{Tail}$) und der Länge der Wasserstofftanks ($l_{HydrogenTank}$). Die Werte für die Berechnung erhält das Modell aus den modellierten Objektklassen und der Instanzvalue LengthHydrogenTank. Der Wasserstofftank hat in dem Fallbeispiel als einziges eine Auswirkung auf die Kabinenfläche, da für dessen Platzierung zusätzlicher Bauraum im Flugzeug bereitgestellt werden muss. Wenn eine Variante ohne Wasserstofftank ausgelegt wird, wird der Wert auf 0 gesetzt.

$$l_{Cabin} = l_{Fuselage} - l_{Cockpit} - l_{Tail} - l_{HydrogenTank} \tag{8.1}$$

Mit der Kabinenlänge kann im nächsten Schritt in einer Annäherung die Kabinenfläche $a_{Cabin,Seats}$ berechnet werden, die für Passagiersitze zur Verfügung steht. Dafür werden von der Fläche der Kabine, berechnet aus der Kabinenlänge und -breite ($b_{Cabin}$), die Flächen für den Mittelgang ($b_{Aisle}$), für die Waschräume ($a_{Lavatory}$), die Küchen ($a_{Galley}$) und die Exits ($a_{Exits}$) abgezogen. Die Berechnung ist in Gleichung 8.2 dargestellt.

$$a_{Cabin,Seats} = \left(\frac{l_{Cabin}}{b_{Cabin}}\right) - l_{Cabin} * b_{Aisle} - \sum a_{Lavatory} - \sum a_{Galley} - \sum a_{Exits} \tag{8.2}$$

Im Anschluss kann mit der Kabinenfläche für die Sitze die Anzahl der Sitzreihen $n_{Rows}$ ermittelt werden, indem die Fläche durch den Sitzabstand ($sp$) multipliziert mit der Sitzbreite ($b_{Seat}$) und der Anzahl der Sitze pro Reihe ($n_{Seats,Row}$) dividiert wird. Die Berechnung der Anzahl an Sitzreihen ist in Gleichung 8.3 aufgeführt. Das Ergebnis wird abgerundet.

$$n_{Rows} = round\left(\frac{a_{Cabin,Seats}}{sp * b_{Seat} * n_{Seats,Row}}\right) \tag{8.3}$$

Im Anschluss werden die Werte für die ermittelte Kabinenlänge (CabinLengthInside) und Anzahl an Sitzreihen (NumberofRows) zusammen mit weiteren Parametern an das Matlab-Modell übergeben. Der Aufruf für die Ausführung lautet run main.m. Nach Beendigung der Berechnungen in Matlab, werden einige Parameter an das SysML-Modell zurück übergeben und die Kabinenobjekte instanziiert.

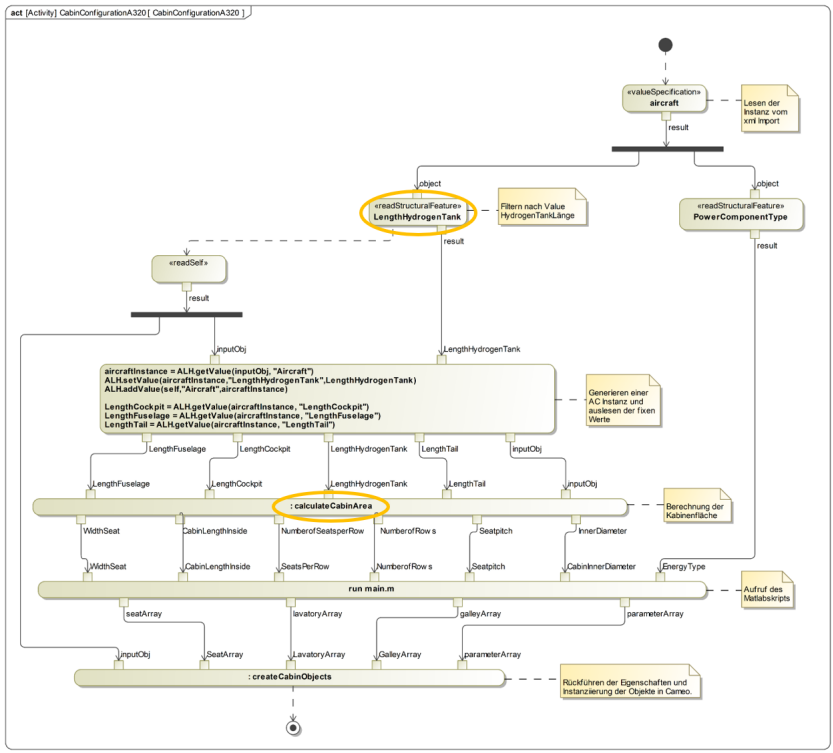



Abbildung 8.4: Aktivitätsdiagramm für die Simulation zur Auslegung der Kabinenarchitektur und Ansteuerung des Matlab-Modells, generiert mit dem Cameo Systems Modeler.

Der zweite Einblick beleuchtet die Weiterverarbeitung von Parametern im Matlab-Modell. In der ersten Fallstudie werden die Startparameter wie Kabinenlänge, Energietyp und Anzahl der Sitzreihen vom SysML-Modell bereitgestellt. Im Gegensatz dazu, werden in der zweiten Fallstudie die Parameter sowohl durch das Matlab-Modell selber und als auch durch die XML-Datei bereitgestellt. Damit das Modell weiß, woher die Daten ausgelesen werden sollen, erfolgt am Anfang der Ausführung eine Abfrage. Abbildung 8.5 zeigt einen Ausschnitt des Hauptskripts (main.m). Die Fallunterscheidung wird mit der Überprüfung der Variablen caseParameter in zwei if-Statements abgefragt. In Zeile 4 wird der Variablen ein Wert zugeschrieben.

Für den Fall der Variantenauslegung erhält die Variable den Wert 1. In den Zeilen 7 bis 19 werden daraufhin die vom Cameo Systems Modeler bereitgestellten Variablen (z.B. NumberofRows) auf der rechten Seite den Matlab-Variablen auf der linken Seite zugeordnet und die Werte überschrieben. Dabei ist zu beachten, dass die Bezeichnung der Variablen identisch zu den Bezeichnungen im Aktivitätsdiagramm in Abbildung 8.4 ist. Zusätzlich wird in Zeile 20 der Name der XML-Datei hinterlegt, um weitere Parameter (z.B. für den 2D Plot) auslesen zu können.

Für den Fall des Customizings, erhält die Variable den Wert 2. In Zeile 23 und 24 können weitere Einstellungen für die Kabinenanpassung, z.B. die Anzahl der Businessklassesitze und die Art der Gepäckablage vorgenommen werden. Zudem wird der Name der XML-Datei hinterlegt, aus dem dann mit der Berechnung in Zeile 26 die weiteren Werte für die Kabinenauslegung ermittelt werden können. Die Auslegung der Kabine erfolgt daraufhin bei beiden Fallstudien identisch.

```
1  %Abfrage Fall I oder Fall II
2  %1=Varianten, MaxPax
3  %2=Customizing 1 Variante
4  caseParameter = 1;

6  if caseParameter ==1
7      params.aircraft.type = "A320";
8      params.cabin.rowsBC = 0;
9      params.cabin.rowsEC = NumberofRows;
10     params.cabin.nPaxBc = 0;
11     params.cabin.nPaxEc = SeatsPerRow*NumberofRows;
12     params.cabin.seatPerRowsEC = SeatsPerRow;
13     params.cabin.seatPerRowsBC = 0;
14     params.cabin.rows = params.cabin.rowsEC + params.cabin.rowsBC;
15     params.cabin.aisleNumber = 1;
16     params.cabin.passenger = SeatsPerRow*NumberofRows;
17     params.seat.pitchEC = Seatpitch*1000*params.mm2i;
18     params.cabin.dia = CabinInnerDiameter;
19     params.cabin.length = CabinLengthInside*1000;
20     xmlAdapterName = "XML_Adapter.xml";
21 end
22 if caseParameter ==2
23    rowsBC = 0;
24    ohscType = "large"; % regular, large, extraLarge
25    xmlAdapterName = "XML_V1.1b.xml";
26    [params,EnergyType] = setLOPA(params,"A320",rowsBC,xmlAdapterName);
27 end
```

Abbildung 8.5: Matlab-Code für die Abfrage der Fallstudie und Zuordnung der Parameter.

Um die Unterschiede in beiden Fallstudien aufzuzeigen, ist in Abbildung 8.6 dargestellt, welche heterogenen Modellen inklusive deren Funktionen und Elemente innerhalb des Partialmodells der Kabine ausgeführt oder verwendet wurden. Die Darstellung zeigt die Anwendung der Modelle für die Auslegung beider Varianten in Fallstudie I. Alle Elemente die verwendet wurden, sind mit einem “x“ markiert. Insgesamt wurden für die Auslegung der Kabine in den beiden Varianten jeweils 30 Funktionen ausgeführt, 46 Elemente angesteuert und 8 Diagramme verwendet.
Beim Variantenentwurf aus Gesamtsystemebene im Rahmen des konzeptionellen Flugzeugvorentwurfs ist zunächst nur eine vorläufige Dimensionierung und Massenabschätzung für die Nutzlast sowie eine Überprüfung der Missionsanforderungen (z.B. Anzahl Passagiere) notwendig. In dieser Phase wird die Kabine in Form eines vorläufigen Sitzlayouts berücksichtigt und nur die Anordnung der Sitze, Küchen und Waschräume berücksichtigt [Sch15b]. Dadurch kann die Breite des Entwurfsraums abgedeckt und geeignete Architekturen für die Erfüllung einer Transportaufgabe bereits in einer frühen Phase identifiziert werden [Wal24, FNG12]. Ein detaillierter Ent-

wurf erfolgt im Anschluss an die Auswahl einer vielversprechenden Gesamtsystemvariante. Für die Darstellung der Flugzeugvariante als 3D Modell werden daher in der Fallstudie I nur die Monumente dargestellt. Kabinenkomponenten, für die keine hochauflösende 3D Repräsentation vorliegt (z.B. Brennstoffzelle), werden mit einfachen Geometrieelementen (z.B. Block) dargestellt.

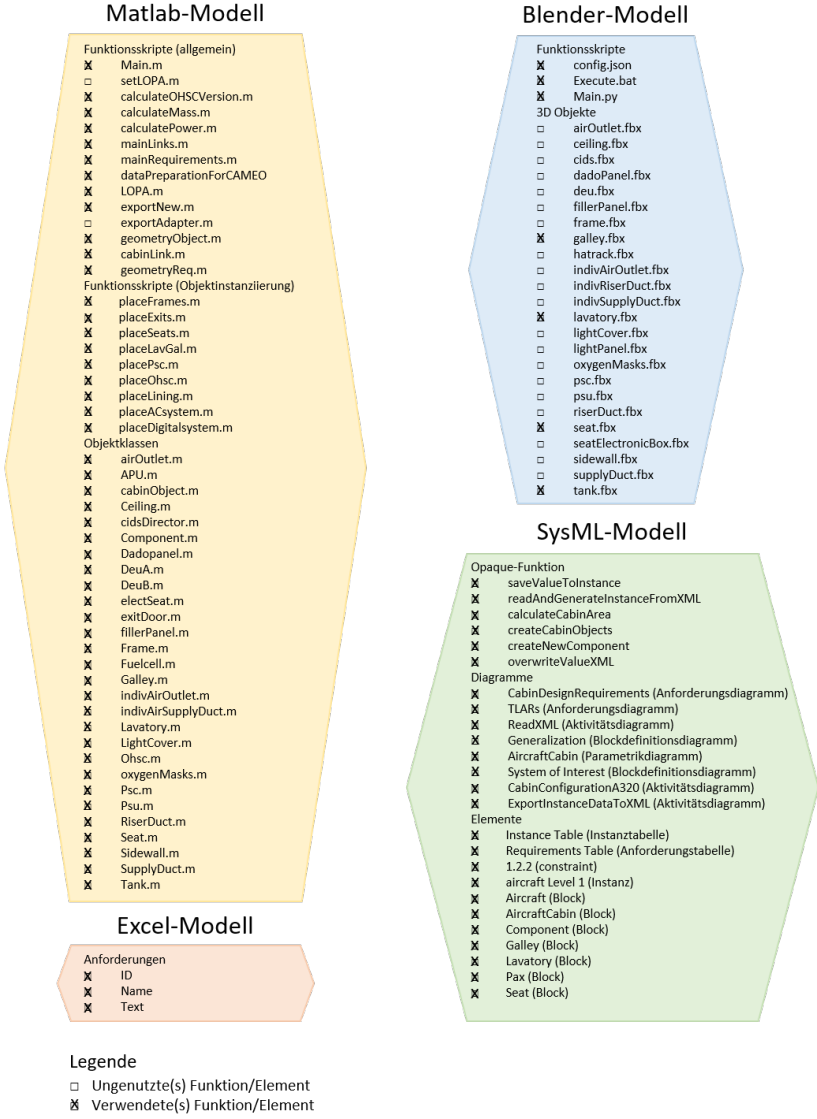


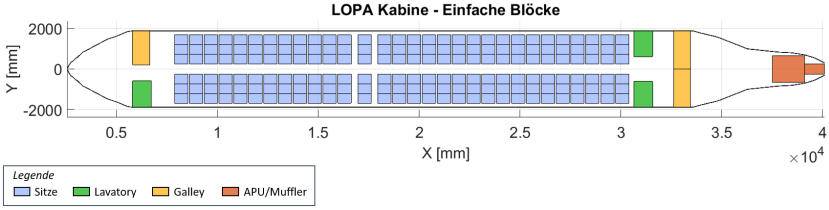

Abbildung 8.6: Übersicht der heterogenen Modelle inklusive Funktionen und Elemente innerhalb des Partialmodells Kabine für Fallstudie I.

Damit ist die Vorstellung des Partialmodells der Kabine abgeschlossen. Einblicke in die beiden Partialmodelle des Wasserstofftanks und des Brennstoffzellensystems sind den Abschnitten A.6.2 und A.6.3 im Anhang zu entnehmen.

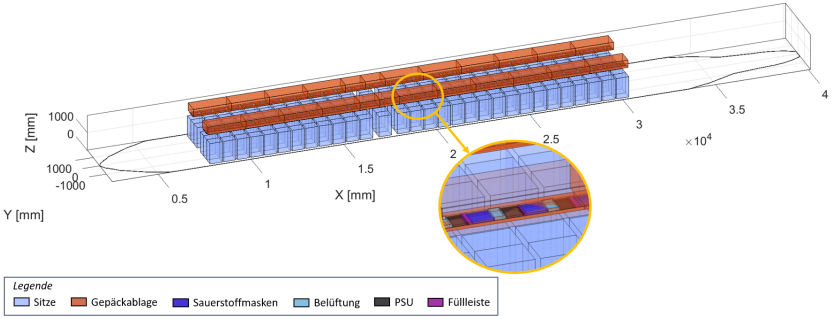
### 8.2.2 Ergebnisse der Variantenbildung

Die Ergebnisse der Variantenbildung für zwei unterschiedliche Flugzeugkonfigurationen sind im Folgenden aufgeführt. Bei der Darstellung und Berechnung der Varianten wird das globale Koordinatensystem des Flugzeugs verwendet. Dadurch befindet sich der Ursprung des Koordinatensystem 2,54 m vor der Flugzeugnase. Die Kabine startet dadurch nach dem Cockpit bei x=5,794 m.

Basis für die Auslegung stellt eine A320neo Rumpfstruktur dar. In der ersten Variante wird mit der entwickelten Methode eine nach heutigen Standards ausgelegte A320-Kabinenkonfiguration generiert, bei der die Energieversorgung der Kabinensysteme mithilfe einer Hilfsturbine durchgeführt wird. Abbildung 8.7 zeigt die Ergebnisse für Variante 1a. Abbildung 8.7a zeigt die mit Matlab generierte 2D Darstellung der Positionen für die Sitze (hellblau), Toiletten (grün) und Bordküchen (gelb) innerhalb der Kabine. Zusätzlich ist die Position der Hilfsturbine inkl. des Schalldämpfers (Muffler) in orange dargestellt. Alle Komponenten sind als einfache Geometrien (z.B. Rechteck) dargestellt. Im Bereich der mittleren Notausgänge wird bei den Sitzen ein größerer Sitzabstand verwendet, um im Fall einer Notevakuierung den geforderten Freiraum vor den Ausgängen zu gewährleisten.




(a) 2D LOPA Darstellung der Kabine.

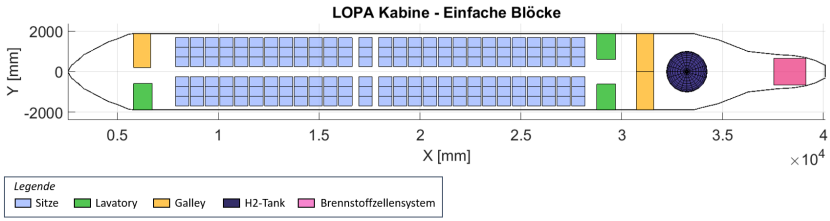



(b) 3D Layout der Sitzverteilung und regulären Gepäckablagen inkl. der Servicefunktionen.

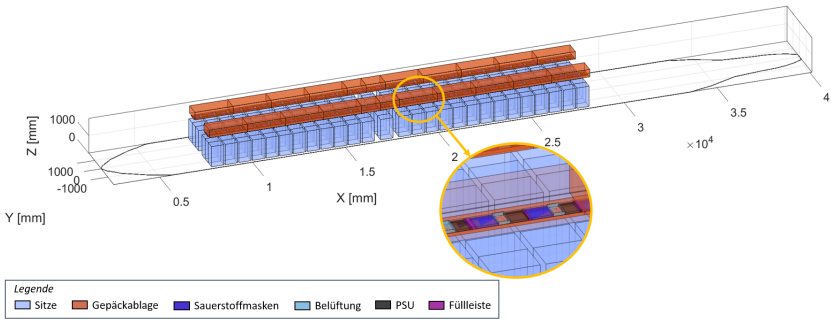
Abbildung 8.7: Ergebnisse der Variantenbildung mit Partialmodellen für Variante 1a - Energieversorgung durch eine Hilfsturbine, dargestellt in Matlab.

Ein Einblick in die Anordnung der Passagierservicefunktionen und der Gepäckablage (dunkelorange) ist in Abbildung 8.7b gegeben. Bei den Notausgängen werden zwei verkürzte Gepäckablagen installiert, die sowohl Stauraum für Equipment der Crew als auch für Gepäck des Passagiers bieten. Sowohl bei Variante 1a als auch bei Variante 1b wird eine Standard-Gepäckablage (regular hatrack) verwendet. Die Vergrößerungsansicht zeigt die Servicekomponenten Sauerstoffmasken (dunkelblau), individuelle Belüftungen (hellblau), Füllleisten (pink) und Passagier Service Einheiten (Passenger Service Unit, PSU) (schwarz). Die PSU umfasst dabei die Leselichter, den Lautsprecher, die Hinweiszeichen und den Flugbegleiterrufknopf.

Die Ergebnisse der zweiten Flugzeugvariante sind in Abbildung 8.8 dargestellt. Dabei wird die Hilfsturbine durch ein Wasserstoff-Brennstoffzellensystem ersetzt. Das 2D Kabinenlayout ist in Abbildung 8.8a dargestellt. Für die Platzierung des Brennstoffzellensystems (pink) wird der Bauraum der APU im hintersten Abschnitt des Flugzeugs genutzt. Der Wasserstofftank (lila) wird im hinteren Kabinenbereich platziert. Dies führt zu einer Reduzierung der nutzbaren Kabinenfläche für die Anordnung der Kabinenmonumente. Die Anordnung der Standard-Gepäckablage inklusive der Passagierservicefunktionen und der Sitze innerhalb der Flugzeugkabine ist in Abbildung 8.8b dargestellt.




(a) 2D LOPA Darstellung der Kabine.




(b) 3D Layout der Sitzverteilung und regulären Gepäckablagen inkl. der Servicefunktionen.

Abbildung 8.8: Ergebnisse der Variantenbildung mit Partialmodellen für Variante 1b - Energieversorgung durch ein Wasserstoff-Brennstoffzellensystem, dargestellt in Matlab.

Weiterhin wird für beide Varianten jeweils ein 3D Modell erzeugt. Abbildung 8.9 ist die automatisch mit der Grafiksoftware Blender generierte 3D Darstellung der Flugzeugvariante 1a zu entnehmen. Sie zeigt nur die Kabinenmonumente und die APU

inkl. Muffler (rot) sowie die angedeutete Flugzeugrumpfstruktur. Die Verkleidung und die Gepäckablagen sowie Kabinensubsysteme werden in dieser ersten Fallstudie mit berechnet, aber nicht visuell dargestellt.
Abbildung 8.10 gibt die 3D Geometriedarstellung der Kabinenmonumente, des Wasserstofftanks (lila) und des Brennstoffzellensystems (pink) sowie die angedeutete Flugzeugrumpfstruktur ohne die Verkleidung und Kabinensubsysteme wieder.

Die Ergebnisse der Bewertungskriterien jeder Variante sind in Tabelle 8.4 aufgeführt. Um die Methode zu validieren, wird ein bestehendes Flugzeug in Variante 1a entworfen. Das gewählte Flugzeug ist der Airbus A320, welches als Referenz für viele Initialparameter Verwendung findet. Bei Variante 1a können bei einer maximalen Einklassenbestuhlung 180 Passagiere transportiert werden. Bei Variante 1b reduziert sich der Wert auf 162 Passagiere. Das Gewicht für die Monumente (Sitze, Bordküchen und Toiletten) beträgt bei Variante 1a 2400 kg und für Variante 1b 2220 kg. Das Gewicht für die Passagiere und ihr Gepäck entspricht der Masse, die als Zuladung unter anderem transportiert werden kann. In der Variante 1b können aufgrund des neuen Energiesystems im Vergleich zu der Variante 1b 10 % weniger Passagiere und Gepäck mitgeführt werden. Das Gewicht der APU beträgt 145 kg [Sch15c]. Für die Versorgung der Kabine im ersten Fall wird eine Leistung von 87 kW benötigt. Im Vergleich dazu beträgt das Gesamtgewicht des Wasserstoff-Brennstoffzellensystems 619 kg, dass sich aus dem Brennstoffzellensystem (170 kg) und dem befüllten Wasserstofftank (449 kg) ergibt [FAF+23]. Weitere Gewichte für die Ventile oder Pumpen bei der Massenberechnung des Tanks sind nicht berücksichtigt [FAF+23]. Für die Versorgung der Kabine muss eine Leistung von 85,2 kW bereitgestellt werden.

Tabelle 8.4: Ergebnisse der Bewertungskriterien für die Flugmission für beide Varianten.

| **Bezeichnung** | *Variante 1a* **Wert** | *Variante 1b* **Wert** |
|---|---|---|
| Anzahl Passagiere | 180 | 162 |
| Gewicht Kabinenmonumente | 2400 kg | 2220 kg |
| Gewicht Gepäck & Passagier | 17100 kg | 15390 kg |
| Gewicht Energiesystem | 145 kg | 619 kg |
| Benötigte Kabinenleistung | 87,0 kW | 85,2 kW |

Für die Validierung des Kabinenmodells wird ein Vergleich der ermittelten Werte der Variante 1a (A320) mit den Werten in der Literatur für eine A320 durchgeführt (Tabelle 8.5). Im Flugzeugentwurf wird bei der Erstauslegung einer Flugzeugvariante eine Standardkonfiguration mit maximaler Passagierauslastung in einer Einklassenbestuhlung untersucht. Für die Machbarkeitsanalyse der Konfiguration sind daher die Anzahl der Passagiere und die Massen der Sitze und Monumente gefordert. Daher werden für eine Validierung des Kabinenmodells diese beiden Aspekte verglichen.

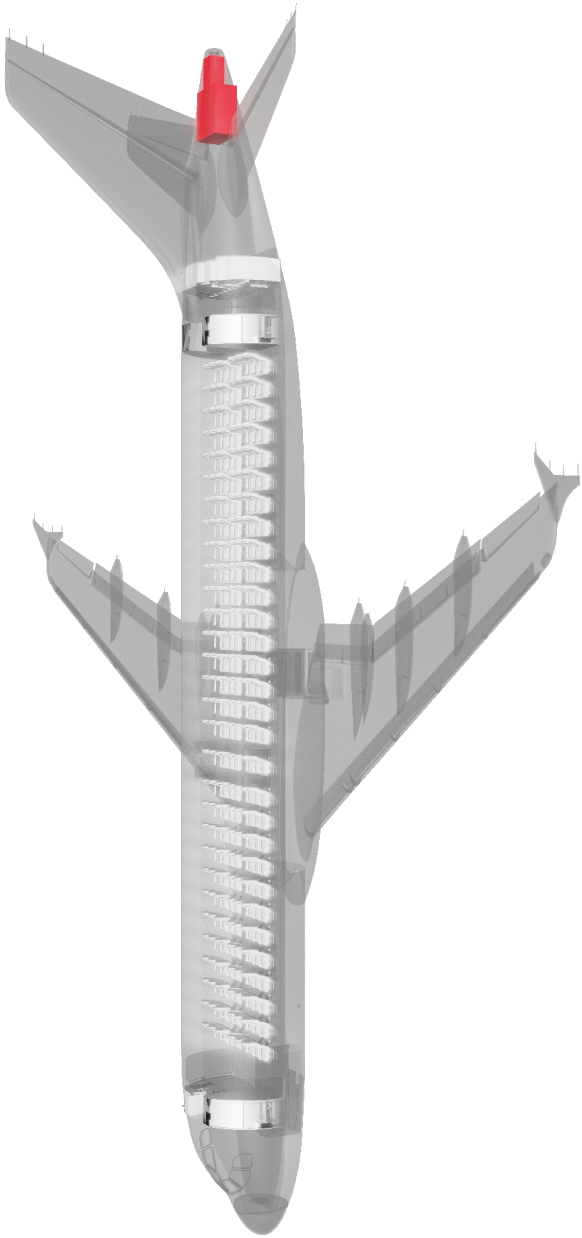


Abbildung 8.9: 3D Darstellung der Flugzeug- und Kabinenkonfiguration (nur Monumente) in Blender für Variante 1a. Die APU und der Muffler sind rot hervorgehoben.

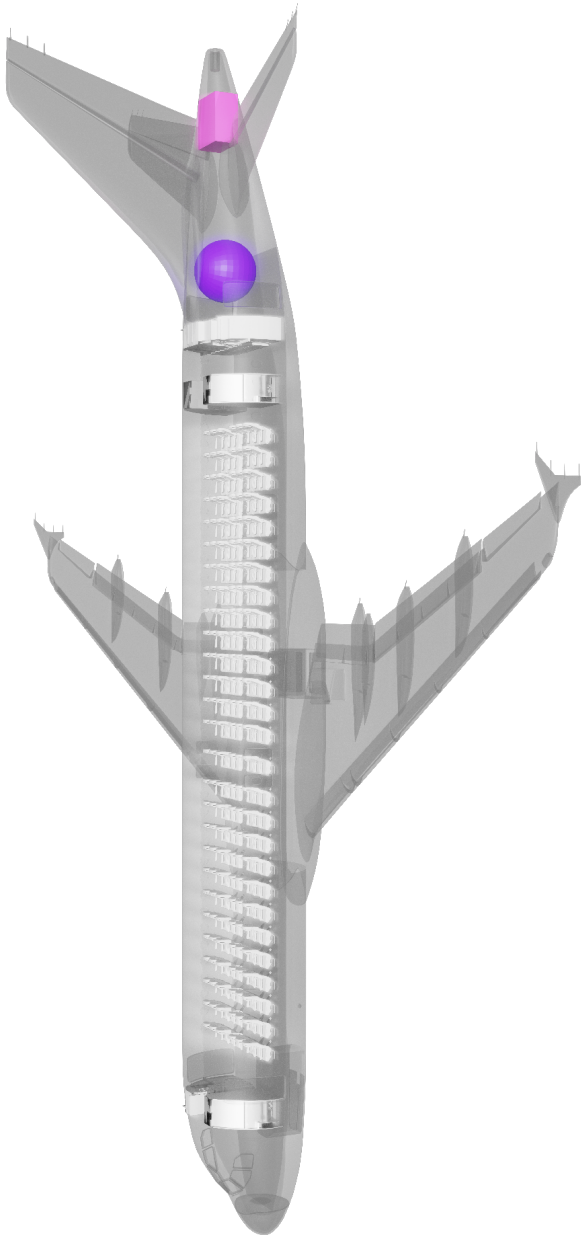


Abbildung 8.10: 3D Darstellung der Flugzeug- und Kabinenkonfiguration (nur Monumente) in Blender für Variante 1b. Die Brennstoffzelle (pink) und der Wasserstofftank (lila) sind farblich hervorgehoben.

Ein Aspekt der Validierung betrachtet die Massen. Die Quelle der Komponentenmassen sind Fuchte [Fuc14] und das Handbuch für Gewicht und Balance von Airbus [Air88]. Das Gewicht der Sitze und Toiletten adressiert Airbus unter dem Begriff „Operator Items“. Unter diesem Begriff werden die Sitze, die Waschräume, das In-flight Entertainment und die Notfallausrüstung zusammengefasst. Für die Sitze und die Küchen ergibt sich für die mit der Methode ausgelegte Flugzeugvariante 1a ein Wert von 2250 kg. Damit weicht der ermittelte Wert für die Sitze und Küchen mit 20 % von dem Gewichtswert der A320 in der Literatur ab (2750 kg [Fuc14, Air88]). Allerdings ist dies akzeptabel, da die getroffenen Annahmen für die Massen dem neuesten Stand der Technik entsprechen und zu Einsparungen beim Sitz (3er Sitzreihe: früher 41 kg, heute 30 kg (-31 %)) führen. Ein weiterer Punkt in der Validierung betrifft die Auslegung der Kabine. Ein Vergleich der LOPA aus Abbildung 8.7a und der 3D Darstellung aus Abbildung 8.7b mit den entsprechenden LOPA und 3D Darstellungen des Airbus A320 Manuels für Flugzeugmerkmale zeigen direkte Übereinstimmungen [Air20]. Zudem ergibt sich mit der Methode und den getroffenen Annahmen für Variante 1b ein Passagierkompartmentvolumen von 144 $m^3$, welches mit dem angegebenen Volumen von Airbus mit 139 $m^3$ mit ausreichender Genauigkeit übereinstimmt [Air20].

Tabelle 8.5: Vergleich der Literaturangaben mit den Ergebnissen der Variante 1a. Die Literaturangaben wurden [Fuc14], [Air88] und [Air20] entnommen.

| | *A320 Literatur* | *Variante 1a* | *Abweichung* |
|---|---|---|---|
| **Bezeichnung** | **Wert** | **Wert** | **%** |
| Anzahl Passagiere | 180 | 180 | 0,0 |
| Gewicht Sitze und Küchen | 2750 kg | 2250 kg | -20,0 |
| Passagierkompartmentvolumen | 139 $m^3$ | 144 $m^3$ | +3,5 |

Die Ergebnisse der vorläufigen Dimensionierung und Massenabschätzung ermöglichen eine frühzeitige Bewertung der Flugzeugvariante im Rahmen des konzeptionellen Entwurfs. Dabei können mit Hilfe der Methode in kürzer Zeit mehrere Flugzeugvarianten für ein definiertes Missionsprofil durchgerechnet und die Breite des Entwurfsraums abgesucht werden. Zudem können erste quantitative und geometrische, bauraumbezogene Analysen neuer Flugzeugvarianten durchgeführt und die Ergebnisse mit den Missionsanforderungen verglichen werden. Die Bewertungskriterien unterstützen dabei die Entwurfsuntersuchung. In der gezeigten Fallstudie ist bei der gleichen Mission eine Versorgung des Flugzeugs mit einem Wasserstoff-Brennstoffzellensystem möglich, unter der Bedingung, dass nur 162 Passagiere transportiert werden können. Ausgehend von der Gewichtung der Anforderungen an das Gesamtsystem, könnte in einem nächsten Schritt beispielsweise die Mission angepasst oder ein anderes Flugzeugmodell (größerer Bauraum) untersucht werden, um einen Transport von 180 Passagieren zu ermöglichen.

Die Durchlaufzeiten für beide Varianten sind in Tabelle 8.6 dargestellt. Hierbei wurde die Animationsgeschwindigkeit für die Simulation im Cameo Systems Modeler auf den Wert 95/100 eingestellt. Die Zeiten wurden manuell mit einer digitalen Stoppuhr gemessen. Jede Variante wurde jeweils zehn Mal wiederholt durchlaufen. Dabei ist die

Standardabweichung bei den Messungen jedes Prozessschrittes maximal im Bereich $\pm$ 2,18 s, die auf Messungenauigkeiten beim manuellen Stoppen zurückzuführen sein könnten. Die Ergebnisse der Messungen sind in Anhang A.6.4 zusammengetragen. In Tabelle 8.6 sind die durchschnittlich gemessenen Zeiten für die Ausführung jedes Partialmodells als auch für die Visualisierung oder für manuelle Tätigkeiten aufgeführt. Bei Variante 1b ist die Ausführung des Tankmodells mit 25 s am schnellsten. Die Auslegung der Brennstoffzelle nimmt bei Variante 1b die längste Zeit (375 s) in Anspruch. Die Ausführung des Partialmodells Kabine unterteilt sich nochmals in die Auslegung mit dem Cameo Systems Modeler und mit Matlab. Für die Variante 1a benötigt das Ausführen der Simulation mit dem CSM 84 s und mit Matlab 28 s. Für Variante 1b fallen die Zeiten etwas kürzer aus (CSM 71 s und Matlab 27 s). Bei beiden Varianten entfallen wiederum 12 s der Simulationszeit des Cameo Systems Modeler auf das Einlesen und Auslesen der Daten aus der XML-Datei (Partialmodell Kabine).

Tabelle 8.6: Ergebnisse der Durchlaufzeiten der einzelnen Prozessschritte für jede Variante.

| **Bezeichnung** | *Variante 1a* **Zeit [s]** | *Variante 1b* **Zeit [s]** |
|---|---|---|
| Partialmodell Tank | - | 25,00 |
| Partialmodell Brennstoffzelle | - | 375,00 |
| Partialmodell Kabine | 112,59 | 98,53 |
| *davon Modell Cameo Systems Modeler* | *84,22* | *71,26* |
| *davon Modell Matlab* | *28,37* | *27,27* |
| 3D-Blender Visualisierung | 106,11 | 95,91 |
| Manuelle Tätigkeiten | 200,00 | 200,00 |
| Gesamt | 418,70 | 794,44 |

Manuelle Tätigkeiten werden für beide Prozessdurchläufe konservativ mit 200 s abgeschätzt. Dazu gehören das Öffnen des Cameo Systems Modeler für die drei Partialmodelle und das Starten des Python-Codes für die Ansteuerung der 3D Modellierung in Blender. Die Zeit für die 3D Geometriemodellierung in Blender beträgt bei Variante 1a 106,11 s. Bei Variante 1b verkürzt sich die Zeit auf 95,91 s aufgrund der geringen Anzahl zu modellierender Kabinenobjekte. Insgesamt ergeben sich für die Modellierung und Visualisierung einer Flugzeugvariante Gesamtprozesslaufzeiten von 418,70 s (06:58 min) für Variante 1a und 794,44 s (13:14 min) für Variante 1b. Damit ist die Prozesszeit der Variantenauslegung auf Gesamtsystemebene unter Berücksichtigung von Missionsanforderungen für eine vorläufige Dimensionierung in der Größenordnung anderer Tools und Methoden einzuordnen. Experteninterviews mit Nutzern des Softwaretools OpenAD für detaillierte Flugzeugkonzeptstudien beim DLR ergaben Prozesszeiten von 30 s bis 30 min für den Vorentwurf von Flugzeugkonfigurationen, je nach Berücksichtigung der Details [FHAZ23, WAS$^+$20]. Föderierte

Partialmodelle ermöglichen damit low-fidelity Analysen mit einer kurzen Prozesslaufzeit, um umfangreiche Designraumuntersuchungen für Flugzeugvarianten durchzuführen. Der Einsatz dieser erhöht nicht die Analysezeit im Vergleich zum Stand der Technik.

Der Fokus der Fallstudie I lag auf dem Zusammenspiel der Partner und deren Modellen. Wie bereits in Kapitel 3.1 erläutert, sind am (Flugzeug-)Kabinenentwurf eine Vielzahl an Experten unterschiedlicher Fachdisziplinen und viele Zulieferer (große Lieferantenkette) beteiligt. Mit Centerline Design und der HAW Hamburg als externe Partner im Projekt MIWa wurde daher die Zusammenarbeit in einer Zulieferkette simuliert. Zusammenfassend wurde in der Fallstudie I gezeigt, dass expertenübergreifend Daten ausgetauscht und sowohl Partialmodelle als auch interne heterogene Modelle miteinander gekoppelt werden konnten. Der in dieser Arbeit entwickelte Ansatz föderierter Partialmodelle basiert auf der Verteilung von Wissen auf mehrere Modelle und der anschließenden Kopplung dieser für die gemeinsame Modellierung und Bewertung eines Gesamtsystems. Besonders für die Zusammenarbeit mit externen Partnern oder Experten anderer Fachdisziplinen ist dieser Ansatz der Kooperation von Vorteil. Im Projekt MIWa wurde gezeigt, dass die entwickelte Methode einer dezentralen Verteilung des Wissens im industriellen Kontext anwendbar ist, wenn die Schnittstellen und Austauschparameter eindeutig definiert sind. Die kurze Prozesszeit von 7 min bis 13 min für die Auslegung einer Variante begünstigt zudem die schnelle Analyse von Auswirkungen im Gesamtmodellkontext, wenn Partner Änderungen in ihren Modellen vornehmen. Zudem zeigte sich während der Fallstudie, dass die interne Strukturierung der SysML-Modelle nach dem EVA-Prinzip die Detektion von Fehlern und die Durchführung von Modellanpassungen begünstigt. Gleichzeitig verwendeten alle Partialmodelle für die interne Kopplung der heterogenen Modelle (z.B. CMS und Matlab/Simulink und Python) das Modellelement Opaque Aktion des Cameo Systems Modeler. Damit ist allerdings nur eine begrenzte Anzahl an Schnittstellen realisierbar. Für die Anbindung weiterer Modelle, z.B. FEM-Modelle müssen spezifische Lösungen gefunden werden. Dennoch ist eine Kopplung und der damit verbundene Aufwand für ganzheitliche Analysen vielversprechend.

#### Zwischenfazit

Für die Auslegung der beiden Varianten wurde dieselbe Mission und Reichweite als Ausgangslage gewählt. Die Auslegung der Variante 1a zeigt, dass mit der entwickelten Methode eine heutige A320-Referenzflugzeugkonfiguration mit Eingabe zertifizierter Systemkomponenten erzielt werden kann. Bei Variante 1b wurden der Parametersatz eines bereits dimensionierten Subsystems ausgetauscht und weitere Partialmodelle angeschlossen, mit denen eine Berechnung und Dimensionierung neuer Subsysteme und damit die Auslegung einer neuen Flugzeugkonfiguration möglich ist. Die Validierung anhand realer Daten ermöglicht einen Vergleich einzelner Komponentenmassen, anstelle der Gesamtmasse, und mit den resultierenden Leistungsdaten eine Wettbewerbsanalyse der Flugzeugvarianten.

Die aufgeführten Ergebnisse zeigen die Werte nach einem Auslegungsdurchlauf mit der gewählten Referenzmission an. Dabei wurden alle jeweils relevanten Partialmodelle genau einmal angesteuert und ausgeführt. Weitere Iterationen können durchlaufen werden, um eine optimale Auslegung zwischen der benötigten Kabinenleistung,

der Anzahl Passagiere, der Baugröße des Energiesystems und der Flugmission zu finden. Bei der zweiten Variante (1b) beispielsweise ergab die Auslegung eine geringere geforderte Kabinenleistung aufgrund der reduzierten Anzahl an zu transportierender Passagiere. Als Eingangsparameter kann nun die im ersten Durchlauf berechnete, reduzierte Kabinenleistung genutzt werden. Dieser Wert hat wiederum Einfluss auf die Größe des Wasserstoff-Brennstoffzellensystems. Nach dem Durchlauf mehrerer Iterationen kann dadurch die optimale Flugzeugauslegung für diese Variante gefunden werden.

Ein Faktor für die Bewertung des Modellierungsansatzes ist die Durchlaufzeit. Für die Auslegung einer Variante und damit die Ansteuerung sowie Ausführung mehrerer hintereinander geschalteter Partialmodelle beträgt die Prozesszeit zwischen 7 min und 13 min. Die manuellen Tätigkeiten im Prozess können zukünftig durch die automatisierte Ansteuerung (siehe Abschnitt 7.5) über eine externe GUI reduziert werden. Zudem machen die Simulation bzw. Berechnungen in Matlab bei den Partialmodellen Kabine und Brennstoffzelle einen großen Anteil aus. Computer mit besserer Rechenleistung können hier Abhilfe schaffen und die Berechnungen beschleunigen. Als Resultat kann damit die Gesamtdurchlaufzeit des Prozesses reduziert und der Prozess kompakter gestaltet werden.

## 8.3 Fallstudie II: Customizing-Studie in der Kabine

In der zweiten Fallstudie wird mit der in dieser Arbeit entwickelten Methode das Customizing in der Kabine untersucht. Der Wunsch des Kunden zur Individualisierung ist hoch und in der Luftfahrt für Großpassagierflugzeuge nur über die Kabine realisierbar. Daher wird in dieser Studie die Anpassung der Kabine einer nach dem Stand der Technik ausgelegten Flugzeugvariante mit spezifischen Kundenwünschen aus Sicht des Kunden simuliert.

### 8.3.1 Beschreibung des Anwendungsfalls und des Auslegungsprozesses

Die Durchführung der Fallstudie wird nach den Schritten aus Abschnitt 8.1.1 strukturiert. Im Folgenden werden die drei Schritte Definition, Vorgehen und Anwendung der Methode und deren Artefakte vorgestellt. Anschließend werden die Ergebnisse dokumentiert und analysiert.

#### Definition

In diesem Anwendungsfall wird die Standardkonfiguration der Kabine modifiziert. Basis für die Änderungen innerhalb der Kabine bildet der generierte Datensatz (XML-Datei) der Flugzeugvariante 1b aus der Fallstudie I. Bei dieser Variante wurde die Kabine in einer Einklassenbestuhlung mit der maximalen Anzahl an Passagieren ausgelegt. Zudem wurde für die Kabine eine Standardausstattung der Gepäckablage angenommen. Aus der Basiskonfiguration werden zwei unterschiedliche Customizingvarianten für die Kabine abgeleitet. Als Anwendungsfall werden verschiedene

Gepäckablagen samt unterschiedlicher Systemarchitekturen sowie eine Zweiklassenbestuhlung untersucht. Ziel der Studie ist es, nachträgliche Modifikationen einzelner Subkomponenten durch den Kunden im Rahmen des Customizings zu demonstrieren. Das Customizing wird anhand der Systemgruppe der Gepäckablage untersucht.

### Vorgehen

Die Gepäckablage bietet dem Passagier Stauraum für sein Gepäck. Zusätzlich werden unterhalb dieser die Passagierservicefunktionen entweder direkt oder über den Passagierservicekanal installiert. Die Passagierservicefunktionen können in zwei Kategorien unterteilt werden. Die erste Kategorie umfasst alle gesetzlich vorgeschriebenen Funktionen. Dazu gehören die Sauerstoffmasken, Anschnallzeichen, Hinweiszeichen (z.B. Nichtraucher-Flug) und Lautsprecher für Sprachansagen. Die zweite Kategorie beschreibt Servicekomponenten, die dem Komfort der Passagiere dienen [Fuc18]. Dazu gehören das Leselicht, der Flugbegleiterrufknopf und die individuelle Belüftung. Anordnung und Anzahl der Komponenten variieren mit der Passagierkonfiguration und dem Sitzabstand [Fuc18]. Leerräume zwischen den Komponenten werden mit Füllleisten verkleidet. Abbildung 8.11 zeigt eine reguläre Gepäckablage mit dem Servicekanal und den Servicefunktionen sowie den Füllleisten.

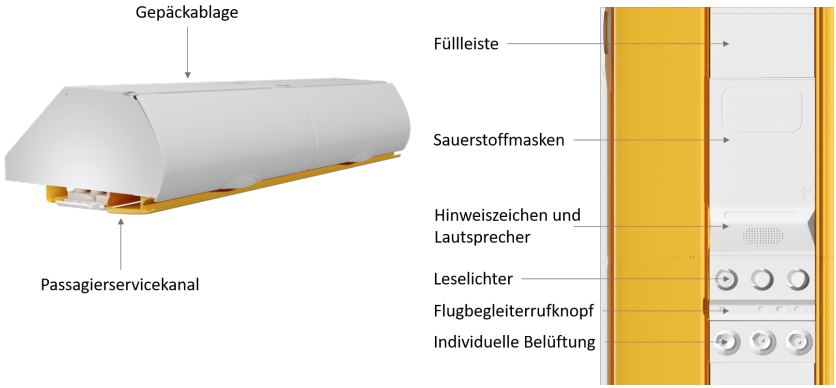



Abbildung 8.11: 3D Geometriedarstellung der Systemgruppe Gepäckablage, Passagierservicekanal, Füllleisten und Passagierservicefunktionen.

In dieser Studie werden drei unterschiedliche Typen der Gepäckablage betrachtet. Die Form der Gepäckablage wird unter anderem bestimmt durch die Rumpfkontur und durch Features (z.B. großer Stauraum). Je nachdem wie Monumente, Notausgänge oder Sitzklassen in der Flugzeugkabine verteilt werden, variiert ebenfalls die Länge der Gepäckablage. Durch die genannten Einflüsse entstehen eine Vielzahl an Sondergrößen bei der Gepäckablage. Um die Auswirkungen und potentiellen Vorteile der einzelnen Sonderformen zu untersuchen, werden in dieser Fallstudie exemplarisch drei verschiedene Systemgruppen der Gepäckablage betrachtet. Die Systemgruppen variieren in ihren Architekturen und Bauteilabmessungen. Die drei Gepäckablagevarianten sind in Abbildung 8.12 dargestellt. Die oberste Reihe zeigt die Darstellung

der Gepäckablagen in einer Schrägansicht, die mittlere Reihe in einer Seitenansicht. In der untersten Reihe sind die Gepäckablagen mit den zugehörigen Passagierservicefunktionen in den verschiedenen Einbauszenarien dargestellt. Bei der regulären Gepäckablage (M) werden die Servicekomponenten in einem Servicekanal unterhalb der Gepäckablage installiert. Im Vergleich dazu verteilen sich die Komponenten bei der größeren Gepäckablage (L) auf zwei Servicekanäle. Bei der extra großen Variante (XL) verteilen sich die Servicekomponenten über den gesamten unteren Bereich der Gepäckablage und werden ohne einen Servicekanal direkt montiert. Darüber hinaus ermöglichen die Gepäckablagen L und XL eine Vormontage der Passagierservicefunktionen, wodurch Einbaukosten und Installationszeiten in der Endmontage reduziert werden können.

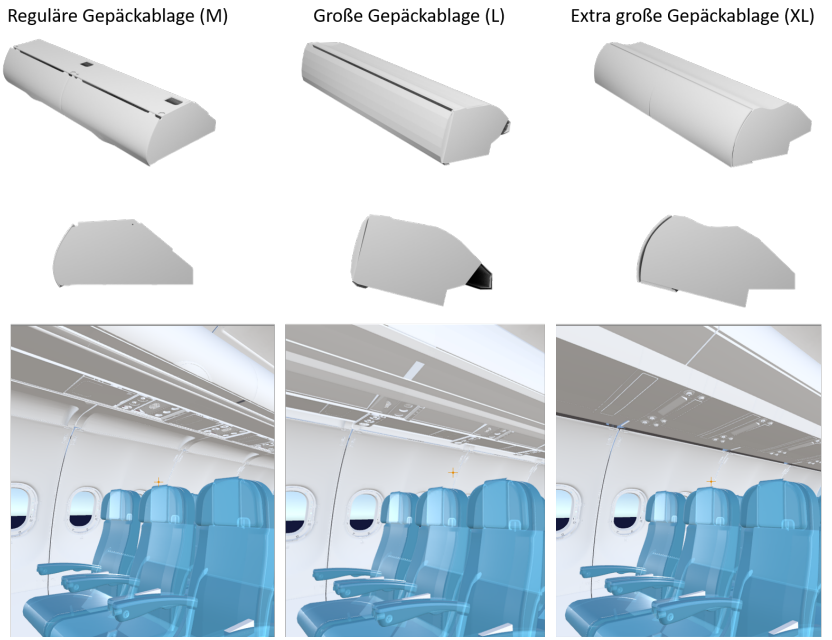
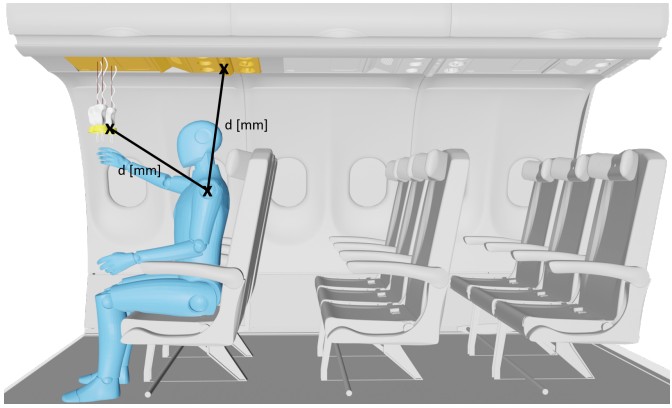



Abbildung 8.12: 3D Geometriedarstellung der drei Gepäckablagevarianten sowie die Anordnung der zugehörigen Passagierservicefunktionen von [FGBN23].

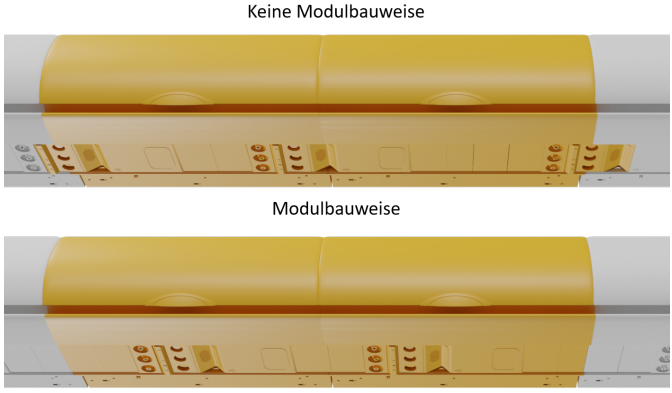
Die Gepäckablage M wurde bereits im vorherigen Anwendungsfall verwendet. Für die erste abgeleitete Customizingvariante wird anstelle des regulären Typs eine größere Gepäckablage ausgewählt. Diese bietet dem Passagier mehr Stauraum für sein Gepäck und die Möglichkeit, die Passagierservicefunktionen auf zwei Servicekanäle aufzuteilen. Bei der zweiten Customizingvariante wird eine extra große Gepäckablage verwendet, die den größten Stauraum bietet. Darüber hinaus wird eine Zweiklassenbestuhlung bestehend aus Business- und Economyklasse untersucht. Als Annahme wird die Anzahl der Reihen mit Businessklassensitzen auf den Wert vier gesetzt. Als Referenz dient die Anordnung der Businessklasse im Airbus A321LR von der Airli-

ne SAS (Schmalrumpfflugzeug, 1 Gang), bei dem die Sitze abwechselnd in einer 1:1 bzw. 2:2 Konfiguration verbaut sind [Fra21].

Die Bewertung der ausgelegten Kabinenkonfigurationen erfolgt anhand zweier Kriterien (Abbildung 8.13). Das erste Kriterium ist der Passagierkomfort. In dieser Arbeit wird der Komfort anhand der durchschnittlichen Erreichbarkeit der Bedienelemente der Passagierservicefunktionen durch den sitzenden Passagier gemessen. Die Entfernung zwischen den Bedienelementen und dem Passagier errechnet sich aus der Schulterposition der sitzenden Person und der Position des Bedienelements. Für die anthropometrischen Daten des Menschen werden die Werte eines durchschnittlichen




(a) Erreichbarkeit der Bedienelemente der Passagierservicefunktionen.




(b) Gegenüberstellung der nicht modularen und modularen Bauweise für die Gepäckablage.

Abbildung 8.13: Vorstellung der beiden Bewertungskriterien für die Kabinenkonfigurationen.

Europäers (50. Perzentil) angenommen [DIN13]. Daraus ergibt sich als Referenz für den heutigen Passagierkomfort-Standard für die Distanz ein Wert von 715 mm. Ab-

bildung 8.13a veranschaulicht diesen Zusammenhang am Beispiel der Erreichbarkeit d [mm] der Sauerstoffmasken und der Bedienelemente der Leselichter.

Das zweite Kriterium betrifft die Modulfähigkeit der Systemgruppe. Derzeit wird die Systemgruppe einzeln in der Kabine montiert. Dabei werden zuerst die Gepäckablage und der Servicekanal in die Kabine eingebaut, anschließend die Füllleisten und die Komponenten der Servicefunktionen montiert. Eine modulare Bauweise, bei der sich alle Systemkomponenten innerhalb der geometrischen Abmaße der Gepäckablage befinden, ermöglicht eine Vormontage der Systemgruppe und bietet Einsparpotential bei den Einbauzeiten und -kosten [Fuc18]. Abbildung 8.13b veranschaulicht die modulare Bauweise. Während der Platzierung der Komponenten der Passagierservicefunktionen überprüft das Matlab-Modell die Modulfähigkeit. Sind alle Komponenten innerhalb des Bauraums der Gepäckablage platzierbar, erfüllt die Kabinenkonfiguration das Kriterium der modularen Bauweise. Sobald mindestens bei einer Gepäckablage dies nicht erfüllt werden kann, verfehlt die Konfiguration die Anforderung an eine modulare Bauweise.

Der Prozess für die Customizing-Studie ist in Abbildung 8.14 schematisch dargestellt. Für die Untersuchung von Kabinenkonfigurationen wird nur das Partialmodell der Kabine im Entwurfsprozess angesteuert. Die Auslegung der Grundvariante 1b ist bereits durchgeführt und deren Parameter abgespeichert worden. Dieses Vorgehen ermöglicht, dass zu Beginn der Auslegung der Gesamtsystemvariante nicht alle Variationen des Susbsystems bekannt sein und berechnet werden müssen. Variationen in den Subsystemen wie der Kabine können im Nachgang erweitert und untersucht werden, wodurch Rechenzeit eingespart und die modulare Erweiterung innerhalb der Partialmodelle begünstigt wird.

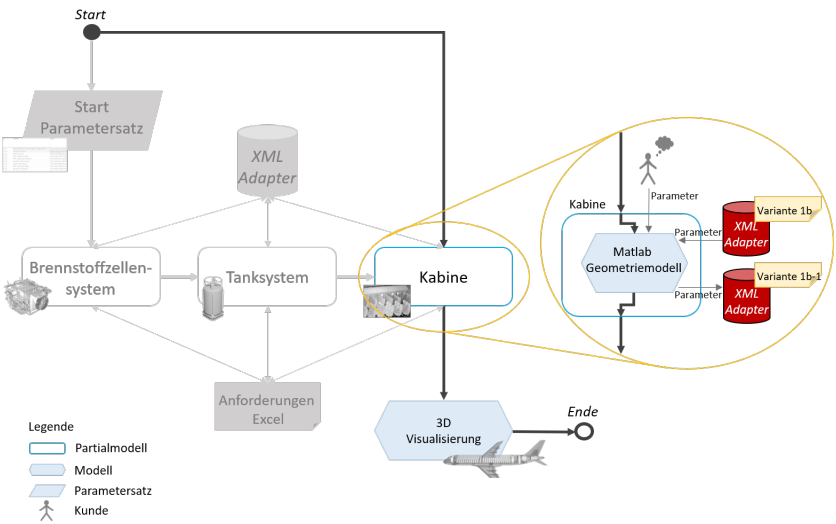



Abbildung 8.14: Prozess für Änderungen an der Kabinenkonfiguration im Rahmen des Customizings am Beispiel der Variante 1b-1. Die Auslegung der Grundvariante 1b ist bereits abgeschlossen, sodass nur das Partialmodell der Kabine angesteuert wird.

Der Prozess beginnt mit dem Partialmodell der Kabine. Allerdings wird beim Customizing nur das Matlab-Modell ausgeführt. Zuerst werden die zu ändernden Parameter durch den Kunden festgelegt. Dazu gehören zum Beispiel die Anzahl der Sitzreihen für die Businessklasse und der Typ der Gepäckablage. Durch die manuelle Eingabe der neuen Werte wird im Matlab-Modell der Customizingprozess aktiviert. Im Vergleich zum Variantenprozess werden die Informationen für die Auslegung der Kabine nicht aus dem SysML-Modell, sondern direkt durch Matlab selber bereitgestellt. Sofern die Änderungen nicht die Sitze, Bordküchen oder Toiletten betreffen, verläuft der interne Auslegungsprozess in Matlab gleich. In diesem Fall werden die Parameter für die Platzierung der Sitze aus der XML-Datei der Basiskonfiguration (Variante 1b) ausgelesen. Wird hingegen eine Änderung in der Sitzkonfiguration gewünscht (z.B. feste Vorgabe an Sitzreihen für die Businessklasse), werden aus der XML-Datei der Basiskonfiguration die Kabinenflächenparameter ausgelesen. In Matlab wird dann die geometrische Platzierung der Monumente und Sitze neu berechnet.

Im Anschluss an den Auslegungsprozess werden die generierten Parameter aus der Kabinenauslegung in eine neue XML-Datei geschrieben. Mit dieser wird dann im letzten Schritt die neu ausgelegte Kabinenvariante mit Blender als 3D Geometriemodell visualisiert.

#### Anwendung der Methode und deren Artefakte

Für die Auslegung der Kabinenkonfigurationen wird dasselbe Partialmodell der Kabine verwendet wie in Fallstudie I. Die Artefakte (Modelle und deren Elemente) für das Partialmodell der Kabine wurden bereits in Kapitel 7 und in Abschnitt 8.2.1 vorgestellt. Abbildung 8.15 zeigt die Übersicht aller Funktionen und Elemente innerhalb des Partialmodells der Kabine, die in der Fallstudie II Anwendung finden.

Die Darstellung zeigt die Anwendung der Modelle für die Auslegung beider Kabinenkonfigurationen. Alle Elemente die verwendet wurden, sind mit einem “x“ markiert. Insgesamt wurden für die Auslegung der zwei neuen Kabinenkonfigurationen jeweils 25 Funktionen ausgeführt, 51 Elemente angesteuert und 0 Diagramme verwendet. Im Vergleich zu der Fallstudie I wird beim Customizing das SysML-Modell nicht angesteuert. Stattdessen wird die Neuauslegung der Kabine im Matlab-Modell durchgeführt. Die Funktion zur Datenübergabe an den Cameo Systems Modeler wird daher nicht mehr ausgeführt. Zudem wird die Objektklasse der APU nicht verwendet, da die Flugzeugvariante mit einem Brennstoffzellen- und Wasserstoffsystem ausgestattet ist. Im Gegensatz zu der Fallstudie I wird ein Detailentwurf gefordert. Im Rahmen des Customizings werden für die Bewertung der Kabinenkonfigurationen neben quantitativen Kriterien (Gewicht, Passagieranzahl) auch qualitative Bewertungskriterien (z.B. Optik) abgefragt. Daher verwendet das Blender-Modell für die Visualisierung der Kabinenkonfiguration für alle Kabinenkomponenten hochauflösende 3D Modelle. Zusammen mit der Rumpfstruktur und der Flugzeughülle wird damit ein erster realistischer Eindruck der potentiellen Kabine bereitgestellt.

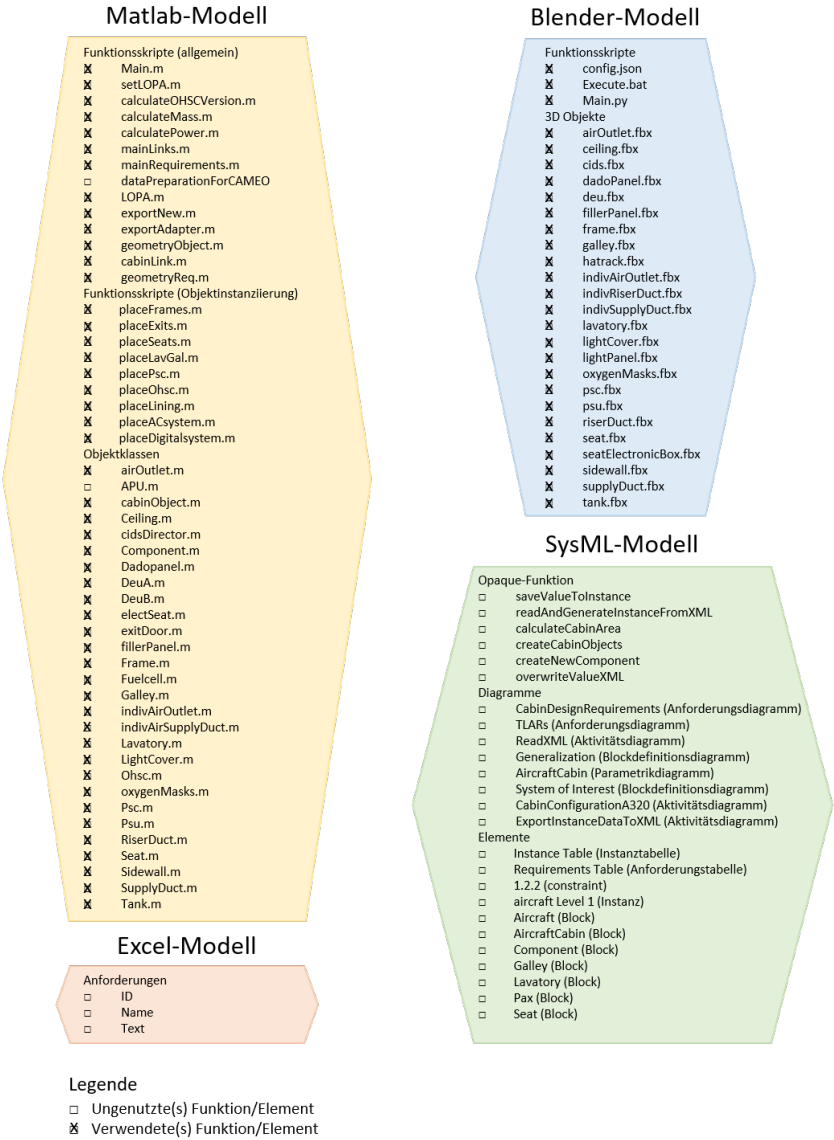
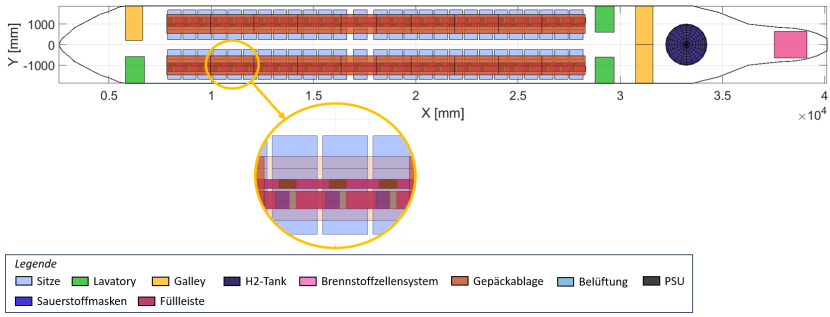


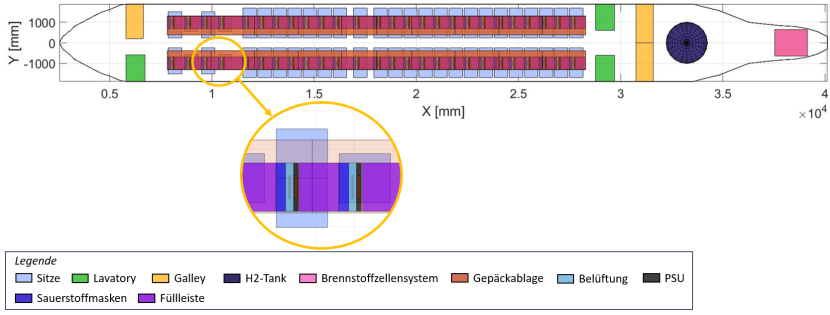

Abbildung 8.15: Übersicht der heterogenen Modelle inklusive Funktionen und Elemente innerhalb des Partialmodells Kabine für Fallstudie II.

### 8.3.2 Ergebnisse der Customizing-Studie

Die Ergebnisse der beiden Durchläufe für die Kabinenkonfiguration im Rahmen des Customizings werden im Folgenden vorgestellt. Bei beiden Varianten bleibt die Platzierung des Wasserstoff-Brennstoffzellensystems und der zur Verfügung stehende Kabinenbauraum für die Platzierung der Kabinenkomponenten unverändert. Abbildung 8.16 zeigt für beide Versionen jeweils die 2D LOPA inklusive der Systemgruppe Gepäckablage in Matlab. In Abbildung 8.16a ist die Einklassenbestuhlung (162 Passagiere) der Wasserstoff-Brennstoffzellen-Flugzeugvariante und die Platzierung der Systemgruppe Gepäckablage L abgebildet. Die Komponenten der Passagierservicefunktionen verteilen sich auf zwei Servicekanäle, wodurch eine kompaktere Anordnung der Komponenten möglich ist.
Bei der weiteren Variante 1b-2 wurde eine Zweiklassenbestuhlung und der Einbau der Gepäckablage XL untersucht. Mit der Vorgabe von vier Reihen mit Businessklassensitzen ergeben sich 12 Sitzplätze in der Businessklasse und 132 Sitzplätze in der Economyklasse (Abbildung 8.16b). Die Anordnung der Komponenten der Servicefunktionen erfolgt über die gesamte untere Fläche der Gepäckablage ohne einen Servicekanal.




(a) 2D LOPA inklusive der Systemgruppe Gepäckablage L für Variante 1b-1 in Matlab.




(b) 2D LOPA inklusive der Systemgruppe Gepäckablage XL für Variante 1b-2 in Matlab.

Abbildung 8.16: Abgeleitete Customizingvarianten der Wasserstoffflugzeugvariante.

Einen detaillierteren Einblick in die Anordnung und das Design der Kabine gibt Abbildung 8.17. Die 3D Modelle der Kabinenkonfigurationen wurden mit der 3D Grafiksoft-

ware Blender erstellt. Abbildung 8.17a zeigt die Verteilung der Passagierservicefunktionen auf zwei Versorgungskanäle. Dadurch ist eine kompaktere Anordnung der Serviceelemente möglich. Die Bedienbarkeit der Leselichter wird dabei allerdings für den Fensterplatz erschwert, da die Platzierung der Komponente in Richtung Gang gewandert ist. Abbildung 8.17b zeigt die Verteilung der Passagierservicefunktionen auf der gesamten Unterseite der Gepäckablage. Dadurch verteilen sich die Bedienelemente über die gesamte Breite und ermöglichen für jeden Sitzplatz eine bessere Erreichbarkeit. Eine Installation innerhalb eines Versorgungskanals entfällt.

(a) Gepäckablagendesign für Variante 1b-1. (b) Gepäckablagendesign für Variante 1b-2.

Abbildung 8.17: 3D Detailansicht der Kabinenkonfigurationen für die beiden Customizingvarianten in Blender.

In Analogie zu Fallstudie I werden in dieser Anwendung ebenfalls die Kabinensysteme abgebildet. Abbildung 8.18 und 8.19 geben jeweils einen Einblick in die gesamte 3D Darstellung der Flugzeugkabine inklusive der Systeme. Dafür wurden für die Kabinensystemkomponenten detaillierte 3D Objekte verwendet und entsprechend der kalkulierten Positionierung aus Matlab platziert. In den Abbildungen sind jeweils die Sitze hellblau, die Gepäckablagen in orange und das Belüftungsverteilungssystem in grün hervorgehoben. Ein Vergleich der beiden Abbildungen miteinander zeigt, dass neben der Berücksichtigung einer Businessklasse bei Variante 1b-2 auch die Systemarchitektur der Belüftung unterschiedlich ist. Bei der Variante mit den extra großen Gepäckablagen ist in der Modulbauweise ebenfalls die Einbindung eines neuen Belüftungssystems vorgesehen. Bei diesem werden die individuellen Luftauslässe für den Passagier unterhalb jeder Gepäckablage jeweils über die Belüftungsrohre des generellen Belüftungsverteilungssystem mit Luft versorgt. Bei Variante 1b-1 wird die Luftversorgung durch eine einzige Versorgungsleitung für alle individuellen Luftauslässe hinter den Gepäckablagen auf einer Flugzeugseite realisiert. Damit ermöglicht

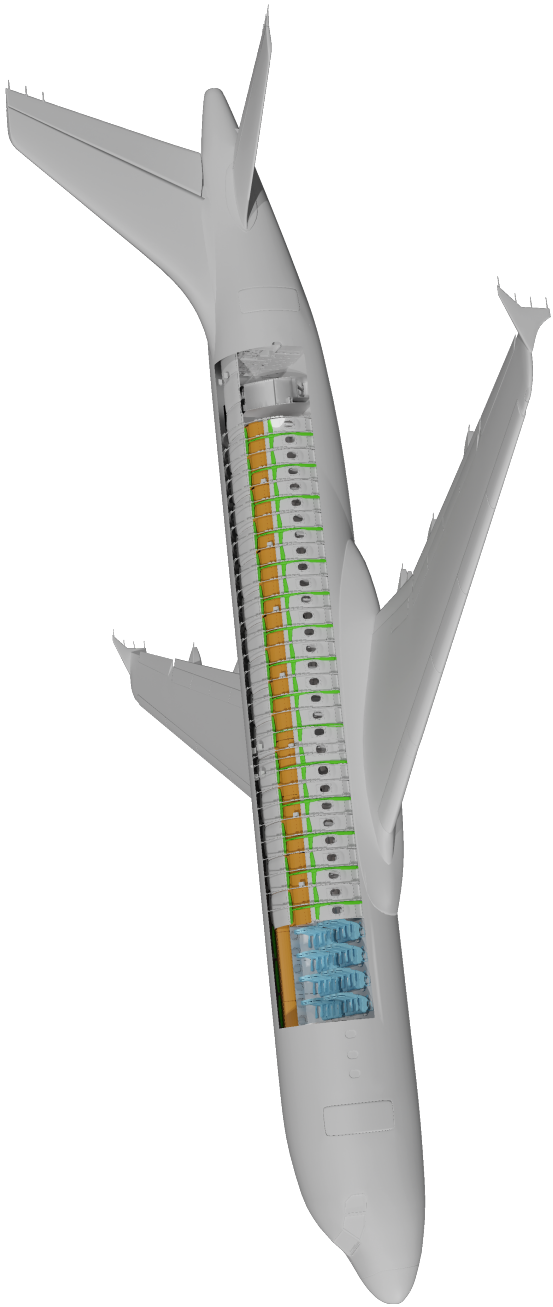


Abbildung 8.18: 3D Darstellung der Flugzeug- und Kabinenkonfiguration mit Systemen in Blender für Variante 1b-1. Die Sitze (blau), die Gepäckablagen (orange) und das Belüftungsverteilungssystem (grün) sind farblich hervorgehoben.

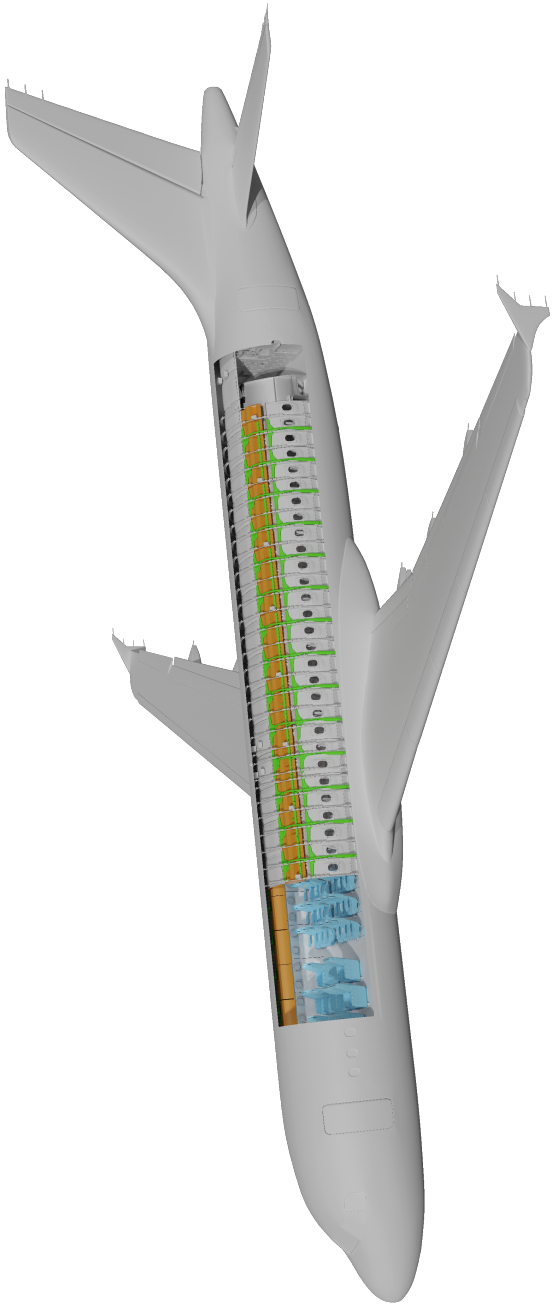


Abbildung 8.19: 3D Darstellung der Flugzeug- und Kabinenkonfiguration mit Systemen in Blender für Variante 1b-2. Die Sitze (blau), die Gepäckablagen (orange) und das Belüftungsverteilungssystem (grün) sind farblich hervorgehoben.

Variante 1b-2 eine Erweiterung der Modulbauweise, indem das Subsystem Belüftung mit betrachtet wird.

Die Ergebnisse der Customizingstudie sind in Tabelle 8.7 zusammengefasst. Als Basiskonfiguration wird die Variante 1b verwendet. Diese stellt mit der Standardauslegung und Anordnung der Komponenten der Servicefunktionen den Vergleichswert für den Passagierkomfort, sodass hier der Wert auf 1 gesetzt wird. Die Ergebnisse der beiden weiteren Varianten werden auf diesen Wert normiert. Für die Variante 1b mit der Gepäckablage M gibt sich für die Anzahl der Passagiere ein Wert von 162. Aufgrund der Abmessungen der einzelnen Servicekomponenten und des begrenzten Installationsbauraums im Servicekanal ist eine Modulbauweise nicht möglich. Bei Variante 1b-1 können ebenfalls 162 Passagiere transportiert werden. Die Distanz zu den Servicefunktionen erhöht sich im Vergleich zu der Basisvariante 1b um 18,5 %. Dadurch sinkt der Passagierkomfort, da die Erreichbarkeit der Bedienelemente für Passagiere mit einer kleineren Körpergröße abnimmt. Allerdings ermöglicht die gewählte Gepäckablage L die Modulbauweise.

Bei der Variante 1b-2 sinkt die Passagierkapazität auf 144. Grund hierfür ist die Zweiklassenbestuhlung. Aufgrund der getroffenen Annahme von vier Businessklassenreihen reduziert sich die Gesamtanzahl an Sitzplätzen. Insgesamt können 12 Passagiere in der Businessklasse und 132 Passagiere in der Economyklasse transportiert werden. Die Distanz zwischen der Sitzposition und den Servicefunktionen verschlechtert sich bei der Gepäckablage XL nur um 0,7 % im Vergleich zur regulären Auslegung. Darüber hinaus ist mit der Gepäckablage XL ebenfalls eine Modulbauweise möglich.

Tabelle 8.7: Ergebnisse der Customizingstudie für Passagieranzahl, Erreichbarkeit der Servicefunktionen, Modulbauweise, Durchlaufzeit und Anzahl benötigter Partialmodelle.

| **Bezeichnung** | *Variante 1b* *Basiskonfiguration* **Wert** | *Variante 1b-1* **Wert** | *Variante 1b-2* **Wert** |
|---|---|---|---|
| Anzahl Passagiere | 162 | 162 | 144 |
| Distanz zu den Servicefunktionen | ∅ 696 mm | ∅ 825 mm (+18,5 %) | ∅ 701 mm (+0,7 %) |
| Modulbauweise | nein | ja | ja |
| Durchlaufzeit (gesamt) | 794,44 s (13 min) | 467,26 s (7 min) | 870,48 s (14 min) |
| *davon Matlab* | *27,27 s* | *18,07 s* | *19,32 s* |
| *davon Blender* | *95,91 s* | *449,19 s* | *851,16 s* |
| Anzahl Partialmodelle | 3 | 1 | 1 |

Beim Passagierkomfort erzielt die Basiskonfiguration 1b die besten Werte. Allerdings ist eine Modulbauweise mit der gegebenen Bauweise nicht möglich. Bei der Variante

1b-2 kann eine Modulbauweise realisiert werden, ohne Einbuße beim Passagierkomfort zu erzielen. Die Abweichung bei der Erreichbarkeit der Servicefunktionen beträgt unter 1 %. Die Modulbauweise ermöglicht dafür eine Reduzierung der Kosten durch resultierende Durchlaufverkürzungen der Einbauzeit in der Endmontage, da die Installation der Servicefunktionen im Rahmen einer Vormontage durchgeführt werden kann.

Ein weiterer Vergleichswert in der Customizingstudie ist die Dauer des Prozesses (Tabelle 8.8). Die Auslegung einer Konfiguration mit anschließender Darstellung als 3D Modell benötigt bei der Basiskonfiguration 13 min, wobei das Matlab-Modell davon 27 s und das Blender-Modell 95 s für die jeweilige Ausführung benötigen (Tabelle 8.7). Bei der 3D Darstellung werden nur die Monumente und die Flugzeughülle visualisiert. Bei den beiden abgeleiteten Varianten 1b-1 und 1b-2 werden hingegen alle generierten Kabinenkomponenten dargestellt. Die Zeit für die Ausführung des Blender-Modells erhöht sich dadurch auf 07:29 min bei Variante 1b-1 und auf 14:11 min bei Variante 1b-2. Bei den angegebenen Zeiten handelt es sich um Mittelwerte von zehn gemessenen Durchläufen (siehe Anhang A.6.4). Die nahezu Verdopplung der Prozesszeit für die 3D Visualisierung bei Variante 1b-2 ist darin begründet, dass bei der Systemgruppe der Gepäckablage XL detailliertere 3D Objekte verwendet werden. Die Platzierung der Objekte erfordert dadurch eine höhere Rechenkapazität und damit verbunden eine längere Modellierzeit. Dennoch ist die entwickelte Methode mit anderen Ansätzen für die Visualisierung von 3D Kabinenmodellen vergleichbar. Mit der KBE-Applikation FUGA (Fuselage Geometry Assembler) von Walther wird die Visualisierung einer A321neo vergleichbaren Flugzeugstruktur mit Kabine und Triebwerk in 447,9 (07:27 min) realisiert [Wal24]. Mit Pacelab ACE myCabin sind LOPAs der Kabine und 3D Visualisierungen ohne die Kabinensysteme in 15 min umsetzbar [PAC24].

Tabelle 8.8: Vergleich der Prozesszeiten mit den Literaturwerten anderer Kabinenauslegungsmethoden. Die Literaturangaben wurden [Wal24], [Fuc14] und [PAC24] entnommen.

| | *Pacelab ACE myCabin* | *Fuchte* | *Walther (FUGA)* | *Variante 1b-1, Variante 1b-2* |
|---|---|---|---|---|
| | *Literatur* | *Literatur* | *Literatur* | |
| **Bezeichnung** | **Wert** | **Wert** | **Wert** | **Wert** |
| 3D Visualisierung | 15 min | - | 7:27 min | 7-14 min |
| Auslegung Kabine | - | 10-20 s | 15,79 s | 18-19 s |

Die Prozesszeit des Matlab-Modells für die Auslegung der Kabinenkomponenten verändert sich nur gering und benötigt bei Variante 1b-1 18,07 s und bei Variante 1b-2 19,32 s. Im Vergleich zur Basiskonfiguration 1b sind die Prozesszeiten für die Ausführung der Matlab-Modelle etwas kürzer, da die Ansteuerung aus dem Cameo Systems Modeler heraus entfällt und einige Sekunden beansprucht. Insgesamt liegt die Auslegungszeit der Kabine inkl. der Berücksichtigung von Kunden- und Sicherheitsanforderungen ebenfalls in der Größenordnung anderer Methoden. Die regelbasierte Berechnung mit KBE-Methoden beträgt bei FUGA 15,79 s [Wal24]. Die Prozesszeit für die Berechnung der Kabine (Platzierung der Monumente, Sitze und Gepäckablagen)

und der Rumpfstruktur ohne die Betrachtung der Kabinensysteme beträgt bei Fuchte 10-20 s [Fuc14]. Mit PreSTO (Aircraft Preliminary Sizing Tool) der HAW Hamburg können ebenfalls Kabinenkonfigurationen und Rumpfberechnungen unter Berücksichtigung von Anforderungen aus der CS-25 durchgeführt und unmittelbar im Anschluss in einem 2D Querschnitt visualisiert werden [See08]. Weitere Kabinensysteme und detailliertere 3D Visualisierungen werden hierbei nicht berücksichtigt. Bei allen drei Methoden/ Tools liegt der Fokus der Modellierung auf der Geometrie und der Platzierung von Komponenten. Weiterführende Analysen, Simulationen oder Bewertungen sind nicht vorgesehen. Diese können jedoch mit dem in dieser Arbeit gezeigten Partialmodell durchgeführt werden. Dabei ermöglicht das Matlab-Modell bereits erste Analysen für z.B. die Berechnung des Passagierkomforts. Darüber hinaus lassen sich Modelle für weiterführende Simulationen, wie in Fallstudie I gezeigt, ankoppeln. Insgesamt ist die Ausführungszeit der Prozesskette für die Auslegung und Generierung eines hochauflösenden 3D Modells mit anderen Ansätzen vergleichbar.

Zusammenfassend wurde in der Fallstudie II gezeigt, wie Variationen eines Subsystems innerhalb eines Partialmodells durchgeführt werden können. Am Beispiel des Customizings der Flugzeugkabine wurde die Methode erprobt. Dabei wurden Änderungen innerhalb der Kabine durchgeführt, während die weiteren Subsysteme und der Bauraum unverändert bleiben. Durch die Aufteilung in mehrere Partialmodelle müssen bei der Auslegung neuer Kabinenkonfigurationen im Gegensatz zu einem 150% Einzelmodell nur die kabinenrelevanten Modelle ausgeführt werden. Die Auslegung der anderen Subsysteme mit deren Partialmodellen muss nicht wiederholt durchlaufen werden, da in diesen keine Änderungen vorgesehen sind. Somit lassen sich vertiefende Untersuchungen innerhalb eines Subsystems realisieren. Potentiell auftretende Änderungen auf Gesamtsystemebene durch die neuen Konfigurationen innerhalb des Partialmodells (z.B. Kabine) werden erst bei erneuter Ausführung der gesamten Prozesskette erkennbar.

#### Zwischenfazit

Die Customizing-Studie zeigt die Neuauslegung einer Kabinenkonfiguration für eine bereits festgelegte Flugzeugvariante. Die Beispiele symbolisieren Anpassungswünsche durch den Kunden im Rahmen des Customizings. Variiert wurden sowohl der Typ der Gepäckablage, die Anordnungsmöglichkeiten und Abmaße der Komponenten der Passagierservicefunktionen als auch die Sitzklassen. Die Auswirkungen innerhalb der Kabine auf die Platzierung weiterer Komponenten sind in den Abbildungen ersichtlich. Zudem können Bewertungskriterien berechnet werden, um die unterschiedlichen Varianten anschließend miteinander vergleichen zu können. Zudem lassen sich mit der Anbindung an das Blender-Modell durch die XML-Datei die Kabinenkonfigurationen ebenfalls als hoch detaillierte 3D Modelle visualisieren. Weiterhin können die generierten 3D-Dateien für zusätzliche Studien, z.B. als digitales Mock-up in der Virtuellen Realität, Verwendung finden. Kunden können dadurch ihre konzeptionellen Designs in einem 1:1 Maßstab erleben und interaktiv erkunden.

Die Methode ermöglicht eine schnelle Auslegung und Visualisierung verschiedenster Konfigurationen innerhalb der Kabine unter Berücksichtigung geltender Sicherheitsanforderungen. Zudem lassen sich spezifische Auswahlkriterien überprüfen. Durch die Bewertungskriterien für den Passagierkomfort und die Bauweise kann für den

Kunden eine Handlungsempfehlung für die Auswahl der geeignetsten Kabinenkonfiguration je nach Geschäftsmodell abgeleitet werden.

## 8.4 Kapitelfazit

In diesem Kapitel wurden zwei industrielle Fallstudien vorgestellt, um die in dieser Arbeit entwickelte Methode in der praktischen Anwendung zu evaluieren. In der ersten Fallstudie lag der Fokus auf der Untersuchung der partnerübergreifenden Kopplung von Wissensmodellen und der Beschreibung eines Gesamtsystems mit verknüpften Partialmodellen. Dazu wurden zwei Flugzeugvarianten mit der entwickelten Methode ausgelegt. Die Variante 1a betrachtet eine Standardkonfiguration, bei der die Energieversorgung der Kabine durch eine Hilfsturbine bereitgestellt wird. Bei der Variante 1b wird die Energieversorgung durch ein Wasserstoff-Brennstoffzellensystem ersetzt. Bei beiden Varianten wird die gleiche Standardkabinenausstattung mit einer Einklassenbestuhlung verwendet. Mit der Variante 1a können 180 Passagiere transportiert werden, während im Vergleich dazu bei Variante 1b durch die Bauraumreduzierung der Kabine, bedingt durch die Platzierung des Wasserstofftanks, nur 162 Passagiere Platz finden. Für eine Validierung der Ergebnisse fand ein Abgleich der Werte von Variante 1a mit den Literaturwerten einer A320 statt. Die Ergebnisse der Modelle stimmen für die Anwendung der Methode in der Fallstudie hinreichend genau überein. Die Prozesszeiten für die Auslegung einer Variante betragen zwischen 7 min und 13 min. Damit demonstriert die erste Fallstudie eine schnelle Gewinnung erster Erkenntnisse für Varianten auf Gesamtsystemebene mit dezentralen Wissensmodellen, um eine Bewertung vornehmen zu können. Zudem wurde demonstriert, dass die Auslegung eines Gesamtsystems in mehrere Subsystemmodelle aufgebrochen werden kann. Je nach Variante, können die Subsystemmodelle (Partialmodell) beliebig miteinander gekoppelt oder auch erweitert werden, solange die Schnittstellen klar definiert sind.

Im Rahmen des konzeptionellen Machbarkeitsnachweis ließ sich zeigen, wie verschiedene Partialmodelle miteinander über eine XML-Datei verlinkt werden können und damit ein system- und unternehmensübergreifender Austausch von Parametern realisiert werden kann. Darüber hinaus haben die verschiedenen Partialmodelle unterschiedliche Detailtiefen und Umfänge. Zudem bestehen diese aus heterogenen Modellierungsumgebungen, sowohl open-source als auch kommerziell, die miteinander gekoppelt werden konnten. Das APU-Partialmodell liegt als reiner Parameterdatensatz vor, bei dem das System bereits von externen Partnern im Detail ausgelegt wurde. Das Partialmodell Wasserstofftanksystem umfasst eine Architekturmodellierung im Cameo Systems Modeler zusammen mit einem Python-Interface für die geometrische Tankmodellierung und wurde ebenfalls von einem externen Partner bereitgestellt. Für die Auslegung des Brennstoffzellensystem besteht das Partialmodell aus einem Architekturmodell im CSM und einem Simulationsmodell in Matlab/Simulink. Das Partialmodell der Kabine inkl. Subsysteme besteht aus der Architekturmodellierung und Anforderungsüberprüfung im CSM und der Geometriemodellierung in Matlab. Im Anschluss kann das Gesamtsystem als 3D Geometriemodell in Blender visualisiert werden.

In der zweiten Fallstudie wurde das Customizing in der Kabine untersucht. Der Anwendungsfall fokussiert die Systembaugruppe der Gepäckablage und der Passagier-

servicefunktionen. Als Ausgangskonfiguration wurde die Variante 1b aus dem ersten Fall gewählt. Von dieser wurden zwei neue Kabinenkonfigurationen abgeleitet. In der ersten wurde eine größere Gepäckablage gewählt, die mehr Stauraum für das Gepäck der Passagiere bietet. Bei der zweiten Konfiguration wurde die größte Gepäckablage gewählt sowie eine Zweiklassenbestuhlung innerhalb der Kabine bestehend aus Business- und Economyklasse untersucht. Ein Vergleich der Varianten erfolgte über zwei Kriterien. Diese sind der Passagierkomfort (Erreichbarkeit) und die Modulbauweise der Systemgruppe Gepäckablage. Im Vergleich zur Standardkonfiguration bietet die Variante 1b-2 mit der XL Gepäckablage nahezu identische Werte für den Komfort des Passagiers wie bei der Basiskonfiguration. Gleichzeitig bietet die extra große Gepäckablage die Möglichkeit der Modulbauweise und damit potentielle Kosten- sowie Zeiteinsparungen bei der Montage innerhalb des Flugzeugs. Die Studie zeigt, dass Varianten beliebig erweitert und modifiziert werden können, ohne dass manuell ein neues Modell der Architektur aufgebaut werden muss. Die entwickelte Methode ist damit sowohl auf Gesamtsystemebene modular als auch in den Modellen selbst erweiterbar. Im Rahmen des Customizings muss dadurch nicht die gesamte Architektur neu berechnet, sondern nur einzelne Modelle innerhalb eines Partialmodells ausgeführt werden.

Zusammengefasst ermöglicht die in dieser Arbeit entwickelte Methode für das Customizing eine schnelle erste Auslegung und Visualisierung der Ergebnisse. Die Durchlaufzeit einer Kabinenneuauslegung beträgt 18-19 s. Die Visualisierung aller Kabinenkomponenten und -systeme in einem 3D CAD-Modell dauert je nach Detailgrad zwischen 7 min und 14 min. Neue Kabinenkonfigurationen können nach den Wünschen des Kunden damit schnell überprüft und eine Vielzahl an Varianten durchgerechnet und bewertet werden. Dadurch lassen sich Handlungsempfehlungen für vertiefende Untersuchungen ausgewählter Kabinenkonfigurationen ableiten und weiter gezielter analysieren. Eine Diskussion der Ergebnisse und der entwickelten Methodik im Hinblick auf die Forschungsfrage erfolgt im nächsten Kapitel.

# 9. Diskussion und Bewertung der Ergebnisse

Das folgende Kapitel widmet sich der Überprüfung des in Abschnitt 3.3 gestellten Forschungsziels inkl. des Subziels und den Anforderungen an eine modulare sowie kollaborative Methode für die Variantenmodellierung unter Berücksichtigung der praktischen Anwendbarkeit im industriellen Kontext. Hierbei werden die Erkenntnisse der Fallstudien zusammengefasst und im Kontext des Stands der Wissenschaft beleuchtet. Die vorliegende Analyse dient nicht nur der Beantwortung der Forschungsfrage, sondern auch der Generierung neuer Erkenntnisse und der Identifikation zukünftiger Forschungsvorhaben. Damit konzentriert sich das Kapitel auf die Diskussion und Reflexion der durchgeführten Arbeiten und schließt die Deskriptive Studie II der DRM ab.

## 9.1 Reflexion der Anforderungen

Zuerst wird überprüft, ob die an die zu entwickelte Methode gestellten Anforderungen erfüllt sind. Diese Überprüfung erfolgt anhand der Ergebnisse der Fallstudien. Dies dient als Grundlage für die Beantwortung der eingangs formulierten Forschungsfrage.

#### A.1 Induzierte Modellkomplexität

Durch die Auslegung von Subsystemen in Partialmodellen und die Entkopplung der Variantenmodellierung auf mehrere Einzelmodelle konnte die Anforderung an eine Reduzierung der internen Modellkomplexität erfüllt werden. Dabei erzeugt ein komplexes Subsystem weiterhin ein komplexes Modell. Allerdings ist der Modellumfang durch die weniger darzustellenden Elemente und Knoten sowie unterstützenden Elemente im SysML-Modell deutlich geringer und kompakter als bei den 150% Einzelmodellen. Dadurch wird das Modell des Gesamtsystems weniger komplex und besser handhabbar. Folglich sind die Modelle in ihrer Modellgröße kleiner und können durch den Menschen einfacher interpretiert und bearbeitet werden. Die Anforderung **A.1** ist durch die entwickelte Methode vollständig erfüllt.

#### A.2 Erweiterbarkeit der Modelle

Da im Rahmen der Methode die Modellierung der Subsysteme in separaten Partialmodellen erfolgt und das Gesamtsystem erst durch die Kopplung dieser entsteht, wird

die Erweiterung beliebig vieler neuer Subsysteme in Partialmodellen sowie die Untersuchung neuer Varianten ermöglicht. Dafür werden für neue Subsysteme, die durch die Weiterentwicklung von Technologien oder Kundenbedürfnisse entstanden sind, weitere Partialmodelle aufgebaut. Diese ersetzten entweder bereits vorhandene Partialmodelle oder erweitern die Funktionalität des Gesamtsystems. Aufgrund der Entkopplung wird zusätzlich eine einfachere Durchführung von nachträglichen Anpassungen oder Ergänzungen innerhalb der Partialmodelle ermöglicht. Darüber hinaus sind die heterogenen Modellen ebenfalls intern modular aufgebaut, wodurch neue Subsystemkonfigurationen oder neue Komponenten nachträglich integriert werden können. Die Variantenmodellierung wird dadurch modularer und flexibler. Die föderierten Partialmodelle erfüllen vollständig die Anforderung **A.2**.

### A.3 Wiederverwendbarkeit der Modelle

Der entwickelte Ansatz ermöglicht die Wiederverwendbarkeit und Übertragung von vorhandenen Partialmodellen bei der Auslegung neuer Gesamtsystemvarianten durch die Aufgliederung bei den Subsystemen. Einige der Subsysteme finden in mehreren Varianten Anwendung und müssen nicht neu entwickelt werden. Hierdurch können die Partialmodelle inklusive der bereits geleisteten Entwicklungsarbeit durch die Experten bei neuen Untersuchungen weiterhin eingesetzt und bei Bedarf angeschlossen werden. Zudem ermöglicht die Wiederverwendung eine Verbesserung des Aufwand-Nutzen-Verhältnisses und erfüllt somit die Anforderungen an eine praxistaugliche Lösung. Die Anforderung **A.3** ist vollständig erfüllt.

### A.4 Multidisziplinäre Zusammenarbeit

Eine umfassende Modellierung des Gesamtsystems und die Betrachtung unterschiedlicher Aspekte eines Systems sind nur möglich, wenn verschiedene Modelltypen konsequent miteinander verbunden werden können. Durch die Vielfalt an Entwicklungsumgebungen jeder Fachdisziplin müssen daher sowohl Modelle kommerzieller Software als auch Modelle, die mit open-source Software entwickelt wurden, miteinander gekoppelt werden können. In den beiden Fallstudien I und II aus Kapitel 8 ließ sich zeigen, dass die Partialmodelle Tool-offen sind und die interdisziplinäre Zusammenarbeit mit mehreren Partnern in einem industriellen Kontext möglich ist. In diesem Zusammenhang wurde demonstriert, dass einerseits innerhalb eines Partialmodells analytische Modelle und deskriptive Modelle miteinander gekoppelt werden können. Andererseits wurde aufgezeigt, dass ein Partialmodell auch nur aus einem einzigen Modell bestehen kann. Die Anwendung einer XML-Datei als Adapter ermöglicht die Anbindung vieler Entwicklungsumgebungen, da diese Schnittstelle weit verbreitet ist und Parameter von vielen Modellen gelesen und geschrieben werden können. Durch die einfache Zugänglichkeit wird der Aufbau von Schnittstellen für Modelle und damit der Anschluss weiterer heterogener Partialmodelle an das Gesamtsystem für alle Experten erleichtert.

**A.4.1 Einfluss auf Modellgröße** Die Kopplungsart mittels XML hat nur einen geringen Einfluss auf die Modellgröße. Lediglich für das Ein- und Auslesen der Daten werden Modellelemente benötigt. Im Gegensatz zu kostenpflichtigen oder Tool-

spezifischen Lösungen zur Kopplung von mehreren Modellen, besonders in Modellierungsumgebungen für die SysML, werden keine weiteren indirekt abhängigen Modellelemente aus den anderen Partialmodellen hinzugefügt. Dadurch wird das Modellverständnis gefördert als auch die Anpassungsfähigkeit infolge von Modelländerungen verbessert.

**A.4.2 Modellpflege** Durch die klare Struktur der Modelle nach dem EVA-Prinzip werden die Modellelemente für die Kopplung klar von den Modellelementen für die Systemmodellierung getrennt und sind direkt erkennbar. Dies führt zu einem besseren Modellverständnis als auch zu einer verbesserten Pflege der Modelle. Änderungen an den Inputs und Outputs können einfacher und gezielter durchgeführt werden, ohne das Modell zur Beschreibung des Subsystems zu kompromittieren. Somit ist die Anforderung **A.4** erfüllt.

#### A.5 Unternehmensübergreifende Zusammenarbeit

Der entwickelte Lösungsansatz föderierter Partialmodelle realisiert die kollaborative Entwicklung von Varianten mit mehreren externen Partnern. Dadurch können eine Vielzahl an Experten gemeinsam Systemvarianten unter Wahrung von IP-Rechten und Berücksichtigung der IT-bezogenen Einschränkungen untersuchen – wodurch die Methodik die notwendige Akzeptanz in der Praxis erlangt.

**A.5.1 Lizenzabhängigkeit** Die Methode verwendet einen XML-Adapter. Die Kopplung der Partialmodelle zwischen unterschiedlichen Experten ist daher von einer Lizenz unabhängig. Dieses offene und standardisierte Schnittstellenformat ermöglicht eine universelle Integration, sodass jedes Partialmodell unabhängig von den intern verwendeten Modellierungsumgebungen eingebunden werden kann. Folglich können präferierte und spezialisierte Modellierungsumgebungen bei den einzelnen Experten weiter verwendet werden.

**A.5.2 Automatisierung** Mit Hilfe des XML-Adapters wird ein spezifischer Datenaustausch zwischen den disziplinären Tools ermöglicht, wodurch eine beschleunigte multidisziplinäre Analyse des Gesamtmodells realisiert werden kann. Dabei müssen die Ergebnisse der Partialmodelle nicht manuell zwischen den Experten ausgetauscht und eingelesen werden. Stattdessen ermöglichen die meisten Modellierungsumgebungen ein automatisiertes Ein- und Auslesen der Daten, sodass eine effiziente Zusammenarbeit möglich ist. Folglich ist die Anforderung **A.5** vollständig erfüllt.

In der Gesamtheit erfüllt somit die vorgestellte Methode der föderierten Partialmodelle umfänglich alle Anforderungen aus Kapitelabschnitt 3.3.

## 9.2 Reflexion der Forschungsfrage

Dieser Abschnitt diskutiert, inwieweit die Forschungsfrage unter Berücksichtigung der ergänzenden Forschungsfrage beantwortet werden konnte. Dafür wird die Anforderungserfüllung aus Abschnitt 9.1 herangezogen.

**Forschungsfrage** Wie kann methodisch eine Modularität des Gesamtsystems ermöglicht werden, damit Varianten eines komplexen technischen Systems abgebildet und für die multidisziplinäre Analyse des Systemverhaltens kollaborativ modelliert werden können?

**Ergänzende Forschungsfrage** Wie kann eine praxisorientierte Methodik zur kollaborativen Entwicklung von Varianten gestaltet werden, die im industriellen Kontext anwendbar ist und Komplexität sowie Aufwand-Nutzen-Verhältnis angemessen berücksichtigt?

Die Forschungsfrage dieser Arbeit kann dahingehend beantwortet werden, dass mit Hilfe föderierter Partialmodelle die genannten Herausforderungen an die Modularität des Gesamtsystems adressiert werden und ein kollaborativer Variantenentwurf sowie die Untersuchung des Customizings im Flugzeugvorentwurf möglich sind. Dabei wird der Ansatz zur Dekomposition eines Gesamtsystems auf die Modellebene übertragen, sodass ein Gesamtsystem nicht durch ein Einzelmodell, sondern durch mehrere Partialmodelle abgebildet wird. Jedes Subsystem wird in einem separaten Modell ausgelegt. Das Partialmodell selber kann aus einem oder mehreren (heterogenen) Modellen bestehen. Jedes Partialmodell wird als Black-Box verstanden, bei dem nur die Schnittstellen betrachtet werden. Dies vereinfacht die Anbindung interner und externer Partner, da die interne Modellstruktur nicht mit anderen Partnern geteilt werden muss. Somit ist auch eine Anbindung von Partialmodellen ohne SysML-Modell oder mit SysML-Modellen anderer Entwicklungsumgebungen als dem Cameo Systems Modeler möglich. Dadurch ermöglicht der Lösungsansatz eine multidisziplinäre Kopplung von Modellen.

Eine Auslegung des Gesamtsystems ergibt sich durch die Kopplung der Partialmodelle. Die Kopplung erfolgt durch einen XML-Adapter. Über diesen werden Parameter über die Modellgrenzen hinweg ausgetauscht. Je nach Zusammenschluss unterschiedlicher Partialmodelle entsteht eine Variante des Gesamtsystems. Alle Variationsmöglichkeiten eines Gesamtsystems müssen daher nicht zu Beginn definiert sein. Varianten eines Flugzeugs können laufend erweitert werden. Hierfür werden neue Partialmodelle generiert und anschließend entweder mit einem Partialmodell des gleichen Subsystemtyps ausgetauscht oder als Ergänzung angekoppelt. Die Integration neuer Partialmodelle von alternativen Subsystemen durch z.B. eine neue Technologie ist damit vereinfacht. Die Methode weist eine hohe Modularität in der Struktur der Entwurfsmodelle und Algorithmen innerhalb der heterogenen Modelle sowie auf der Gesamtsystemebene durch die Austauschbarkeit der Partialmodelle auf. Insgesamt erleichtert der modulare Ansatz das Konfigurationsmanagement und ermöglicht eine effizientere Verteilung der Modellierung in großen Projektteams. In den Fallstudien wurde zudem gezeigt, dass die Prozessschritte zur Variantenauslegung schnell durchführbar sind. Eine Skalierung der in dieser Arbeit entwickelten Methode auf die gesamte Auslegung eines Flugzeugs mit allen Fachdisziplinen ist damit umsetzbar. Zudem können mit diesem Ansatz auch größere Datenmengen ohne Leistungseinbuße (Entwurfszeit, Computerperformance) verarbeitet werden. Neue Varianten auf Gesamtsystemebene können dadurch schnell gewonnen und ein großer Entwurfsraum untersucht werden.

Die interne Struktur der Modelle erfolgt nach dem EVA-Prinzip (Eingabe-Verarbeitung-Ausgabe), um die Anforderung an eine einheitliche Modellstruktur zu erfüllen. Hierbei wird nur die Ordnerstruktur und nicht die Modellierung selbst vorgegeben. Die Modellierung der Systeme in den Partialmodellen erfolgt individuell nach den Modellierungsstilen der Systemexperten. Dadurch sind die Experten der Partialmodelle selbstständig verantwortlich für die Konsistenz ihrer Modelle. Dies erhöht jedoch die Zufriedenheit der Experten, da jeder seine favorisierte Modellierungsmethode verwenden kann. Zudem zeigte sich in der praktischen Umsetzung, dass dies ebenfalls einen Vorteil bei der Nachbesetzung von Experten liefert. Durch die EVA-Struktur können fachfremde Experten einen schnellen Einstieg in den Aufbau des Modells erhalten und leichter in die Modellierung einsteigen.

Ein Diskussionspunkt stellt die Kopplung der Partialmodelle dar. Durch die Kopplung können die Wechselwirkungen zwischen den Subsystemen und die Auswirkungen im Gesamtkontext untersucht werden. Dafür liest das Partialmodell zuerst die Inputparameter ein und startet dann die Auslegung des Subsystems. Bei Subsystemänderungen in einem anderen Partialmodell sind die Auswirkungen auf die Subsysteme in den weiteren Partialmodellen allerdings erst nach erneutem Import der Parameter aus der Adapterdatei sichtbar. Eine Zuordnung und Identifizierung dieser Parameter ist für eine globale Konsistenz durch die Verwendung von eindeutigen Objekt-IDs gegeben. Dennoch muss beim Aus- und Einlesen der Parameter auf die genaue Bezeichnung der Einträge geachtet werden. Besonders beim Einleseprozess wird nach spezifischen Bezeichnungen (z.B. Komponententyp) in der XML-Datei gesucht. Verwendet ein Partialmodell eine abweichende Bezeichnung, können im Einleseprozess anderer Partialmodelle die Einträge nicht gefunden und ausgelesen werden. Daher muss in der Zusammenarbeit in einem Team vorher eine einheitliche Begriffsdefinition festgelegt werden. Fortlaufende Änderungen an den Austauschparametern müssen zudem global zwischen den angeschlossenen Partnern kommuniziert werden, damit diese ggf. manuell in den Partialmodellen angepasst werden können. Eine automatisierte Erkennung von Änderungen bzw. Zuweisung der Parameter unabhängig von der Bezeichnung ist bislang nicht möglich.

In der industriellen Anwendung der Methode zeigte sich, dass für die Umsetzung in der Anfangsphase Entwicklungszeit in den strukturellen Aufbau der Modellierung und in die Definition der Systemgrenzen sowie Parameterbezeichnungen investiert werden musste. Gaspar et al. sehen ebenfalls als Herausforderungen bei modularisierten Modellen, dass diese höhere Vorabinvestitionen in die Modellierung benötigen und die Modelle aufgrund der Kopplung mit einer zusätzlichen Datenarchitektur ergänzt werden müssen [GKM23]. In weiterführenden Forschungsarbeiten könnte daher zum einen die Einführung einer Initialarchitektur untersucht werden. Mit dieser wird dem Nutzer im SysML-Modell ein bereits leeres Strukturkonstrukt mit Artefakten und Beziehungen zur Verfügung gestellt, das dann mit den entsprechenden Informationen des auszulegenden Systems gefüllt wird. Dabei dient die Initialarchitektur als Grundlage und kann beliebig (komplex) erweitert werden. Dieser Ansatz könnte dabei unterstützen, den Einstieg in die Modellierung der Systeme zu erleichtern und eine einheitliche Modellierungsstrategie aufzubauen.

Der im Rahmen dieser Forschungsarbeit entwickelte Lösungsansatz zur Kopplung interoperabler Modelle wurde im Projekt MIWa im Luftfahrtkontext hinsichtlich seiner

industriellen Anwendbarkeit erprobt, ist aber auf Produkte anderer Branchen übertragbar. Die Evaluation zeigte, dass der Ansatz eine Vereinfachung im Entwicklungsprozess ermöglicht. Dies zeigte sich insbesondere in der Möglichkeit zur verteilten Modellierung über Unternehmensgrenzen hinweg mithilfe der SysML im Cameo Systems Modeler sowie in der flexiblen Handhabung der Modelltiefe. Letztere erlaubte es Fachexperten weiterhin ihre domänenspezifischen, heterogenen Modellierungsumgebungen einzusetzen. Zudem führte die Aufteilung der Modellierung des Gesamtsystems in mehrere föderierte Partialmodelle dazu, dass die interne Komplexität der Modelle in einem bearbeitbaren Rahmen blieb und so ihre Handhabung deutlich verbessert wurde.

Ein zentrales Ergebnis stellt die Übertragbarkeit des föderierten Modellierungsprinzips nicht nur auf andere Modellierungsumgebungen dar. Zwar kamen in den durchgeführten Fallstudien ausschließlich Modelle im Cameo Systems Modeler zum Einsatz, jedoch ist der Lösungsansatz nicht auf dieses Tool beschränkt. Artefakte des Lösungsansatzes wie das EVA-Prinzip zur klaren Trennung von Schnittstellen- und Systemelementen sowie der Einsatz eines Adapters zur Kopplung heterogener Modelle sind allgemeingültig und unabhängig von der konkreten Modellierungsumgebung anwendbar. Die grundlegenden Paradigmen, insbesondere die Übertragung der Systemdekomposition auf die Modellierungsebene, um mit Hilfe modularisierter, interoperabler Modelle Systemvarianten auszulegen, bleiben erhalten. So ermöglichen auch Werkzeuge wie Capella (Python-Interface) [Git25, BSJ25] oder IBM Rhapsody (über die Java API for Customized Scripting) [San20] den Zugriff auf Parametereinträge innerhalb einer XML-Datei und deren Überführung ins SysML-Modell, wodurch die Zusammenarbeit mit vielen Partnern im Entwurf wiederum erleichtert wird. Die konkrete Ausgestaltung der Kopplung zwischen den Partialmodellen kann je nach gewählter Modellierungsumgebung variieren. Diese Variationen lassen sich jedoch durch entsprechende Anpassungen im Adapter einfach berücksichtigen. Im Hinblick auf zukünftige Entwicklungen, insbesondere im Kontext von Open-Source-Standards wie OpenJDK in Verbindung mit SysML v2, ergeben sich weiterführende Potenziale für die Erweiterung und Standardisierung interoperabler Modellkopplungen.

Somit sind in weiteren Untersuchungen Methoden für den Zugriff auf die SysML-Modellelemente und deren Beziehungen zu prüfen. Im Gegensatz zu anderen grafischen Programmiersprachen ist der Zugriff von extern auf die Verbindungen zwischen den Modellelementen limitiert. Die Rückverfolgbarkeit z.B. zwischen Anforderungen und Attributen instanziierter Objekte existiert nur innerhalb des Modells im Cameo Systems Modeler. Ein möglicher Ansatz ist der Zugriff auf das UML-Modell und die Notationselemente über die Open Java API. Der zugehörige Leitfaden stellt Anleitungen zur Implementierung benutzerdefinierter Plugins durch Pakete bereit, mit denen auf die Kernklassen der SysML-Anwendung[15] sowie auf die Modellelemente und deren Beziehungen[16] zugegriffen werden kann [No 23]. Mit diesen Paketen können dann Makros für das Auslesen von Daten oder der gesamten Modellarchitektur sowie die Erzeugung neuer Elemente geschrieben werden. Damit ergeben sich weitere methodische Möglichkeiten, die Kopplung der Partialmodelle zu realisieren.
Einen weiteren Ausblick gibt die Einführung der SysML v2. Zwei vielversprechende Maßnahmen sind die Interoperabilität und die integrierte Modellierung von Varianten. Mit der SysML v2 soll die Fähigkeit zum Austausch und zur Transformation von

[15] Package: com.nomagic.magicdraw.core.Application
[16] Package: com.nomagic.magicdraw.openapi.uml.SessionManager

Daten mit anderen Modellen gestärkt werden [Wei24]. Die standardisierte API und Tool-unabhängige Schnittstelle ermöglicht den Zugriff auf das SysML-Modell aus jeder anderen technischen Anwendung [Wei24]. Zusätzlich wird mit sogenannten Supersets die Modellierung der Variabilität eines Systems nach ISO/IEC 26580 ermöglicht [Wei24]. Dennoch entstehen dadurch wieder 150% Modelle. Die Anwendung der hier entwickelten Methode der föderierten Partialmodelle ist daher weiterhin ein potentieller Lösungsansatz, um die Variantenmodellierung in großen Systemen und im Hinblick auf die Übertragbarkeit auf die SysML v2 zu untersuchen.

In diesem Kontext ist die Entwicklung eines Standards für die Modellbildung anzudenken. Einerseits, kann ein XML-Datei-Standard als Adapter zwischen den Partialmodellen für die Modellbildung aufgebaut werden. Einige Ansätze in der Luftfahrt, wie das Datenaustauschformat Common Parametric Aircraft Configuration Schema (CPACS) [AMJN20], untersuchen dies bereits. Die Ergebnisse dieser Arbeit können als Grundlage für den Aufbau eines standardisierten Adapters dienen, der so ausgestaltet ist, dass er eine strukturierte Datenorganisation mit einheitlichem Vokabular und konsistenten Benennungen gewährleistet, alle anzukoppelnden Modelle auflistet, eindeutige Objekt-IDs zur globalen Konsistenz bereitstellt und eine Versionierung für Varianten ermöglicht.
Andererseits können die in dieser Arbeit vorgestellten Modellelemente mit Opaque Aktion zum Auslesen von Parametern und deren Export in den Adapter ebenfalls eine Grundlage zur Schnittstellenstandardisierung darstellen. Aufbauend auf den Ergebnissen dieser Arbeit könnten zukünftig standardisierte Modellelemente zum Export von Parametern als Erweiterung werkzeugseitig in Autorensysteme und Modellierungsumgebungen integriert werden. In dieser Arbeit wurden zwar die Datenaustausch-Modellelemente für die SysML im Cameo Systems Modeler entwickelt, dennoch ist der zugrundeliegende Ansatz auf weitere Modellwerkzeuge übertragbar. Weiterhin sind diese Datenaustausch-Modellelemente losgelöst von den Elementen zur Modellierung eines technischen Systems. Somit können die zu standardisierenden Modellelemente für den Datenaustausch unabhängig von bereits bestehenden Elementen unterschiedlichster Modellierungsansätze entwickelt und anschließend mit diesen kombiniert werden.

Darüber hinaus sind durch die Verwendung von föderierten Partialmodelle die Experten nicht an eine spezifischen Modelltypen und damit an eine Modellierungsmethode innerhalb der Modelle gebunden. Das erarbeitete Lösungskonzept ist grundsätzlich auf andere Forschungs- und Entwicklungsfelder übertragbar. Der verwendete Kopplungsansatz sowie die Adapter ermöglichen eine flexible Verbindung unterschiedlichster fachlicher Modelle, wobei diese nicht nur auf funktionale SysML-Modelle beschränkt sind. Vielmehr können sie beispielsweise auch mit physikalischen Modellen, wie z.B. FEM und CFD, gekoppelt werden, um das Gesamtsystemverhalten noch detaillierter zu analysieren, was letztlich zu effizienteren Entwicklungsprozessen und innovativen Lösungen führen kann.

# 10. Schlussbetrachtung

Im Rahmen der vorliegenden Arbeit wurde eine kollaborative Methodik als Unterstützung für die Auslegung von Systemvarianten mit föderierten Partialmodellen entwickelt und im Rahmen des Flugzeugvorentwurfs im industriellen Kontext und partnerübergreifend umgesetzt. Der nachfolgende Abschnitt 10.1 fasst die wesentlichen Aspekte der Arbeit zusammen. Abschnitt 10.2 beendet die Arbeit mit einem Ausblick auf weiterführende Forschungsfragen, die sich durch die Erkenntnisse der vorliegenden Arbeit ergeben.

## 10.1 Zusammenfassung

Modellbasierte Ansätze aus dem Systems Engineering unterstützen die Produktentwicklung, indem sie Methoden für die Untersuchung von Produktvarianten bereitstellen. Je nach Fachdisziplin und Entwurfsstadium kommen im Entwurfsprozess unterschiedlichste Detaillierungsgrade und Tools zum Einsatz, um disziplinspezifische Bewertungen des Produktentwurfs zu ermöglichen. Dies führt zu einer Vielzahl unterschiedlicher Modellierungsansätzen und resultiert in einer vorherrschenden, stark heterogenen Methoden- und Modelllandschaft. Für eine ganzheitliche Betrachtung einer Gesamtsystemkonfiguration und für die Überprüfung von Auswirkungen über Subsystemgrenzen hinweg ist eine Kopplung der Modelle erforderlich, was jedoch aufgrund fehlender Interoperabilität und heterogener Schnittstellen eine Herausforderung darstellt.

Gleichzeitig sieht sich die Industrie zunehmend vor der Herausforderung, eine wachsende Anzahl an Produktvarianten anzubieten und ihre Produktentwicklung in Richtung Massenanpassung zu verschieben [Wei16]. Produktvarianten spielen in der Luftfahrt eine wichtige Rolle: Sie ermöglichen sowohl die Erfüllung individueller Kundenanforderungen (Customizing) als auch die Untersuchung von Entwurfsräumen zur Identifikation geeigneter Systemarchitekturen in einer frühen Phase der Entwicklung.

Jedoch zeigen sich bei der Untersuchung von Produktvarianten und in der Zusammenarbeit mit vielen (externen) Partnern Grenzen. Nach dem Stand der Wissenschaft werden zur Variantenmodellierung bestehende methodische Ansätze genutzt, bei denen Einzelmodelle aufgebaut werden, die insbesondere bei großen Systemen eine starke interne Modellkomplexität zur Folge haben und nur mit erhöhtem Aufwand nachträglich erweiterbar sind. Weiterhin werden im industriellen Kontext überwiegend kommerzielle Softwarelösungen eingesetzt, die im Vergleich zu open-source Lösungen in ihrer Flexibilität, Modularität und Erweiterbarkeit limitiert sind.

Insbesondere der Entwicklungsprozess von Flugzeugkabinen zeichnet sich durch eine Vielzahl beteiligter Experten aus. Rund 80% der Kabinenbauteile sind Zukaufteile, wodurch ein Großteil der Systemkomponenten extern modelliert, ausgelegt und optimiert wird. Der technologische Wandel sowie neue Forschungserkenntnisse erfordern zudem eine hohe Anpassungsmöglichkeit der verwendeten Toolkette, damit neue Modelle, Systeme oder Disziplinen fortlaufend integriert werden können.

Ausgehend von den zuvor beschriebenen Herausforderungen entstand der Bedarf, eine modulare Variantenmodellierung sowohl auf Gesamtsystemebene als auch im Customizing zu ermöglichen und zugleich die Zusammenarbeit mit vielen multidisziplinären Partnern zu unterstützen. Gleichzeitig besteht als Teilzielsetzung, eine praxisnahe Methodik für die kollaborative Variantenentwicklung zu schaffen, die Industrieanforderungen sowie das Aufwand-Nutzen-Verhältnis angemessen berücksichtigt.

Der Forschungsansatz dieser Arbeit verfolgt den Einsatz von föderierten Partialmodellen. Bei dieser Methode wird das Gesamtsystem in seine Subsysteme zerlegt und diese Dekomposition auf die Modellebene übertragen. Jedes Subsystem wird durch ein Partialmodell modelliert und ausgelegt. Diese Partialmodelle verfügen über definierte Ein- und Ausgänge zum Austausch von Informationen bzw. Modellparametern und werden als Black-Box betrachtet. Die Variantenbildung wird auf einer höheren Abstraktionsebene durchgeführt. Das Gesamtsystem wird erst durch den Zusammenschluss spezifischer Partialmodelle vollständig definiert. Dadurch wird die Modularität gestärkt, da Varianten erst durch die Kombination mehrerer Partialmodelle generiert werden, was eine erhebliche Flexibilität im Systemdesign ermöglicht und der induzierten Modellkomplexität entgegenwirkt. Die Kopplung der Partialmodelle erfolgt über einen Adapter (XML-Datei). Durch die Black-Box-Betrachtung können die Partialmodelle heterogen modelliert werden, da die Kopplung über eine standardisierte Schnittstelle erfolgt. Dies verbessert die Interoperabilität, da deskriptive (z.B. SysML) und analytische Modelle miteinander interagieren können.

Zur Überprüfung der Zielerreichung der erarbeiteten Methode als Unterstützung in der praktischen Modellierung von Produktvarianten, wurde im Rahmen der vorliegenden Arbeit ein Evaluationsvorgehen basierend auf zwei industriellen Fallstudien gewählt. Der Fokus lag auf der Kollaboration mit mehreren externen interdisziplinären Partnern und der Interoperabilität verschiedener Modelle, um den industriellen Kontext und die Zulieferketten-Problematik zu untersuchen. Dabei werden die Kopplungen sowohl innerhalb der Partialmodelle als auch zwischen den Partialmodellen auf Gesamtsystemebene betrachtet. In den beiden Fallstudien konnte einerseits der Einsatz des Schnittstellenformats XML als Adapter zwischen den Partialmodellen und andererseits die Anforderungserfüllung einer Tool-offenen Methode (open-source und kommerziell) überprüft werden. Darüber hinaus wurde in Fallstudie I die entwickelte Methode in der praktischen Anwendung zur Bildung von Varianten auf Gesamtsystemebene im industriellen Kontext erprobt. In Fallstudie II wurde hingegen die praktische Anwendung der Methode für das Customizing in der Flugzeugkabine untersucht. Als wesentliche Treiber für die Effizienzsteigerung des Auslegungsprozesses haben sich im Rahmen der Methodenentwicklung die Modularität sowohl innerhalb der heterogenen Modelle als auch auf Partialmodellebene und die Wiederverwendbarkeit der Partialmodelle herausgestellt. Anpassungen am Entwurfsprozess können dadurch vereinfacht vorgenommen sowie neue Subsysteme mit Partialmodellen ergänzt werden. Als Ergebnis konnten in den Fallstudien Erstuntersuchungen zweier

Flugzeugvarianten in 7 bis 13 min und neue Kabinenkonfigurationen mit einer hohen Detailtiefe in unter 15 min multidisziplinär ausgelegt und bewertet werden.

Zusammenfassend belegt die Arbeit, dass die Kopplung föderierter, disziplinspezifischer Partialmodelle über einen Adapter einen wirksamen Beitrag zur kollaborativen, multidisziplinären Auslegung von Kabinenvarianten leistet – insbesondere vor dem Hintergrund bisheriger Herausforderungen hinsichtlich mangelnder Interoperabilität und Modularität in der industriellen Praxis. Es zeigt sich, dass föderierte Partialmodelle den Flugzeugvorentwurf unterstützen, indem neue Systemkonfigurationen aufgrund der modularen Struktur laufend erweitert werden, um die Untersuchung neuartiger Varianten und deren Systemarchitekturen über die Zulieferkette und auf Gesamtsystembasis zu ermöglichen. Dezentrale Wissensmodelle schließen dadurch eine Lücke, die durch eine stark heterogene Modelllandschaft bedingt ist, indem sie die Zusammenarbeit mit vielen interdisziplinären (externen) Experten fördern. Die in dieser Arbeit im industriellen Kontext der Luftfahrt erprobte Methode ist darüber hinaus universell einsetzbar und auf komplexe Systeme anderer Industriezweige wie Schienenfahrzeuge oder Schiffe, in denen eine heterogene Modelllandschaft vorliegt, sowie deren Modellierungsumgebungen übertragbar.

## 10.2 Ausblick

Im Rahmen der Diskussion in Kapitel 9 wurde aufgezeigt, dass die in Abschnitt 3.3 definierte Forschungsfrage beantwortet und die erarbeiteten Anforderungen an einen Lösungsansatz umgesetzt werden konnten. Mit Hilfe der entwickelten Methode konnte eine Kopplung heterogener Modelle für die Zusammenarbeit mehrerer Experten und die Untersuchung von Varianten erreicht werden. Für eine vollständige digital durchgängige Prozesskette im Vorentwurf von Flugzeugen und im Customizing von Flugkabinen sind weitere Entwicklungen in verschiedenen Bereichen erforderlich. Diese Aspekte werden nachfolgend als potentielle weiterführende Forschungsfragen vorgestellt.

Ein nächster Schritt sieht Untersuchungen vor, um weitere Modelle für Analysen in Simulationsmodellen oder immersiven (VR) Umgebungen anzuschließen. Vor allem die Anbindung an immersive Modellumgebungen verspricht Vorteile für die Überprüfung weiterer Eigenschaften und qualitativer Kriterien im Rahmen des Customizings durch interaktive virtuelle Mock-ups. Darüber hinaus kann die Kommunikation in interdisziplinären Teams verbessert werden. Die grafischen Oberflächen, die bereits durch die Toolumgebungen bereitgestellt werden, sind teilweise sehr komplex und unübersichtlich. Eine Erweiterung der SysML-Struktur und Anbindung an beispielsweise eine VR-Umgebung kann das Systemverständnis nochmals verbessern und Systemzusammenhänge besser herausarbeiten. Grafische Übersichten verschaffen einen guten Überblick über ein System, ohne dass tiefgehende technische Kenntnisse benötigt werden [BSA+20].

Die erarbeitete Fallstudie I hat gezeigt, wie die bereits vorhandenen internen Schnittstellen des Cameo Systems Modeler für eine Anbindung an externe Modelle und den Austausch von Parametern genutzt werden können. Mit Einführung der SysML v2 soll die Interoperabilität der SysML-Modelle weiter gestärkt und der Austausch von Daten

mit anderen (heterogenen) Modellen verbessert werden [Wei24]. Über die standardisierte API und eine Tool-unabhängige Schnittstelle oder eigene Makros soll der Zugriff auf die Elemente des SysML-Modells und die textuelle Notation von extern gelingen [Wei24]. Darüber hinaus ist die Entwicklung eines Standards sowohl für den Adapter zwischen den Partialmodellen als auch für die Kopplungsfähigkeit der Modellierungswerkzeuge in der interdisziplinären Zusammenarbeit zu erwägen. Erste Ansätze, wie etwa das Datenaustauschformat CPACS [AMJN20], verdeutlichen bereits das Potenzial eines standardisierten Informationsaustauschs zwischen unterschiedlichen Fachdisziplinen und deren Modellen. Ergänzend dazu könnte auf Grundlage der in dieser Arbeit vorgestellten methodischen Umsetzung eine Systemerweiterung der Autorensysteme und Modellierungswerkzeuge entwickelt werden, bei der standardisierte Modellelemente zum Auslesen interner Modellparameter und deren Speicherung in einer XML-Datei werkzeugseitig bereitgestellt werden.

In den beiden erarbeiteten Fallstudien zeigte sich, dass zu Beginn der Modellierung neben der Definition der Schnittstellen, eine einheitliche Begriffsdefinition der Parameter erfolgen muss. Eine Abweichung von der vereinbarten Begriffsdefinition in einem der Modelle lässt eine eindeutige Identifizierung der Parameter nicht mehr zu. Alle Partner sind daher aufgefordert, die global definierten Namen der Parameter in ihren Modellen präzise einzuhalten sowie eine regelmäßige Versionskontrolle der Namensgebung zu betreiben, um eine Zuordbarkeit zu gewährleisten. Ein möglicher Lösungsansatz ist die direkte sowie namensunabhängige Verknüpfung der Attribute und Notationselemente zwischen den heterogenen Modellen sowie die Verwendung eines Glossars.

In einer stark heterogenen Modelllandschaft hat eine wiederkehrende, einheitliche Struktur positive Auswirkungen auf den Umgang mit interner Komplexität. Mit der Einführung der EVA-Struktur für eine klare interne Gliederung der Modelle wurde ein erster Grundstein gelegt. Die Modellierung der Systeme selbst erfolgt individuell durch die Systemexperten. Die Einführung einer Initialarchitektur für SysML-Modelle könnte allerdings den Aufbau einer einheitlichen Modelllierungsstrategie unterstützen und den Einstieg in die Modellierung eines Subsystems erleichtern. In Anlehnung an das Startmodell für die kollaborative CAD-Modellierung von Tecklenburg [Tec10] kann eine Initialarchitektur in Verbindung mit der EVA-Struktur einen durchgängigen Informationsfluss und das systematische Vorgehen bei der Modellierung fördern. Dabei gibt es jedoch nicht den einen Standard, sondern die Initialarchitektur variiert je nach Einsatzzweck. Bei der Erstellung eines Partialmodells wird den Experten bedarfsspezifisch ein inhaltlich leeres SysML-Strukturkonstrukt mit Modellartefakten und Beziehungen zur Verfügung gestellt. Darauf basierend könnten die Systemexperten zukünftig ihre Subsysteme modellieren und die Initialarchitektur beliebig (komplex) erweitern.

# Literaturverzeichnis


[AB98] K. J. Abshire and M. K. Barron. Virtual maintenance real-world applications within virtual environments. In *Annual Reliability and Maintainability Symposium. 1998 Proceedings. International Symposium on Product Quality and Integrity*, pages 132–137. IEEE, 1998.

[ABS+19] A. Albers, N. Bursac, H. Scherer, C. Birk, J. Powelske, and S. Muschik. Model-based systems engineering in modular design. *Design Science*, 5(e17), 2019. Cambridge University Press.

[Abu11] J. Abulawi. Ansätze zur Beherrschung der Eigenkomplexität von parametrisch-assoziativen CAD-Modellen. In U. Freiherr von Lukas, E.-M. Mahnke, K. Haase, and S. Malo, editors, *Go-3D 2011, Computergraphik für die Praxis - Tagungsband der Konferenz Go-3D 2011*, pages 137–149. Fraunhofer Verlag, 2011. ISBN: 978-3839602751.

[Abu12] J. Abulawi. *Ansatz zur Beherrschung der Komplexität von vernetzten 3D-CAD-Modellen*. Dissertation, Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg, 2012.

[Abu22] J. Abulawi. Modellbasiertes Variantenmanagement, 2022. Vorlesungsunterlagen HAW Hamburg; Kurs Systems Engineering.

[Ack13] S. Ackert. Commercial Aspects of Aircraft Customization. *Aircraft Monitor*, 1, 2013.

[Air88] Airbus. Airbus A320 Weight and Balance Manual, 1988. Bericht.

[Air20] Airbus. A320 Aircraft Characteristics Airport and Maintenance Planning, 2020. Bericht, `https://www.airbus.com/sites/g/files/jlcbta136/files/2021-11/Airbus-Commercial-Aircraft-AC-A320.pdf`, Besucht am 16.01.2024.

[Air23a] Air Transport Association. Air Transport Association: ATA Chapter, 2023. `http://public.s1000d.org/Pages/Home.aspx`, Besucht am 13.04.2023.

[Air23b] Airbus. Design: A starting point for the innovations of tomorrow, 2023. `https://www.airbus.com/en/products-services/commercial-aircraft/the-life-cycle-of-an-aircraft/design`, Besucht am 13.04.2023.

[Air23c] Airbus. Production - Building aircraft on time and at top quality, 2023. https://www.airbus.com/en/products-services/commercial-aircraft/the-life-cycle-of-an-aircraft/production, Besucht am 18.10.2023.

[Air23d] Airlines for America. Who We Are, 2023. https://www.airlines.org/, Besucht am 13.04.2023.

[Air24] Airbus. A320 Aircraft Characteristics Airport and Maintenance Planning, 2024. Issue: Sep 30/85.

[Alt16] H.-H. Altfeld. *Commercial Aircraft Projects: Managing the Development of Highly Complex Products*. Routledge, 1. edition, 2016. ISBN: 978-1315572833.

[AM21] A. Aleksandraviciene and A. Morkevicius. *MagicGrid Book of Knowledge*. No Magic, Inc., 2. edition, 2021. ISBN: 978-6094545542.

[AMJN20] M. Alder, E. Moerland, J. Jepsen, and B. Nagel. Recent Advances in Establishing a Common Language for Aircraft Design with CPACS. *Aerospace Europe Conference 2020*, 2020.

[And06] R. Anderl. Virtuelle Produktentwicklung in der Automobilindustrie. *Informatisierung der Arbeit - Gesellschaft im Umbruch*, 2006. ISBN: 3894045477, Edition sigma, Berlin.

[ASA+22] S. S. Albouq, A. A. A. Sen, N. Almashf, M. Yamin, A. Alshanqiti, and N. M. Bahbouh. A Survey of Interoperability Challenges and Solutions for Dealing With Them in IoT Environment. *IEEE Access*, 10:36416–36428, 2022.

[ASP+23] S. Aney, M. Schestakow, L. Prikazchikova, B. Milow, D. Prikazchikov, J. Kaplunov, H. Voggenreiter, and A. Rege. Cellulose-Aerogele als multifunktionale und nachhaltige Alternativen für Flugzeugkabinenelemente. In *Deutscher Luft- und Raumfahrtkongress DLRK 2022*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2023.

[AVP+19] T. Amorim, A. Vogelsang, F. Pudlitz, P. Gersing, and J. Philipps. Strategies and Best Practices for Model-Based Systems Engineering Adoption in Embedded Systems Industry. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 203–212. IEEE, 2019.

[Bad19] A. Badiru. *Systems Engineering Models: Theory, Methods, and Applications*. CRC Press, 2019. ISBN: 978-1351266529.

[BBH+18] J. Buergin, F. Belkadi, C. Hupays, R. K. Gupta, F. Bitte, G. Lanza, and A. Bernard. A modular-based approach for Just-In-Time Specification of customer orders in the aircraft manufacturing industry. *CIRP Journal of Manufacturing Science and Technology*, 21:61–74, 2018.

[BBH+21] R. Brahmi, I. Belhadj, M. Hammadi, N. Aifaoui, and J.-Y. Choley. *A Design Approach of Mechanical Assemblies based on MBSE and CAD Models Interoperability*. Research Square, 2021. https://www.researchsquare.com/article/rs-407506/v1, Preprint, Besucht am 04.02.2024.

[BBJ+20] W. Böhm, M. Broy, M. Junker, A. Vogelsang, and S. Voss. *Praxisnahe Einfürung von Model-based Systems Engineering - Vorgehen und Lessons Learnt*, volume 1. fortiss gemeinnützige GmbH, Juni 2020. Whitepaper.

[BBL+16] L. Berardinelli, S. Biffl, A. Lüder, E. Mätzler, T. Mayerhofer, M. Wimmer, and S. Wolny. Cross-disciplinary engineering with AutomationML and SysML. *at - Automatisierungstechnik*, 64(4):253–269, 2016.

[BBS+19] D. Bilic, E. Brosse, A. Sadovykh, D. Truscan, H. Bruneliere, and U. Rysel. An Integrated Model-Based Tool Chain for Managing Variability in Complex System Design. In *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 288–293. IEEE, 2019. ISBN: 978-1-7281-5125-0.

[BC02] L. Blessing and A. Chakrabarti. DRM: A Design Research Methodology. In *Proceedings of Les Sciences de la Conception*, 2002. 15. - 16. März, INSA de Lyon, Lyon.

[BC09] L. T.M. Blessing and A. Chakrabarti. *DRM, a Design Research Methodology*. Springer-Verlag London, 2009. ISBN: 978-1848825864.

[BC23] J. H. Bussemaker and P. D. Ciampa. MBSE in Architecture Design Space Exploration. In Azad M. Madni, Norman Augustine, and Michael Sievers, editors, *Handbook of Model-Based Systems Engineering*, pages 1–41. Springer International Publishing, Cham, 2023. ISBN: 978-3030274863.

[BCS+20] D. Bilic, J. Carlson, D. Sundmark, W. Afzal, and P. Wallin. Detecting inconsistencies in annotated product line models. In R. E. Lopez-Herrejon, editor, *Proceedings of the 24th ACM Conference on Systems and Software Product Line - Volume A*, New York, NY, USA, 2020. ACM. ISBN: 978-1450375696.

[BCS+22] J. Bussemaker, P. D. Ciampa, J. Singh, M. Fioriti, C. Cabaleiro, Z. Wang, D. Peeters, P. Hansmann, P. Della Vecchia, and M. Mandorino. Collaborative Design of a Business Jet Family Using the AGILE 4.0 MBSE Environment. *AIAA Aviation 2022 Forum*, 2022.

[Bec20] F. Beckert. Automatisierte Kabinendarstellung der CPACS-Flugzeugdefinition in der Virtuellen Realität. Studienarbeit, Technische Universität Hamburg, 2020.

[BFK+18] B. Bender, J. Feldhusen, D. Krause, G. Beckmann, K. Paetzold, and A. Hövel. Grundlagen technischer Systeme und des methodischen

Vorgehens. In K.-H. Grote, B. Bender, and D. Göhlich, editors, *Dubbel*, pages 372–411. Springer Berlin, Heidelberg, 2018. ISBN: 978-3662548042.

[BGG+18] A.-R. Breje, R. Gyorödi, C. Gyorödi, D. Zmaranda, and G. Pecherle. Comparative Study of Data Sending Methods for XML and JSON Models. *International Journal of Advanced Computer Science and Applications*, 9(12), 2018.

[BHZ+22] J. M. Berges, G. Höpfner, Y. Zhang, J. Berroth, and G. Jacobs. Vernetzung von Simulationsmodellen und Model-Based Systems Engineering zur virtuellen Produktentwicklung. *NAFEMS DACH Regionalkonferenz*, 2022.

[Bil20] D. Bilic. *Managing Variability in SysML Models of Automotive Systems*. Dissertation, Mälardalen University Sweden, 2020.

[BKL90] A. Bernardi, C. Klauck, and R. Legleitner. Abschlußbericht des Arbeitspaketes PROD, 1990. Deutsches Forschungszentrum für Künstliche Intelligenz.

[BKRS16] T. Bauernhansl, J. Krüger, G. Reinhart, and G. Schuh. *WGP-Standpunkt Industrie 4.0*. Wissenschaftliche Gesellschaft für Produktionstechnik WGP e.V., 2016.

[BPM09] S. Baïna, H. Panetto, and G. Morel. New paradigms for a product oriented modelling: Case study for traceability. *Computers in Industry*, 60(3):172–183, 2009.

[Bra23] T. Braun. Airbus shows how manufacturers can benefit from product line engineering (PLE), 2023. `https://www.aerospacemanufacturinganddesign.com/news/airbus-shows-how-manufacturers-can-benefit-product-line-engineering-ple/`, Besucht am 13.10.2023.

[BSA+20] D. Bilic, D. Sundmark, W. Afzal, P. Wallin, A. Causevic, C. Amlinger, and D. Barkah. Towards a Model-Driven Product Line Engineering Process. In S. Jain, A. Gupta, D. Lo, D. Saha, and R. Sharma, editors, *Proceedings of the 13th Innovations in Software Engineering Conference on Formerly known as India Software Engineering Conference*, New York, USA, 2020. ACM. ISBN: 978-1450375948.

[BSJ+22] J. M. Berges, K. Spütz, G. Jacobs, J. Kowalski, T. Zerwas, J. Berroth, and C. Konrad. Automated Identification of Valid Model Networks Using Model-Based Systems Engineering. *Systems*, 10(6):250, 2022.

[BSJ25] M. Burgstaller, T. Schichl, and M. Jungwirth. Enhancing Manufacturing Efficiency Through Integration of MBSE and Capella in the Digital Thread. In A. Quesada-Arencibia, M. Affenzeller, and R. Moreno-Díaz, editors, *Computer Aided Systems Theory - EUROCAST 2024*, volume 15172 of *Lecture notes in computer science*, Cham, 2025. Springer.

[BSLK23] M. C. Berschik, T. Schumacher, F. N. Laukotka, and D. Krause. MBSE within the Engineering Design Community – An Exploratory Study. *Proceedings of the International Conference on Engineering Design (ICED23)*, 2023. Bordeaux, France.

[Bur16] N. Bursac. *Model Based Systems Engineering zur Unterstützung der Baukastenentwicklung im Kontext der Frühen Phase der Produktgenerationsentwicklung*. Dissertation, Karlsruher Institut für Technologie, 2016.

[Cam22] Cambashi. Global trends in aerospace manufacturing and software, 2022. Whitepaper.

[CDB+18] L. Cohen, P. Duboé, J. Buvat, D. Melton, A. Khadikar, and H. Shah. Augmented and Virtual Reality in Operations: A guide for investment, 2018. Capgemini Research Institute Report.

[CMRP18] M. Chamas, M. Markthaler, B. Rumpe, and K. Paetzold. Bewertungsmethodik zur Verbesserung der Testfallerstellung auf Basis von SysML. In D. Krause, K. Paetzold, and S. Wartzack, editors, *29. Design for X-Symposium*, 2018.

[CNL13] P. D. Ciampa, B. Nagel, and G. La Rocca. Preliminary design for flexible aircraft in a collaborative environment. *The International Conference of the European Aerospace Societies (CEAS)*, 2013. Linköping, Sweden.

[Cor06] C. Corsten. Interface and requirement traceability. In *SpaceOps 2006 Conference*, pages 5877–5889. AIAA, 2006. ISBN: 978-1624100512.

[Cor23] Cornelsen Verlag GmbH. Duden, 2023. `https://www.duden.de/woerterbuch`.

[CQN+22] R. Colletti, A. Qamar, S. Nuesch, W. Bailey, and C. Paredis. Variant modeling for multi-perspective, multi-fidelity systems simulation. In A. M. Madni, B. Boehm, D. Erwin, M. Moghaddam, M. Sievers, and M. Wheaton, editors, *Recent Trends and Advances in Model Based Systems Engineering*, pages 291–301, Cham, 2022. Springer International Publishing. ISBN: 978-3030820831.

[dA08] A. de Almeida Souza Neto. Antares DSM: Visualization and Optimization Depencies on Design Structures Matrices (Englisch Short Version), 2008. Salvador: Federal University of Bahia.

[DAG+21] R. Dumitrescu, H. Anacker, E.-M. Grote, R. Rasor, J. Tekaat, M. Meyer, J. Gausemeier, and S. Steglich. Erfolgspotentiale für die Zukunft des Engineeringstandorts Deutschland – Ein Beitrag zum Advanced Systems Engineering. *Vorausschau und Technologieplanung - 16. Symposium Vorausschau und Technologieplanung*, 2021.

[Dan03] W. Dangelmaier. *Produktion und Information: System und Modell*. Springer-Verlag Berlin Heidelberg, 2003. ISBN: 978-3642624483.

[Das19] Dassault Systèmes. Airbus und Dassault Systèmes bilden strategische Partnerschaft für die Verwirklichung der europäischen Luft- und Raumfahrtindustrie von morgen. online Pressemitteilung, 2019. https://www.3ds.com/de/newsroom/press-releases/airbus-and-dassault-systemes-embark-strategic-partnership-create-european-aerospace-industry-tomorrow, Besucht am 07.06.2023.

[Das23] Dassault Systèmes. Teamwork Cloud, 2023. https://www.3ds.com/products-services/catia/products/no-magic/teamwork-cloud/, Besucht am 14.06.2023.

[DDBL11] J. Daaboul, C. Da Cunha, A. Bernard, and F. Laroche. Design for mass customization: Product variety vs. Process variety. *CIRP Annals*, 60(1):169–174, 2011.

[DDD+21] C. David, S. Delbecq, S. Defoort, P. Schmollgruber, E. Benard, and V. Pommier-Budinger. From FAST to FAST-OAD: An open source framework for rapid Overall Aircraft Design. *IOP Conference Series: Materials Science and Engineering*, 1024(1):012062, 2021.

[Deu21a] Deutsches Zentrum für Luft- und Raumfahrt e.V. Auf dem Weg zu einer emissionsfreien Luftfahrt, 2021. https://www.dlr.de/de/aktuelles/nachrichten/2021/04/20211215_auf-dem-weg-zu-einer-emissionsfreien-luftfahrt, Besucht am 24.05.2023.

[Deu21b] Deutsches Zentrum für Luft- und Raumfahrt e.V. Auf dem Weg zu einer Emissionsfreien Luftfahrt: Luftfahrtstrategie des DLR zum European Green Deal, 2021.

[Deu23a] Deutsches Zentrum für Luft- und Raumfahrt e.V. DigECAT - Digitale Zwillinge als Forschungswerkzeug und Forschungsobjekt in der Luftfahrt, 2023. https://www.dlr.de/de/ki/forschung-transfer/projekte/digecat, Besucht am 24.05.2023.

[Deu23b] Deutsches Zentrum für Luft- und Raumfahrt e.V. Digitaler Zwilling, 2023. https://factory-of-the-future.dlr.de/pf/digital-twins/index.html, Besucht am 24.05.2023.

[DF84] M. Dorfman and R. F. Flynn. Arts - an automated requirements traceability system. *Journal of Systems and Software*, 4(1):418–428, 1984.

[Die23] Diehl Stiftung & Co. KG. Monuments and CRC's, 2023. https://www.diehl.com/aviation/de/portfolio/monuments-and-crcs/, Besucht am 31.05.2023.

[DIN13] DIN Deutsches Institut für Normung e.V. DIN CEN ISO/TR 7250-2:2011 + A1:2013 Wesentliche Maße des menschlichen Körpers für die technische Gestaltung – Teil 2: Anthropometrische Datenbanken einzelner nationaler Bevölkerungen, 2013. Beuth Verlag.

[DJT06] X. Du, J. Jiao, and M. M. Tseng. Understanding customer satisfaction in product customization. *The International Journal of Advanced Manufacturing Technology*, 31(3-4):396–406, 2006.

[DLA15] A. Dresch, D. P. Lacerda, and J. A. V. Antunes. Design science research. In *Design Science Research: A Method for Science and Technology Advancement*, pages 67–102, Cham, 2015. Springer International Publishing. ISBN: 978-3319073743.

[Dob01] L. Dobusch. *Komplexitätstheoretische Erklärungen der Struktur*. Seminararbeit, Johannes Kepler Universität Linz, 2001.

[Dra21] R. Drath. *AutomationML: A Practical Guide*. De Gruyter Textbook. De Gruyter, 2021. Foreword by Prof. Dr. Alexander Fay.

[DT18] R. Doering and F. Thielecke. Modellbasierte Entwicklung, Test und Bewertung von Last-Management-Algorithmen für den Betrieb einer energieautarken Flugzeug-Bordküche. *Deutscher Luft- und Raumfahrt Kongress, DLRK 2018*, 2018. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V.

[DZS+20] K. Dmitriev, S. A. Zafar, K. Schmiechen, Y. Lai, M. Saleab, P. Nagarajan, D. Dollinger, M. Hochstrasser, F. Holzapfel, and S. Myschik. A Lean and Highly-automated Model-Based Software Development Process Based on DO-178C/DO-331. In *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*, pages 1–10. IEEE, 2020. ISBN: 978-1728198255.

[EDA17] M. Eigner, T. Dickopf, and H. Apostolov. The Evolution of the V-Model: From VDI 2206 to a System Engineering Based Approach for Developing Cybertronic Systems. In J. Ríos, A. Bernard, A. Bouras, and S. Foufou, editors, *Product Lifecycle Management and the Industry of the Future*, volume 517 of *IFIP Advances in Information and Communication Technology*, pages 382–393. Springer International Publishing, Cham, 2017. ISBN: 978-3319729046.

[EDH20] M. Engelmann, D. Drust, and M. Hornung. Automated 3D cabin generation with PAXelerate and Blender using the CPACS file format. In *Deutscher Luft- und Raumfahrt Kongress 2020*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2020.

[EEGE20] T. Eickhoff, A. Eiden, J. C. Göbel, and M. Eigner. A Metadata Repository for Semantic Product Lifecycle Management. *Procedia CIRP*, 91:249–254, 2020.

[EETM12] W. ElMaraghy, H. ElMaraghy, T. Tomiyama, and L. Monostori. Complexity in engineering design and manufacturing. *CIRP Annals*, 61(2):793–814, 2012.

[EGZ12] M. Eigner, T. Gilz, and R. Zafirov. Proposal for Functional Product Decription as Part of a PLM Solution in Interdisciplinary Product Development. *International Design Conference - DESIGN 2012*, pages 1667–1676, 2012.

[EH19] M. Engelmann and M. Hornung. Boarding Process Assessment of the AVACON Research Baseline Aircraft. In *Deutscher Luft- und Raumfahrt Kongress 2019*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V, 2019.

[EH24] M. Engelmann and M. Hornung. Geometric optimization of hydrogen aircraft fuselages in preliminary design. *Proceedings of the 34th Congress of the International Council of the Aeronautical Sciences*, 2024.

[EKH20] M. Engelmann, T. Kleinheinz, and M. Hornung. Advanced passenger movement model depending on the aircraft cabin geometry. *Aerospace*, 7(12), 2020.

[EMS05] R. Eckert, W. Mansel, and G. Specht. STEP AP233 + Standard PDM = Systems Engineering PDM ? *2005 IEEE International Technology Management Conference (ICE)*, pages 405–412, 2005.

[ERZ14] M. Eigner, D. Roubanov, and R. Zafirov, editors. *Modellbasierte virtuelle Produktentwicklung*. Springer Vieweg, Berlin, Heidelberg, 2014. ISBN: 978-3662438169.

[Est08] J. A. Estefan. Survey of Model-Based Systems Engineering (MBSE) Methodologies. *INCOSE MBSE Initiative*, 2008.

[Eur21] European Union Aviation Safety Agency. Certification Specifications and Acceptable Means of Compliance for Large Aeroplanes (CS-25) - Amendment 27, 2021.

[EW23] J. A. Estefan and T. Weilkiens. MBSE Methodologies. In A. M. Madni, N. Augustine, and M. Sievers, editors, *Handbook of Model-Based Systems Engineering*, pages 47–85, Cham, 2023. Springer International Publishing. ISBN: 978-3030274863.

[FAF+23] M. Fuchs, J. Abulawi, C. Fuchs, D. Meier, B. Merkel, and T. Topal. Modellierungsstrategie zur Analyse des Einflusses von alternativen Antriebskonzepten auf Kabinensysteme. In *Deutscher Luft-und Raumfahrt Kongress (DLRK) 2023*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2023.

[FBBN22] M. Fuchs, F. Beckert, J. Biedermann, and B. Nagel. A collaborative knowledge-based method for the interactive development of cabin systems in virtual reality. *Computers in Industry*, 136, 2022.

[FBG+20] M. Fuchs, F. Beckert, M. P. Gopani, A. Gindorf, and B. Nagel. Virtuelle Realität im digitalen Designprozess von Flugzeugkabinensystemen. In *Deutscher Luft- und Raumfahrt Kongress 2020*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V, 2020.

[FD22] M. Friedrichs-Dachale. Establishment of a Digital Interface Between System Definition and System Analysis Models to Optimize the Aircraft Preliminary Sizing Process. Bachelorarbeit, Hochschule für Angewandte Wissenschaften Hamburg, 2022.

[FG21] J. Feldhusen and K.-H. Grote. Grundlagen technischer Systeme. In B. Bender and K. Gericke, editors, *Pahl/Beitz Konstruktionslehre: Methoden und Anwendung erfolgreicher Produktentwicklung*, pages 9–26, Berlin, Heidelberg, 2021. Springer Vieweg. 9. edition, ISBN: 978-3662573037.

[FGA+22] M. Fuchs, Y. Ghanjaoui, J. Abulawi, J. Biedermann, and B. Nagel. Enhancement of the virtual design platform for modeling a functional system architecture of complex cabin systems. *CEAS Aeronautical Journal*, 13(4):1101–1117, 2022.

[FGBN23] M. Fuchs, Y. Ghanjaoui, J. Biedermann, and B. Nagel. An Approach for Linking Heterogenous and Domain-Specific Models to Investigate Cabin System Variants. *INCOSE International Symposium*, 33(1):1418–1434, 2023.

[FHAZ23] B. Fröhler, J. Häßy, and M. Abu-Zurayk. Development of a medium/long-haul reference aircraft. *CEAS Aeronautical Journal*, 14:693–713, 2023.

[FHBF19] M. Fuchs, C. Hesse, J. Biedermann, and J. Fuchte. A multi-disciplinary design optimization of the passenger supply channel in the aircraft cabin. *MATEC Web Conf.*, 304:04009, 2019.

[FJN22] T. Firchau, M. Jetzschmann, and F. Nohka. Applying MBSE to Data Handling System Design of a Small Satellite Platform for Multiple Missions. *2022 IEEE Aerospace Conference*, 2022.

[FLC96] S. T. Frezza, S. P. Levitan, and P. K. Chrysanthis. Linking requirements and design data for automated functional evaluation. *Computers in Industry*, 30:13–25, 1996.

[FMR+22] K. Feichtinger, K. Meixner, F. Rinker, I. Koren, H. Eichelberger, T. Heinemann, J. Holtmann, M. Konersmann, J. Michael, E.-M. Neumann, J. Pfeiffer, R. Rabiser, M. Riebisch, and K. Schmid. Industry Voices on Software Engineering Challenges in Cyber-Physical Production Systems Engineering. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2022.

[FMS14] S. Friedenthal, A. Moore, and R. Steiner. *A Practical Guide to SysML: The Systems Modeling Language*. The MK / OMG Press. Elsevier Science, San Francisco, 3. edition, 2014. ISBN: 978-0-12-800202-5.

[FNG12] J. Fuchte, B. Nagel, and V. Gollnick. Automatic Fuselage System Layout using Knowledge Based Design Rules. In *Deutscher Luft- und Raumfahrt Kongress 2012*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2012.

[For22] M. Forlingieri. The four dimensions of Variability and their impact on MBPLE. In P. Arcaini, X. Devroey, and A. Fantechi, editors, *Proceedings of the 16th International Working Conference on Variability Modelling of Software-Intensive Systems*, New York, NY, USA, 2022. ACM.

[Fra21] Frankfurtflyer. SAS feiert die Airbus A321LR Premiere | Echte Business Class im Schmalrumpfjet, 2021. https://frankfurtflyer.de/sas-feiert-die-airbus-a321lr-premiere-echte-business-class-im-schmalrumpfjet/, Besucht am 28.12.2023.

[Fri10] S. Friedenthal. Modeling with SysML, Juli 2010. Tutorial, INCOSE 2010 Symposium.

[Fri18] C. Friedrich. *Modelle vs. Dokumente - Konzeptionelle Grundlagen und ihr Einsatz in der Produktentwicklung*. Studienarbeit, Technische Universität Kaiserslautern, 2018.

[Fuc14] J. Fuchte. *Enhancement of Aircraft Cabin Design Guidelines with Special Consideration of Aircraft Turnaround and Short Range Operations*. Dissertation, Technische Universität Hamburg, 2014.

[Fuc18] M. Fuchs. Ein MBSE-Ansatz für die Auslegung von Flugzeugkabinen am Beispiel eines Passenger Service Channel. Masterarbeit, Technische Universität Hamburg, Dezember 2018.

[FW22] M. Forlingieri and T. Weilkiens. Two Variant Modeling Methods for MBPLE at Airbus. *32nd Annual INCOSE International Symposium*, 2022.

[GAM+14] M. Grundel, J. Abulawi, G. Moeser, T. Weilkiens, A. Scheithauer, S. Kleiner, C. Kramer, M. Neubert, S. Kümpel, and A. Albers. FAS4M - No more: „Please mind the gap!“. In *Tag des Systems Engineering 2014*, pages 63–74. Carl Hanser Verlag, 2014.

[GDS+13] J. Gausemeier, R. Dumitrescu, D. Steffen, A. Czaja, O. Wiederkehr, and C. Tschirner. Systems Engineering in der industriellen Praxis, 2013. Heinz Nixdorf Institut, Universität Paderborn, Lehrstuhl für Produktentstehung, Fraunhofer-Institut für Produktionstechnologie IPT - Projektgruppe Entwurfstechnik Mechatronik, UNITY AG.

[GF16] M. Glawe and A. Fay. Wissensbasiertes Engineering automatisierter Anlagen unter Verwendung von AutomationML und OWL. *at - Automatisierungstechnik*, 64(3):186–198, 2016.

[GFBN23] Y. Ghanjaoui, M. Fuchs, J. Biedermann, and B. Nagel. Model-based design and multidisciplinary optimization of complex system architectures in the aircraft cabin. *CEAS Aeronautical Journal*, 14:895–911, 2023.

[GH20] I. Gräßler and J. Hentze. The new V-Model of VDI 2206 and its validation. *at - Automatisierungstechnik*, 68(5):312–324, 2020.

[Git25] GitHub. Python for Capella, 2025. https://github.com/labs4capella/python4capella?tab=readme-ov-file, Besucht am 07.08.2025.

[GKM23] A. Gaspar, B. Kukurza, and E. Martens. MBSE Model Integration in a Mixed-Fidelity Environment - Mixed Fidelity Model Integration, 18.07.2023. Vortrag, 33rd Annual INCOSE International Symposium, Honolulu, Hawaii, USA.

[Gli05] M. Glinz. Einführung in die Modelltheorie, 2005. Universität Zürich, Vorlesungsunterlagen, Informatik II: Modellierung.

[GMCR+18] M. Guenov, A. Molina-Cristobal, A. Riaz, S. Sharma, A. Murton, and J. Crockford. Aircraft Systems Architecting: Logical-Computational Domains Interface. *31st Congress of the International Council of the Aeronautical Sciences, ICAS 2018*, 2018.

[GV16] M. Grieves and J. Vickers. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In F.-J. Kahlen, S. Flumerfelt, and A. Alves, editors, *Transdisciplinary Perspectives on Complex Systems*, pages 85–113. Springer, Cham, 2016. ISBN: 978-3319387567.

[GWCS22] Y. Guo, J. Wei, G. Chen, and S. She. A Unified Model-Based Systems Engineering Framework Supporting System Design Platform Based on Data Exchange Mechanisms. In J. Chen, T. Hashimoto, X. Tang, and J. Wu, editors, *Knowledge and Systems Sciences: 21st International Symposium, KSS 2022, Beijing, China, June 11–12, 2022, Proceedings*, volume 1592 of *Communications in Computer and Information Science*, pages 99–112. Springer Nature Singapore, 2022. ISBN: 978-9811936104.

[GZC+18] Z. Guo, D. Zhou, J. Chen, J. Geng, C. Lv, and S. Zeng. Using virtual reality to support the product's maintainability design: Immersive maintainability verification and evaluation system. *Computers in Industry*, 101:41–50, 2018.

[Ham21] Hamburg Aviation. VIP Cabins: EXPLORER design study opens up new themed worlds, 2021. `https://www.hamburg-aviation.de/en/detail/details/news/vip-cabins-explorer-design-study-opens-up-new-themed-worlds.html`, Besucht am 19.10.2023.

[Har09] E. Harms. *Änderungs- und Konfigurationsmanagement unter Berücksichtigung von Verwendungsinstanzen: Arbeitsmethoden für integrierte Produktmodelle im Rahmen des Produkt-Lebenszyklus-Managements der Automobilindustrie*. Dissertation, Universität Karlsruhe, 2009.

[Har15] L. Hart. Introduction To Model-Based System Engineering (MBSE) and SysML, 30. July 2015. Präsentation, Delaware Valley INCOSE Chapter Meeting, `https://www.incose.org/docs/default-source/delaware-valley/mbse-overview-incose-30-july-2015.pdf`, Besucht am 04.02.2024.

[HB21] C. Hesse and J. Biedermann. A hybrid frequency-based/mode component synthesis model for cabin noise prediction with arbitrary damping distribution. In *27th International Congress on Sound and Vibration*, Juli 2021.

[HG23] E. Herzog and E. Gery. Tutorial: Digital threads with OSLC, 16.07.2023. Vortrag; 33rd Annual INCOSE International Symposium, Honolulu, Hawaii, USA.

[HH15] J. Hummell and M. Hause. Model-based Product Line Engineering - Enabling Product Families with Variants. In *2015 IEEE Aerospace Conference*, pages 1–8. IEEE, 2015. ISBN: 978-1479953790.

[HIL+12] H.Dubois, V. Ibanez, C. Lopez, J. Machrouh, N. Meledo, P. Mouy, and A. Silva. The product line engineering approach in a model-driven process. *Embedded Real Time Software and Systems (ERTS2012)*, 2012.

[Hil21] C. F. Hildebrandt. *Engineering ontologischer Modelle in der Automatisierungstechnik*. Dissertation, Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg, 2021.

[Hin19] M. Hinsch. Luftfahrttechnik-Management (6) - Systemintegration und Produktionsplanung bei der Flugzeugherstellung, 2019. `https://www.airliners.de/herstellung-systemintegration-produktionsplanung-flugzeugherstellung/50483`, Besucht am 20.10.2023.

[HJZ+21] G. Höpfner, G. Jacobs, T. Zerwas, I. Drave, J. Berroth, C. Guist, B. Rumpe, and J. Kohl. Model-Based Design Workflows for Cyber-Physical Systems Applied to an Electric-Mechanical Coolant Pump. *IOP Conference Series: Materials Science and Engineering*, 1097(1), 2021.

[HKRT05] V. G. Hegde, S. Kekre, S. Rajiv, and P. R. Tadikamalla. Customization: Impact on Product and Process Performance. *Production and Operations Management*, 14(4):388–399, 2005.

[HMWC13] A. Hall, T. Mayer, I. Wuggetzer, and P. Childs. Future aircraft cabins and design thinking: optimisation vs. win-win scenarios. *Propulsion and Power Research*, 2(2):85–95, 2013.

[HNC+24] E. Herzog, R. Nilsson, J. Crockford, A. Berezovskyi, T. Holm, J. Elkhoury, T. Ringenhall, E. Gery, and S. Albinsson. Enabling Digital Engineering with Federated PLM – Experiences from the Heliple–2 Project. *INCOSE International Symposium*, 34(1):1367–1383, 2024.

[Hoy24] F. Hoyos Garcia. Automatisierte Ansteuerung von gekoppelten Systemmodellen in der Flugzeugentwicklung. Bachelorarbeit, Hochschule für Angewandte Wissenschaften Hamburg, 2024.

[HSS+21] M. Hanna, J. Schwenke, L.-N. Schwede, F. Laukotka, and D. Krause. Model-based application of the methodical process for modular lightweight design of aircraft cabins. *Procedia CIRP*, 100:637–642, 2021.

[HST+23] H. Hick, S. Sanladerer, J. Trautner, K. Ryan, J. Piguet, F. Wilking, D. Horber, C. Faustmann, P. Kranabitl, S. Kollegger, M. Bajzek,

B. Schleich, and S. Wartzack. Combining System Models and CAD for Change Scenario Management. *33rd Annual INCOSE International Symposium*, 33(1), 2023.

[Hub84] V. Hubka. *Theorie Technischer Systeme: Grundlagen einer wissenschaftlichen Konstruktionslehre*. Springer-Verlag Berlin, Heidelberg, 2. edition, 1984. ISBN: 978-3662104460.

[HWFV12] R. Haberfellner, O. de Weck, E. Fricke, and S. Vössner. *Systems Engineering: Grundlagen und Anwendung*. Orell Füssli, 12. edition, 2012. ISBN: 978-3280040683.

[HWM22] S. Husung, C. Weber, and A. Mahboob. Integrating Model-Based Design of Mechatronic Systems with Domain-Specific Design Approaches. *Proceedings of the Design Society*, 2:1895–1904, 2022.

[IEE90] IEEE. IEEE Standard Glossary of Software Engineering Terminology. *IEEE Std 610.12-1990*, pages 1–84, 1990.

[INC07] INCOSE Technical Operations. Systems Engineering Vision 2020. Technical report, INCOSE, September 2007. INCOSE-TP-2004-004-02.

[INC19] INCOSE. About systems engineering. `https://www.incose.org/about-systems-engineering`, Besucht am 23.10.2019, 2019.

[INC21] INCOSE. Systems Engineering Vision 2035: Engineering Solutions for a Better World, 2021.

[Ins21] C. C. Insaurralde. Incorporation of Autonomous Model Analytics for Avionic System Design into Standard Framework for Integrated Engineering. In *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*. IEEE, 2021.

[Int19] International Air Transport Association. Best practices guide - Cabin interior retrofits and entry into service program, 2019.

[Int23] SAE International. Guidelines for Development of Civil Aircraft and Systems ARP4754A, 2023. `https://www.sae.org/standards/content/arp4754a/`, Besucht am 15.05.2023.

[ISO24] ISO International Organization for Standardization. ISO 10303-1:2024-01: Industrial automation systems and integration Product data representation and exchange Part 1: Overview and fundamental principles. Standard, ISO, January 2024.

[JC06] J. Jiao and C.-H. Chen. Customer Requirement Management in Product Development: A Review of Research Issues. *Concurrent Engineering*, 14(3):173–185, 2006.

[JDÖ17] C. Johansson, M. Derelöv, and J. Ölvander. How to use an optimization-based method capable of balancing safety, reliability, and weight in an aircraft design process. *Nuclear Engineering and Technology*, 49(2):404–410, 2017.

[JP14] P. Johannesson and E. Perjons. *An Introduction to Design Science*. Springer Cham, 2014. ISBN: 978-3319106311.

[JSH+20] S. Jaskó, A. Skrop, T. Holczinger, T. Chován, and J. Abonyi. Development of manufacturing execution systems in accordance with industry 4.0 requirements: A review of standard- and ontology-based methodologies and tools. *Computers in Industry*, 123, 2020.

[JSM+20] D. Jones, C. Snider, J. Matthews, J. Yon, J. Barrie, K. Robinson, and B. Hicks. Model-based information navigation for engineering documents. *Computers in Industry*, 121, 2020.

[Kan21] H. Kannan. Formal reasoning of knowledge in systems engineering through epistemic modal logic. *Systems Engineering*, 24(1):3–16, 2021.

[KBBT22] N. Külper, J. Bröhan, T. Bielsky, and F. Thielecke. Systems Architecting Assistant (SARA) - enabling a seamless process chain from requirements to overall systems design. *Proceedings of the 33rd Congress of the International Council of the Aeronautical Sciences*, 2022.

[KCH+90] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical report, Software Engineering Institute, Carnegie Mellon University, Fort Belvoir, VA, 1990. CMU/SEI-90-TR-21.

[KG18] D. Krause and N. Gebhardt. *Methodische Entwicklung modularer Produktfamilien - Hohe Produktvielfalt beherrschbar entwickeln*. Springer Vieweg Berlin, Heidelberg, 1. edition, 2018. ISBN: 978-3662530399.

[KK13] S. Kleiner and C. Kramer. Model Based Design with Systems Engineering Based on RFLP Using V6. In M. Abramovici and R. Stark, editors, *Smart Product Engineering, LNPE*, pages 93–102. Springer-Verlag Berlin Heidelberg, 2013.

[KKK+93] F.-L. Krause, F. Kimura, T. Kjellberg, S.C.-Y. Lu, van der Wolf, L. Alting, H. A. ElMaraghy, W. Eversheim, K. Iwata, N. P. Suh, V. A. Tipnis, M. Week, and A. C. H. Product Modelling. *CIRP Annals*, 42(2):695–706, 1993.

[KKT+18] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, 2018.

[KNK+20] C. Kotronis, M. Nikolaidou, G.-D. Kapos, A. Tsadimas, V. Dalakas, and D. Anagnostopoulos. Employing SysML to model and explore levels–of–service: The case of passenger comfort in railway transportation systems. *Systems Engineering*, 23(1):82–99, 2020.

[KP17] J. Kößler and K. Paetzold. Integration of MBSE into Existing Development Processes – Expectations and Challenge. *Proceedings of the 21st International Conference on Engineering Design (ICED17)*, Vol 3: Product, Services and Systems Design, 2017.

[KS14] B. Klöpper and J. C. Schlake. Aufbrechen der Datensilos - Big Data Forschungsfragen aus dem Bereich Industrial Analytics. In *Informatik 2014*, pages 79–81. Gesellschaft für Informatik e.V., Bonn, 2014. ISBN: 978-3885796268.

[KV23] B. Kaiser and A. Verl. Co-Design of Structural Timber Components Through Automated Model Generation for Manufacturing Simulation of Reconfigurable Manufacturing Systems. In *2023 29th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pages 1–6. IEEE, 2023.

[KVI+21] D. Krause, T. Vietor, D. Inkermann, M. Hanna, T. Richter, and N. Wortmann. Produktarchitektur. In B. Bender and K. Gericke, editors, *Pahl/Beitz Konstruktionslehre: Methoden und Anwendung erfolgreicher Produktentwicklung*, pages 335–396, Berlin, Heidelberg, 2021. Springer Vieweg. 9. edition, ISBN: 978-3662573037.

[LFH24] Z. Li, F. Faheem, and S. Husung. Collaborative Model-Based Systems Engineering Using Dataspaces and SysML v2. *Systems*, 12(1):18, 2024.

[LHK21] F. Laukotka, M. Hanna, and D. Krause. Digital twins of product families in aviation based on an MBSE-assisted approach. *Procedia CIRP*, 100:684–689, 2021.

[LHV15] E. Lelegems, J. Herssens, and J. Vanrie. A V-model for more. An inclusive design model supporting interaction between designer and user. *Proceedings of the 20th International Conference on Engineering Design (ICED15)*, 9:259–268, 2015.

[LLL20] T. Li, H. Lockett, and C. Lawson. Using requirement-functional-logical-physical models to support early assembly process planning for complex aircraft systems integration. *Journal of Manufacturing Systems*, 54:242–257, 2020.

[LRK22] F. Laukotka, C. Rennpferdt, and D. Krause. Digital Twins and Product-Service Systems: A Synergy with Challenges and Opportunities. *Proceedings of the Design Society*, 2:1639–1648, 2022.

[LS21] G. Lashin and R. Stark. Virtuelle Produktentwicklung. In B. Bender and K. Gericke, editors, *Pahl/Beitz Konstruktionslehre: Methoden und Anwendung erfolgreicher Produktentwicklung*, pages 1097–1153, Berlin, Heidelberg, 2021. Springer Vieweg. 9. edition, ISBN: 978-3662573037.

[Lud89] J. Ludewig. Modelle der Software-Entwicklung - Abbilder oder Vorbilder? *Softwaretechnik-Trends*, 9(3):1–12, 1989.

[LW10] J. Lamm and T. Weilkiens. Funktionale Architekturen in SysML. In M. Maurer and S.-O. Schulze, editors, *Tag des Systems Engineering 2010*, pages 109–118. Carl Hanser Verlag, München, Germany, 2010.

[Mae12] K. Maeda. Performance Evaluation of Object Serialization Libraries in XML, JSON and Binary Formats. In *2012 Second International Conference on Digital Information and Communication Technology and it's Applications (DICTAP)*, pages 177–182. IEEE, 2012. ISBN: 978-1467307345.

[Mah06] M. Mahnken. Integration von Kabinensystemen in BWB-Flugzeugkonfigurationen. Diplomarbeit, Hochschule für Angewandte Wissenschaften Hamburg, Januar 2006.

[Mah21] A. Mahboob. *Modelling and use of SysML behaviour models for achieving dynamic use cases of technical products in different VR-systems*. Dissertation, Technische Universität Ilmenau, 2021.

[MAK+24] A. Mehta, O. Alaiashy, P. Kumar, V. Tamilinian, S. Besong, S. Balpande, S. Verma, and K. Dhingra. Advancing model-based systems engineering in biomedical and aerospace research: A comprehensive review and future directions. *Journal of Knowledge Learning and Science Technology ISSN: 2959-6386*, 3(4):133–147, 2024.

[Män00] T. Männistö. *A Conceptual Modelling Approach to Product Families and their Evolution*. Dissertation, Helsinki University of Technology, 2000.

[Mar06] S. Marr. Feature-Diagramme und Variabilität. Hasso-Plattner-Institut, `https://stefan-marr.de/pages/feature-diagramme-und-variabilitat/`, Besucht am 29.08.2024, 2006.

[Mau07] M. Maurer. *Structural Awareness in Complex Product Design*. Dissertation, Technischen Universität München, 2007.

[MB 23] MB CAD GmbH. Digital Mock-up, 2023. `https://www.mbcad.de/glossar/digital-mock-up/`, Besucht am 04.02.2021.

[MBTdA20] M. Merzvinskas, C. Bringhenti, J.T. Tomita, and C.R. de Andrade. Air conditioning systems for aeronautical applications: a review. *The Aeronautical Journal*, 124(1274):499–532, 2020.

[MdSPF23] R. H. Madeira, D. H. de Sousa Pinto, and M. Forlingieri. Variability on system architecture using airbus mbple for moflt framework. *INCOSE International Symposium*, 33(1):601–615, 2023.

[Mei23] D. Meier. Modellbasierte Vorentwicklung eines Brennstoffzellensystems für große Verkehrsflugzeuge. Masterarbeit, Hochschule für Angewandte Wissenschaften Hamburg, 2023.

[MG22] R. A. McDonald and J. R. Gloudemans. Open vehicle sketch pad: An open source parametric geometry and analysis tool for conceptual aircraft design. In *AIAA SCITECH 2022 Forum*, 2022.

[MH23] T. McDermott and K. Henderson. Enterprise Adoption of DE and MBSE: Lessons from Research, 17.07.2023. Vortrag, 33rd Annual INCOSE International Symposium, Honolulu, Hawaii, USA.

[MHS+21] A. Myrodia, L. Hvam, E. Sandrin, C. Forza, and A. Haug. Identifying variety-induced complexity cost factors in manufacturing companies and their impact on product profitability. *Journal of Manufacturing Systems*, 60:373–391, 2021.

[MHW+19] A. Mahboob, S. Husung, C. Weber, A. Liebal, and H. Krömker. The Reuse of SysML Behaviour Models for Creating Product Use Cases in Virtual Reality. *Proceedings of the Design Society: International Conference on Engineering Design*, 1(1):2021–2030, 2019.

[MIKO99] T. Mori, K. Ishii, K. Kondo, and K. Ohtomi. Task Planning for Product Development by Strategic Scheduling of Design Reviews. In *19th Computers and Information in Engineering Conference*, volume 2, pages 115–126. American Society of Mechanical Engineers, 1999.

[Mis20] M. Misol. Active Sidewall Panels with Virtual Microphones for Aircraft Interior Noise Reduction. *Applied Sciences*, 10(19), 2020.

[MLKS22] K. Moenck, F. Laukotka, D. Krause, and T. Schüppstuhl. Digital Twins of existing long-living assets: reverse instantiation of the mid-life twin. In *DS 119: Proceedings of the 33rd Symposium Design for X (DFX2022)*. The Design Society, 2022.

[MLO23] S. McGuinness, J. La, and J. Obenland. REST API for Digital Thread Implementation, 20.07.2023. Vortrag, 33rd Annual INCOSE International Symposium, Honolulu, Hawaii, USA.

[Moh12] B. A. Mohr. *Methodik für den modellbasierten Vorentwurf von Flugzeugsystemen*. Dissertation, Technische Universität München, 2012.

[Mor23] Y. Mordecai. Toward Systems Engineering Meta-Methodology. *Proceedings of the 33rd Annual INCOSE International Symposium*, 33(1), 2023.

[Mot16] M. Motzer. *Integrierte Flugzeugrumpf- und Kabinenentwicklung mit graphenbasierten Entwurfssprachen*. Dissertation, Universität Stuttgart, 2016.

[MPWT22] S. Melzer, H. Peukert, H. Wang, and S. Thiemann. Model-based Development of a Federated Database Infrastructure to support the Usability of Cross-Domain Information Systems. In *2022 IEEE International Systems Conference (SysCon)*, pages 1–8. IEEE, 2022. ISBN: 978-1665439923.

[MS18] A. M. Madni and M. Sievers. Model-based systems engineering: Motivation, current status, and research opportunities. *Systems Engineering*, 21(3):172–190, 2018.

[MW19] C. Muggeo and R. Wolters. Model-based Systems Engineering: MBSE und PLM kombinieren und so modernste Entwicklungsprozesse implementieren. Whitepaper, 2019.

[MWH+17] A. Mahboob, C. Weber, S. Husung, A. Liebal, and H. Kroemker. Model Based Systems Engineering (MBSE) Approach for Configurable Product Use-Case Scenarios in Virtual Environments. *Proceedings of the 21st International Conference on Engineering Design (ICED17)*, 3, 2017.

[MWR+22] B. Menninger, D. Wiechel, S. Rackow, G. Höpfner, C. Oleff, J. Berroth, I. Gräßler, and G. Jacobs. Modellierung und Analyse funktionaler Varianz komplexer technischer Systeme. *Proceedings of the 33rd Symposium Design for X (DfX 2022)*, 2022.

[Nat07] National Aeronautics and Space Administration (NASA). *NASA Systems Engineering Handbook*. NASA SP. U.S. Government Printing Office, 2007. ISBN: 978-0160797477.

[Nat21] National Aeronautics and Space Administration (NASA). NASA PRODUCT DATA AND LIFE-CYCLE MANAGEMENT (PDLM) HANDBOOK, 2021. NASA Technical Handbook.

[NFSH99] H. Negele, E. Fricke, L. Schrepfer, and N. Härtlein. Modeling of integrated product development processes. *INCOSE International Symposium*, 9(1):1044–1052, 1999.

[NM09] F. C. Narváez and C. R. Monroy. How is Configuration Management in Aircraft Industry implemented? *Seventh LACCEI Latin American and Caribbean Conference for Engineering and Technology*, 2009.

[No 22a] No Magic Inc. Cameo Simulation Toolkit 19.0 LTR Documentation: User Guide, 2022. `https://docs.nomagic.com/display/CST190/Supported+scripting+languages`, Besucht am 05.06.2023.

[No 22b] No Magic, Inc. MagicDraw 2022x Documentation, 2022. `https://docs.nomagic.com/display/MD2022x/User+Guide`, Besucht am 12.06.2023.

[No 23] No Magic. No Magic 2022x API Document Overview - All Classes - Packages. `https://jdocs.nomagic.com/2022x/index.html`, Besucht am 26.01.2024, 2023.

[NZW+12] C. Noon, R. Zhang, E. Winer, J. Oliver, B. Gilmore, and J. Duncan. A system for rapid creation and assessment of conceptual large vehicle designs using immersive virtual reality. *Computers in Industry*, 63(5):500–512, 2012.

[OAS23] OASIS Open Project. Open Services for Lifecycle Collaboration, 2023. `https://open-services.net/`, Besucht am 14.06.2023.

[Obe17] M. Obergrießer. Parametrisch-assoziative modellierungsansätze. In *Digitale Werkzeuge zur integrierten Infrastrukturbauwerksplanung: Am Beispiel des Schienen- und Straßenbaus*, pages 59–134, Wiesbaden, 2017. Springer Fachmedien Wiesbaden. ISBN: 978-3658167820.

[Obj22] Object Management Group. OMG Systems Modeling Language (OMG SysML™) Version 1.7, 2022. https://www.omg.org/spec/SysML/1.7/.

[oos19] oose Innovative Informatik eG. Was ist systems engineering. https://www.oose.de/nuetzliches/was-ist-systems-engineering/, Besucht am 23.10.2019, 2019.

[PAC24] PACE. Pacelab ACE myCabin - Create LOPAs in 15min, 2024. https://pace.txtgroup.com/products/product-configuration/mycabin/, Besucht am 16.01.2024.

[PAD23] PADLab Software. Preliminary Aircraft Design Lab – PADlab, 2023. https://www.tu.berlin/luftbau/forschung/projekte/abgeschlossene-projekte/padlab, Besucht am 15.06.2023.

[Par04] T. Pardessus. Concurrent Engineering Development and Practices for Aircraft Design at Airbus. *24th International Congress of the Aeronautical Sciences*, 2004.

[PEM24] F. Prokic, C. Eriksson, and R. C. Munjulury. Design and visualization of a knowledge-based aircraft cabin in virtual reality. *Proceedings of the 34th Congress of the International Council of the Aeronautical Sciences*, 2024.

[Pen23] M. J. Pennock. Integrating Heterogenous Models. In A. M. Madni, N. Augustine, and M. Sievers, editors, *Handbook of Model-Based Systems Engineering*, pages 417–440, Cham, 2023. Springer International Publishing. ISBN: 978-3030274863.

[Pic15] A. C. R. Picard. *Integriertes Werkstückinformationsmodell zur Ausprägung werkstückindividueller Fertigungszustände*. Dissertation, Technische Universität Darmstadt, 2015.

[PN19] J. Page Risueno and B. Nagel. Development of a Knowledge-Based Engineering Framework for Modeling Aircraft Production. In *AIAA Aviation 2019 Forum*. American Institute of Aeronautics and Astronautics, 2019. ISBN: 978-1624105890.

[Pow23] J. A. Powelske. *MBSE-gestützte Methoden zur Strukturierung und Anwendung von Baukästen in der Frühen Phase der PGE – Produktgenerationsentwicklung mechatronischer Steuergeräte im Einsatz in Kleinantrieben im Automobilbereich*. Dissertation, Karlsruher Institut für Technologie, 2023.

[Pro22] Project Base. Definition Produktversion, 2022. https://projectbase.org/projektmanagement-glossar/produktversion/, Besucht am 26.10.2023.

[PS92] T. Petersen and P. Sutcliffe. Systems engineering as applied to the Boeing 777. In *Aerospace Design Conference*, Reston, Virigina, 1992. American Institute of Aeronautics and Astronautics.

[PTC23] PTC. pure::variants, 2023. https://www.pure-systems.com/de/purevariants, Besucht am 25.07.2023.

[PV22] M. Pech and J. Vrchota. The Product Customization Process in Relation to Industry 4.0 and Digitalization. *Processes*, 10(3):539, 2022.

[PYS14] K. Parhiala, M. Yalcinkaya, and V. Singh. Maintenance of Facilities and Aircrafts: A Comparison of IT-Driven Solutions. In *11th IFIP International Conference on Product Lifecycle Management (PLM)*, pages 11–20. Springer, 2014.

[Ren07] I. Renner. *Methodische Unterstützung funktionsorientierter Baukastenentwicklung am Beispiel Automobil*. Produktentwicklung. Hut, München, 1. edition, 2007. Dissertation, ISBN: 9783899635676.

[RFG+23] F. Rauscher, M. Fuchs, Y. Ghanjaoui, N. Markusheska, J. Biedermann, F. Meller, and B. Nagel. Permanently updated 3D-model of actual geometries of research environments. *CEAS Aeronautical Journal*, 14:739–751, 2023.

[RKS24] J.-E. Rath, J. Koch, and T. Schüppstuhl. Towards Model-Based Assembly System Configuration Supported by SysML and AutomationML. In F. J. G. Da Silva, A. Pereira, and R. D. S. G. Campilho, editors, *Flexible Automation and Intelligent Manufacturing: Establishing Bridges for More Sustainable Manufacturing Systems: Proceedings of FAIM 2023, June 18–22, 2023, Porto, Portugal, Volume 1: Modern Manufacturing*, Lecture Notes in Mechanical Engineering, Cham, 2024. Springer Nature Switzerland and Imprint Springer.

[RM24] J. L. Rebelo Moreira. The Role of Interoperability for Digital Twins. In T. P. Sales, S. de Kinderen, H. A. Proper, L. Pufahl, D. Karastoyanova, and M. van Sinderen, editors, *Enterprise Design, Operations, and Computing. EDOC 2023 Workshops*, pages 139–157, Cham, 2024. Springer Nature Switzerland. ISBN: 978-3031547126.

[RMME+22] F. Reimer, I. Moerland-Masic, A. End, J. Schadow, T.-M. Bock, F. Meller, and B. Nagel. Safety & Privacy in Urban Air Mobility (UAM) - A User Centric Design Approach Providing Insights into People's Preferences for UAM Cabin Designs. In *Human Factors in Transportation*, volume 60. AHFE, 2022.

[Roh20] F. Rohlf. Feasibility Study on Variability Realization Methods in SysML Models. Masterarbeit, Technische Universität Kaiserslautern, 2020.

[Rup14] C. Rupp. *Requirements-Engineering und -Management: Aus der Praxis von klassisch bis agil*. Hanser, München, 6. edition, 2014. ISBN: 978-3446443136.

[RW17] K. Richter and J. Walther. *Supply Chain Integration Challenges in Commercial Aerospace*. Springer International Publishing, Cham, 2017. ISBN: 978-3319461540.

[RWKGVV19] R. Rajamani, M. Whitfield, and R. Kumar G. V. V. Data Interoperability for Aerospace IVHM Systems. In *AeroTech Americas*. SAE International, 2019. SAE Technical Paper.

[SAMW17] B. Schleich, N. Anwer, L. Mathieu, and S. Wartzack. Shaping the digital twin for design and production engineering. *CIRP Annals*, 66(1):141–144, 2017.

[San20] R. S. Sanvordenker. Visualization and testing of an autonomously driving truck's SysML models in a virtual 3D simulation environment. Masterarbeit, Eindhoven Univerity of Technology, 2020.

[SATW20] R. Stark, R. Anderl, K.-D. Thoben, and S. Wartzack. WiGeP-Positionspapier: „Digitaler Zwilling". In *Zeitschrift für wirtschaftlichen Fabrikbetrieb*, volume 115, pages 47–50. Walter de Gruyter, 2020.

[SBA+21] A. Schäfer, M. Becker, M. Andres, T. Kistenfeger, and F. Rohlf. Variability realization in model-based system engineering using software product line techniques. In M. R. Mousavi and P.-Y. Schobbens, editors, *Proceedings of the 25th ACM International Systems and Software Product Line Conference - Volume A*, pages 25–34, New York, USA, 2021. ACM. 978-1450384698.

[SBWF10] R. Stark, G. Beier, T. Wöhler, and A. Figge. Cross-Domain Dependency Modelling – How to achieve consistent System Models with Tool Support. *7th European Systems Engineering Conference, EuSEC (Vol. 2010)*, 2010.

[SCC+23] M. Sadeghi, A. Carenini, O. Corcho, M. Rossi, R. Santoro, and A. Vogelsang. Interoperability of Heterogeneous Systems of Systems: Review of Challenges, Emerging Requirements and Options. In J. Hong, M. Lanperne, J. W. Park, T. Cerny, and H. Shahriar, editors, *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, pages 741–750, New York, NY, USA, 2023. ACM.

[SCD+10] M. Shafto, M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne, and L. Wang. DRAFT Modeling, Simulation, Information Technology & Processing Roadmap: Technology Area 11, 2010.

[Sch02] H. Schmidt. *Beitrag zum Variantenmanagement und zur Prozessoptimierung im Wagenkastenbau von Schienenfahrzeugen*. Dissertation, Technische Universität Berlin, 2002.

[Sch05] G. Schuh. *Produktkomplexität managen: Strategien - Methoden - Tools*. Hanser, München and Wien, 2. edition, 2005. ISBN: 978-3446400436.

[Sch15a] D. Schlabe. *Modellbasierte Entwicklung von Energiemanagement-Methoden für Flugzeug-Energiesysteme*. Dissertation, Technische Universität Dresden, 2015.

[Sch15b] D. Scholz. Aircraft Design, 2015. Lecture Notes, Hamburg University of Applied Sciences. http://LectureNotes.AircraftDesign.org, Besucht am 30.11.2023.

[Sch15c] D. Scholz. An Optiomal APU for Passenger Aircraft, 2015. 5th CEAS Air and Space Conference 2015, Delft, Foliensatz.

[Sch24] D. Scholz. PreSTo - Aircraft Preliminary Sizing Tool, 2024. https://www.fzt.haw-hamburg.de/pers/Scholz/PreSTo.html, Besucht am 13.04.2025.

[SD19] R. Stark and T. Damerau. Digital Twin. In S. Chatti and T. Tolio, editors, *CIRP Encyclopedia of Production Engineering*, Berlin, Heidelberg, 2019. Springer. ISBN: 978-3642359507.

[SDT18] G. Schuh, C. Dolle, and C. Tonnes. Methodology for the derivation of a digital shadow for engineering management. In *2018 IEEE Technology and Engineering Management Conference (TEMSCON)*. IEEE, 2018. ISBN: 978-1538620472.

[See08] K. Seeckt. Aircraft Preliminary Sizing with PreSTo - Re-Design of the Boeing B777-200LR, 2008. Bericht, https://www.fzt.haw-hamburg.de/pers/Scholz/GF/SEECKT-RE-KTH_Re-Design_B777_08-09-28.pdf, Besucht am 16.01.2024.

[Sen13] U. Sendler. *Industrie 4.0*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN: 978-3642369162.

[SH15] R. Sturm and M. Hepperle. Crashworthiness and ditching behaviour of blended-wing-body (BWB) aircraft design. *International Journal of Crashworthiness*, 20(6):592–601, 2015.

[She18] S. A. Sheard. Evolution of systems engineering scholarship from 2000 to 2015, with particular emphasis on software. *Systems Engineering*, 21(3):152–171, 2018.

[SHW+21] A. Shaikh, A. Hafeez, A. A. Wagan, M. Alrizq, A. Alghamdi, and M. S. A. Reshan. More Than Two Decades of Research on Verification of UML Class Models: A Systematic Literature Review. *IEEE Access*, 9:142461–142474, 2021.

[SI21] T. Schumacher and D. Inkermann. Heterogene Modellierung - Verknüpfung und Integration von Systemmodellen der SysML mit CAD-Modellen. In *DS 111: Proceedings of the 32nd Symposium Design for X*. The Design Society, 2021.

[Sin20] W. Sinn. Rules, tools and fools - was Topmanager aus der Krise lernen müssen: Wirtschaftsdeutschland nach dem Coronavirus, 13.07.2020. https://www.manager-magazin.de/unternehmen/industrie/coronavirus-und-die-folgen-fuer-unternehmen-veraenderungen-en-masse-a-a9d3e9aa-50a4-41a0-b308-6a310c19f55f, Besucht am 29.01.2021.

[SK97] Günter Spur and Frank-Lothar Krause. *Das virtuelle Produkt: Management der CAD-Technik*. Carl Hanser Verlag München Wien, 1997. ISBN: 978-3446191761.

[SKY23] SKYbrary Aviation Safety. ATA Classification, 2023. `https://skybrary.aero/articles/ata-classification`, Besucht am 05.12.2023.

[SMM22] T. Schmidt, A. Marbach, and F. Mantwill. Recommender Systems for Variant Management in the Automotive Industry. In *DS 119: Proceedings of the 33rd Symposium Design for X (DFX2022)*, page 10. The Design Society, 2022.

[SMW+21] V. Srinivasan, N. Markusheska, J.-N. Walther, J. Biedermann, F. Meller, and B. Nagel. Digital shadow model for automated cabin assembly process. In *Deutscher Luft-und Raumfahrt Kongress (DLRK) 2021*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2021.

[Sri22] V. Srinivasan. ROS framework to generate digital shadow for automated cabin assembly process. In *ONERA-DLR Aerospace Symposium*, Juni 2022.

[SS00] D. Scheithauer and A. Schindler. A Standardisation Concept for Non-Standard Development Projects. In *Proceedings of EuSEC 2000*, pages 83–92. GfSE, München, 2000.

[SS18] K. Sinha and E. S. Suh. Pareto-optimization of complex system architecture for structural complexity and modularity. *Research in Engineering Design*, 29(1):123–141, 2018.

[SS20] M. M. Schmidt and R. Stark. Model-Based Systems Engineering (MBSE) as computer-supported approach for cooperative systems development. *Proceedings of 18th European Conference on Computer-Supported Cooperative Work*, 2020.

[SS21] T. L. S. Santos and M. S. Soares. A qualitative study on sysml based on perceived views from industry professionals. In O. Gervasi, B. Murgante, S. Misra, C. Garau, I. Blečić, D. Taniar, B. O. Apduhan, A. M. A. C. Rocha, E. Tarantino, and C. M. Torre, editors, *Computational Science and Its Applications – ICCSA 2021*, pages 299–310, Cham, 2021. Springer International Publishing.

[SS23] T. L. S. Santos and M. S. Soares. A survey on what users think about SysML. *Systems Engineering*, 26(4):379–392, 2023.

[SSK19] F. Seiler, L.-N. Schwede, and D. Krause. MBSE-basierte Produktkonfiguratoren zur Analyse der Modularisierung bei der Entwicklung modularer Baukastensysteme. *Entwerfen Entwickeln Erleben EEE2019 Band 2*, pages 55–70, 2019.

[SSSW23] D. Schmeling, A. Shishkin, D. Schiepel, and C. Wagner. Numerical and experimental study of aerosol dispersion in the Do728 aircraft cabin. *CEAS Aeronautical Journal*, 14(2):509–526, 2023.

[Sta73] H. Stachowiak. *Allgemeine Modelltheorie*. Springer-Verlag, Wien und New York, 1973. ISBN: 3211811060.

[Sta22] J. Stark. *Product Lifecycle Management (Volume 1): 21st Century Paradigm for Product Realisation*. Decision Engineering. Springer, Cham, 5. edition, 2022. ISBN: 978-3030985783.

[Ste81] D. V. Steward. The design structure system: A method for managing the design of complex systems. *IEEE Transactions on Engineering Management*, EM-28(3):71–74, 1981.

[SW18] V. Singh and K. E. Willcox. Engineering Design with Digital Thread. *AIAA Journal*, 56(11):4515–4528, 2018.

[TE20] B. Tekinerdogan and F. Erata. Automated reasoning framework for traceability management of system of systems. *Science of Computer Programming*, 191, 2020.

[Tec08] G. Tecklenburg, editor. *Die digitale Produktentwicklung : Parametrisch assoziative Entwicklung von Baugruppen der Fahrzeugkarosserie; Visionen und Erfahrungen für zukünftige Entwicklungsprozesse*. Haus der Technik Fachbuch. expert-Verlag, Renningen, 2008. Band 89, ISBN: 978-3816927761.

[Tec10] G. Tecklenburg. *Design of Automotive Body Assemblies with Distributed Tasks under Support of Parametric Associative Design (PAD)*. Dissertation, University of Hertfordshire, Hamburg University of Applied Sciences, 2010.

[The23] The MathWorks, Inc. MATLAB Help Center, R2023a, 2023. `https://de.mathworks.com/help/matlab/index.html?s_tid=hc_panel`, Besucht am 25.08.2025.

[Tit21] P. Tittmann. *Graphentheorie: Eine anwendungsorientierte Einführung*, volume 4. Carl Hanser Verlag, München, 2021. ISBN: 978-3446471962.

[Top22] T. Topal. Konzepterstellung zur modellbasierten Beschreibung der Ausgangsdaten einer Flugzeugkonfiguration für eine künftige multidisziplinäre Optimierung in einem MBSE-Modell. Bachelorarbeit, HAW Hamburg, September 2022.

[vB72] L. von Bertalanffy. Vorläufer und Begründer der Systemtheorie. In R. Kurzrock, editor, *Systemtheorie*, pages 17–28, Berlin, 1972. Colloquium Verlag.

[VB24] V. Voth and O. Bertram. Aircraft System Design: A Model-based and Collaborative Approach. *Proceedings of the 34th Congress of the International Council of the Aeronautical Sciences*, 2024. ISSN 2958-4647.

[VDI10] VDI Verein Deutscher Ingenieure. Richtline 3633: Simulation von Logistik-, Materialfluss- und Produktionssystemen – Begriffe, 2010.

[VDI21] VDI Verein Deutscher Ingenieure. VDI 2206:2021 Entwicklung mechatronischer und cyber-physischer Systeme, November 2021. Beuth-Verlag, Berlin.

[Wac21] T. B. Wack. *Zur Co-Simulation heterogener mathematischer Modelle in der Verfahrenstechnik*. Dissertation, Ruhr-Universität Bochum, Bochum, 2021.

[Wal24] J.-N. Walther. *Knowledge-based Engineering to provide Aircraft Fuselage Design Details for Multidisciplinary and Multifidelity Analysis Model Generation*. Dissertation, Technische Universität Berlin, 2024.

[WAS+20] S. Wöhler, G. Atanasov, D. Silberhorn, B. Fröhler, and T. Zill. Preliminary Aircraft Design within a Multidisciplinary and Multifidelity Design Environment. In *Aerospace Europe Conference 2020*, April 2020.

[Wei08] T. Weilkiens. *Systems Engineering mit SysML/UML*. dpunkt Verlag, Heidelberg, 2. edition, 2008. ISBN: 978-3898645775.

[Wei16] T. Weilkiens. *Variant Modeling with SysML*. Tim Weilkiens, 1. edition, 2016. ISBN: 978-3981787542.

[Wei23] T. Weilkiens. Model Interoperability. In A. M. Madni, N. Augustine, and M. Sievers, editors, *Handbook of Model-Based Systems Engineering*, pages 815–831. Springer International Publishing, Cham, 2023. ISBN: 978-3030274863.

[Wei24] T. Weilkiens. The SysML v2 and a bit AI, 18.01.2024. Vortrag; DLR MBSE Ops Event.

[WGB+23] D. Wilke, R. Grothe, L. Bretz, H. Anacker, and R. Dumitrescu. Lessons Learned from the Introduction of Systems Engineering. *Systems*, 11(3):119, 2023.

[WHPK19] S. Wöhler, J. Hartmann, E. Prenzel, and H. Kwik. Preliminary Aircraft Design for a Midrange Reference Aircraft taking Advanced Technologies into Account as Part of the AVACON Project for an Entry into Service in 2028. In *Deutscher Luft- und Raumfahrt Kongress 2019*. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., 2019.

[Wie14] R. J. Wieringa. *Design Science Methodology for Information Systems and Software Engineering*. Springer Berlin Heidelberg, 2014. ISBN: 978-3662438381.

[Win23] Wingdesign GmbH. 3er LUFTHANSA Flugzeugsitzbank Aircraftseat, 2023. `https://wingdesign.com/shop/vergriffen/3er-lufthansa-flugzeugsitzbank/`, Besucht am 20.08.2023.

[WMC+20] S. Wolny, A. Mazak, C. Carpella, V. Geist, and M. Wimmer. Thirteen years of SysML: a systematic mapping study. *Software and Systems Modeling*, 19(1):111–169, 2020.

[Wol19] T. Wolfenstetter. *Model Integration and Traceability for Product Service Systems Engineering*. Dissertation, Technische Universität München, 2019.

[Wor16] World Wide Web Consortium. Extensible Markup Language (XML), 2016. `https://www.w3.org/XML/`, Besucht am 18.08.2023.

[Wor24] Working Group of OMG Systems Modeling Community (SMC). OMG Systems Modeling Language™ (SysML®) v2 Release, 2024. `https://github.com/Systems-Modeling/SysML-v2-Release`, Besucht am 06.03.2025.

[Wös15] C. Wösle. Integration von SysML und Modelica: Analyse und Ausarbeitung eines Prototyps zur automatisierten Integration von SysML und Modelica. Masterarbeit, Hochschule für Angewandte Wissenschaften München, 2015.

[WRF+15] D. D. Walden, G. J. Roedler, K. Forsberg, R. D. Hamelin, and T. M. Shortell. *INCOSE Systems Engineering Handbook: A guide for systems life cycle processes and activities*. John Wiley & Sons, 2015. Version 4, ISBN: 978-1118999400.

[WSSW22] F. Wilking, C. Sauer, B. Schleich, and S. Wartzack. SysML 4 Digital Twins - Utilization of System Models for the Design and Operation of Digital Twins. *Proceedings of the Design Society*, 2:1815–1824, 2022.

[WSW21] F. Wilking, B. Schleich, and S. Wartzack. Digital Twins - Definitions, Classes and Business Scenarios for Different Industry Sectors. *Proceedings of the Design Society*, 1:1293–1302, 2021.

[WZ20] Y. Wang and X. Zhang. Requirements Management Applied in Airworthiness Certification in the Civil Aircraft. *IOP Conference Series: Materials Science and Engineering*, 751(1), 2020.

[YC19] B. Young and P. Clements. Model Based Engineering and Product Line Engineering: Combining Two Powerful Approaches at Raytheon. *INSIGHT*, 22(2):67–75, 2019.

[YCP+17] B. Young, J. Cheatwood, T. Peterson, R. Flores, and P. Clements. Product Line Engineering Meets Model Based Engineering in the Defense and Automotive Industries. In M. Cohen, M. Acher, L. Fuentes, D. Schall, J. Bosch, R. Capilla, E. Bagheri, Y. Xiong, J. Troya, A. Ruiz-Cortés, and D. Benavides, editors, *Proceedings of the 21st International Systems and Software Product Line Conference - Volume A*, pages 175–179, New York, USA, 2017. ACM. ISBN: 978-1450352215.

[YHS09] T. Yoshikawa, S. Hayashi, and M. Saeki. Recovering traceability links between a simple natural language sentence and source code using

domain ontologies. In *IEEE International Conference on Software Maintenance, ICSM*, pages 551–554. IEEE, Canada, 2009.

# Abbildungsverzeichnis

# Tabellenverzeichnis

# A. Anhang

## A.1 Einblick in den Flugzeug- und Kabinensystementwurf

Der folgende Abschnitt gibt einen Einblick in den Produktentwicklungszyklus für Flugzeuge am Beispiel von Airbus. Anschließend wird eine Einführung in den Kabinen- und Systementwurf gegeben sowie Grundlagen zu der Klassifizierung der Flugzeugsysteme vorgestellt.

### A.1.1 Entwicklungsphasen des Flugzeugvorentwurfs

Das Flugzeug stellt ein komplexes Gesamtsystem dar, bei dem viele Teilsysteme miteinander gekoppelt sind [Mot16]. Im Wesentlichen gibt es für Großraumpassagierflugzeuge zwei große Wettbewerber (Airbus und Boeing). An das zu entwickelnde Produkt werden sehr hohe Sicherheitsanforderungen gestellt. Abbildung A.1 zeigt den strukturierten Entwicklungszyklus für Flugzeuge inklusive der Meilensteine. Dieser dient als Grundlage für alle Airbus-Programme [Par04]. Von der ersten Idee eines Flugzeugs bis zur Einführung beträgt der Entwicklungszeitraum bis zu 20 Jahre.

Die Entwicklung eines Flugzeugs erfolgt in fünf Phasen. Die drei vorgelagerten Phasen Machbarkeit (Feasibility), Konzept (Concept) und vorläufige Definition (Definition) sind stark interaktiv und iterativ [Par04]. In diesen Phasen gibt es viele Freiheitsgrade. In der Machbarkeitsphase werden zunächst die Erwartungen und Anforderungen des Marktes untersucht. Um diesen Anforderungen zu genügen, werden neue Technologien, Strukturkonzepte und Systemarchitekturen entwickelt und zu einem Gesamtkonzept integriert. Dabei gibt es verschiedene Optionen, die in einer Vielzahl von potentiellen Konzepten enden. Weiterverfolgt und vertieft im Detail ausgearbeitet werden nur Konzepte, die eine hohe Modularität sowie operationale Vorteile wie Kosten und Gewicht aufweisen. Das Ergebnis ist eine Produktspezifikation, mit der einerseits die Detailkonstruktion von Bauteilen begonnen wird oder die als Spezifikation von Systemelementen an Ausrüstungslieferanten versendet wird [Air23b]. Letzteres ist vor allem für die Kabine relevant, da diese zu 80% aus Zukaufteilen besteht [Air23c].

Zum Ende der Definitionsphase werden die Dimensionierungen aller elementaren Bauteile und ihrer Schnittstellen festgelegt und der Kabinenkonfigurationsguide für das Flugzeug eingefroren [Par04, Air23b]. Der Meilenstein M7 (Go ahead) markiert das Ende der Produktentwurfsdefinition. In der vierten Phase Entwicklung (Development) sind die Freiheitsgrade eingeschränkt und die industrielle Leistung (Verbindung

zwischen Fertigung und Konstruktion) steht im Vordergrund [Par04]. Hier werden alle Bauteile gefertigt und montiert. Ziel ist, mit Hilfe von Tests mit einem physischen Prototyp, das vorhergesagte Verhalten der Flugzeugsysteme und der Flugzeugstruktur zu überprüfen und nachzuweisen, dass diese in der Lage sind, allen Bedingungen während der Lebensdauer des Flugzeugs standzuhalten, einschließlich aller denkbaren Fehlerfälle [Air23b]. Diese Phase endet mit dem Meilenstein M13 (Entry Into Service, EIS). EIS steht für das Datum, an dem das Flugzeug vom Hersteller an den ersten Betreiber ausgeliefert wird und in den kommerziellen Betrieb geht. Dieser Meilenstein markiert den Beginn der letzten Phase, die Serienproduktion (Series). In dieser Forschungsarbeit wird der Fokus auf die zweite Phase (Concept) im Entwicklungsprozess gelegt.

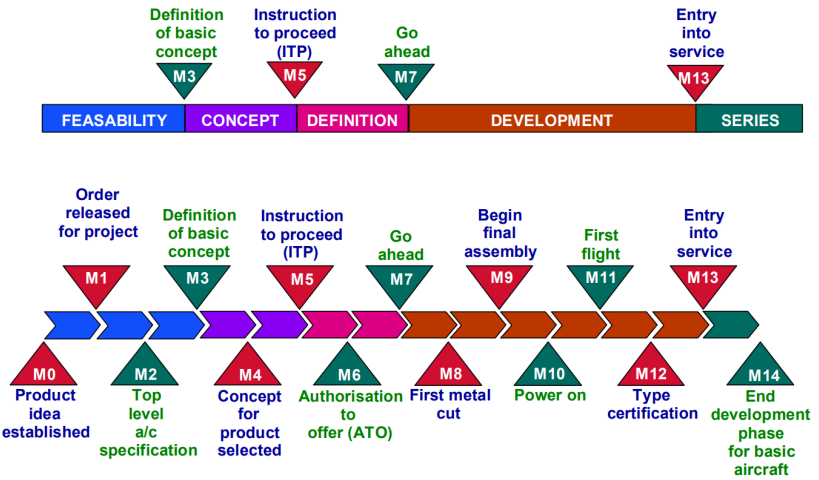



Abbildung A.1: Airbus-Entwicklungszyklus für Flugzeuge inklusive der Meilensteine von [Par04].

Die Konzeptphase unterteilt sich in weitere drei Abschnitte: den konzeptionellen Entwurf, Vorentwurf und Detailentwurf. Der Konzeptentwurf stellt den ersten Abschnitt dar. In diesem werden Konzeptskizzen eines Flugzeugs abgeleitet, basierend auf den Anforderungen an die Flugmission. Missionsanforderungen sind unter anderem der Zweck (Transport) sowie die Reichweite und die Anzahl der Passagiere. Zunächst werden die Anforderungen in Form von Kunden- und Zertifizierungsanforderungen auf Flugzeugebene gesammelt. Während dieses Abschnittes werden verschiedene mögliche Konfigurationen erzeugt, die unterschiedliche Hauptkomponenten besitzen und gleichzeitig die betrieblichen Anforderungen erfüllen. Der konzeptionelle Entwurf berücksichtigt Aerodynamik, Struktur und Antriebssysteme und legt die Form des Flugzeugs und die Position der Hauptkomponenten wie Flügel und Triebwerke fest. In diesem Abschnitt wird erstmals die Kabine in Form eines vorläufigen Sitzlayouts berücksichtigt. Ein erstes Layout umfasst die Platzierung und Anordnung der Sitze, der Küchen und der Waschräume für die Berechnung des benötigten Kabinenbereichs (cabin space) und für die Bestimmung des Platzes für die Nutzlast [Sch15b]. Der

Platz für die Nutzlast ist ein essentieller Bestandteil des konzeptionellen Vorentwurfs und stellt eine der top-level Randbedingungen dar. Dabei wird der top-down Gedanke verfolgt, bei dem zuerst die äußere Struktur des Flugzeugs entworfen wird und dann die inneren Komponenten des Flugzeugs (Kabine) passend ausgelegt werden.

Im nächsten Abschnitt, dem Vorentwurf, wird das im konzeptionellen Entwurf entwickelte Konzept optimiert und verfeinert. Die vorgegebenen Parameter an das Endprodukt leiten sich aus den Anforderungen ab. Dabei wird eine Flugzeugkonfiguration ausgesucht, die die gestellte Transportaufgabe unter Einhaltung der berücksichtigten Randbedingungen am wirtschaftlichsten erfüllt.

Während im Vorentwurf die Systeme und deren Komponenten grob ausgelegt werden, findet die detaillierte Auslegung aller Flugzeugkomponenten in Hinblick auf Position, Masse, Dimension und Anzahl im Detailentwurf statt. Zudem wird die Realisierung der Systemschnittstellen im Detail ausgearbeitet. Hier findet ebenfalls die detaillierte Auslegung der Kabinensysteme sowie deren Platzierung und das Kabelrouting statt.

Im Vergleich zu anderen Transportmitteln werden Flugzeuge in niedriger Stückzahl produziert und vorwiegend manuell gefertigt. Gleichzeitig stellt das Flugzeug aufgrund der Vielzahl an Subsystemen und Abhängigkeiten untereinander ein komplexes System dar. Bei der Entwicklung müssen viele Fachdisziplinen miteinander vernetzt werden, um alle Aspekte und Eigenschaften des Gesamtsystems abzudecken. Zudem ist der Entwurf, besonders der für die Kabine, durch die Zusammenarbeit mit vielen Zuliefern geprägt.

Angesichts der hohen Kundennachfrage zur Individualisierung der Flugzeuge bietet Airbus Anpassungsmöglichkeiten ihrer Produkte an. Nach Auswahl eines Flugzeugmodells sind allerdings Anpassungen für dieses nur innerhalb der Kabine möglich. Für die Kabine stellt Airbus eine breite Palette an Konfigurationsmöglichkeiten bereit. Dabei entstehen durch die Vielzahl an kundenspezifischen Anforderungen an die Kabinenkonfiguration jeweils neu entworfene Flugzeugversionen [BBH+18]. Bei einer Bestellung durch den Kunden werden in der Regel mehrere Flugzeuge mit derselben Konfiguration geordert. Das zuerst konfigurierte und gefertigte Flugzeug einer Bestellung wird als Head of Version (HoV) bezeichnet [PYS14]. Flugzeuge der gleichen Konfiguration, die im weiteren Verlauf gefertigt werden, sind Kopien der HoV [PYS14]. Mit Hilfe der hohen Individualisierungsmöglichkeit in der Kabine kann der Hersteller die unterschiedlichen Kundenbedürfnisse erfüllen, erhöht aber gleichzeitig die Komplexität im Entwurfsprozess. Das Resultat ist eine langer Produktentwicklungszyklus.

### A.1.2 Einführung in den Kabinen- und Systementwurf

Die Aufgabe der Kabine eines Großraumpassagierflugzeugs ist es, den Aufenthalt von Passagieren und der Crew in einer lebensfeindlichen Umgebung zu ermöglichen. Die Kabine dient dazu, dem Menschen eine angenehme und zugleich sichere Umgebung auf Reiseflughöhe zu bieten. Die meisten Passagiere erwarten zudem Komfort, Unterhaltung, Stauraum und Verpflegung [Fuc18]. In Notsituationen und bei Unfällen muss zudem gewährleistet sein, dass Menschenleben geschützt werden.

Die Kabine selber stellt ein komplexes System dar, aufgrund der vielen Funktionen die diese bereitstellen muss und der starken Vernetzung der Subsysteme. Deshalb sind

eine Vielzahl an Experten unterschiedlicher Fachdisziplinen am Entwicklungsprozess beteiligt und müssen miteinander kommunizieren. Verstärkt wird diese Herausforderung zusätzlich durch die Anbindung externer Experten und deren Wissen. Die Kabinenausstattung, Sitze und andere Elemente werden von Zulieferern entwickelt und in der Regel durch den Erstausrüster erworben. Flugzeughersteller wie Airbus agieren in der Rolle eines Integrators. Sie bauen die Kabine in die Primärstruktur des Flugzeugs ein, bevor die endgültige physische Validierung mit der Fluggesellschaft erfolgt [Air23b].

Zu der Kabine gehören die Baugruppen Sitze, Serviceeinrichtungen (Toilette, Bordküche und Passagierserviceeinheiten), Kabinensysteme, Gepäckablagen, Stauschränke, Ruheräume für die Besatzung (Crew Rest Compartments) und Verkleidung [Fuc18]. Für den Betrieb der Flugzeugkabine werden daher unterschiedliche Systemgruppen benötigt. Dazu gehören zum Beispiel das elektrische System, das Wasser-/Abwassersystem und das digitale System. Bei der Gestaltung der Kabine und deren Systeme müssen diverse Anforderungen und Sicherheitsvorschriften berücksichtigt werden.

Aufgrund der Komplexität des Gesamtsystems durch die Systemkopplungen treten Wechselwirkungen auf, die nicht immer direkt erkennbar sind. Hinzu kommt, dass der verfügbare Platz für die Platzierung der Kabinenkomponenten und -systeme durch die Struktur des Rumpfes begrenzt ist [FBBN22]. Änderungen an der Anordnung von Systemen führen zu verändernden Flugzeugkonfigurationen [FNG12].

Besonders die Integration neuer Technologien in die bereits bestehende Architektur erfordert neue Designvariationen unter Einhaltung der vorhandenen Qualitätsansprüche sowie die Kopplung zu vorhandenen Teilsystemen und steigert damit die Entwurfskomplexität [FBG+20, SS00]. Für die physische Installation im Flugzeug müssen die Kopplungspunkte des neu zu integrierenden Subsystems zu den Schnittstellen der weiteren Systeme definiert werden. Die Berechnung hierfür und Überprüfung erfolgt mit Modellen im Rahmen einer virtuellen Absicherung. Dabei müssen die Modelle jedes Systems ebenfalls über virtuelle Schnittstellen miteinander gekoppelt werden, um sowohl die direkten als auch die indirekten Wechselwirkungen auf weitere Teilsysteme abbilden zu können. Nur durch die Verknüpfung der verschiedenen Disziplinen mithilfe geeigneter Methoden oder Lösungen können diese vielfältigen Informationen im Algorithmus für den Entwurf des Kabinensystems berücksichtigt werden.

Daher wird einerseits vom System selber als auch von den Modellen bzw. Algorithmen für den Entwurf Modularität gefordert. Abbildung A.2 zeigt die Vernetzung zwischen Kabinenkomponenten am Beispiel eines Airbus A320. Die Kreise in Kombination mit Nummern im linken Bild stellen jeweils eine Kabinenkomponente dar. Die Linien zwischen zwei Kreisen zeigen die Vernetzung zwischen Kabinenkomponenten und wie diese in Verbindung stehen. Beispielsweise besteht zwischen einem Sitz und einer Sitzelektronikbox eine elektrische Verbindung, da letztere den Sitz mit ausreichend Strom versorgt. Im gezeigten Beispiel sind neben Komponenten der Verkleidung und der Sitze nur kabinenrelevante Komponenten wie die des elektrischen Systems, des Belüftungsverteilungssystems und der Passagierkommunikation abgebildet. Das Netz veranschaulicht die enge Verflechtung und Komplexität innerhalb des Subsystems Kabine. Die Netzstruktur und Komplexität steigen bei der Darstellung des Gesamtsystems Flugzeug nochmals deutlich an.

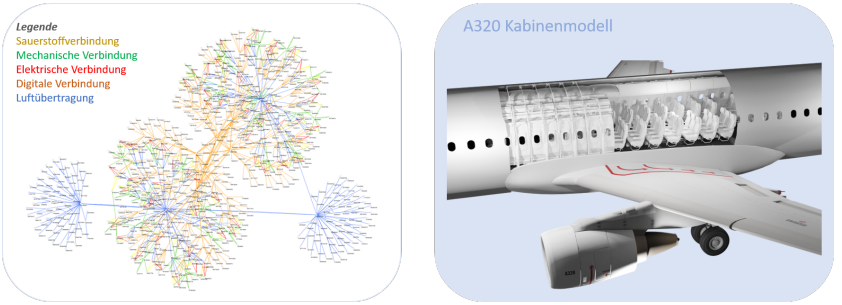



Abbildung A.2: Darstellung der engen Vernetzung der Kabinenkomponenten miteinander und Komplexität für das Subsystem Kabine abgegrenzt nach Systemgruppe am Beispiel einer A320, modifiziert von [FBBN22].

Änderungen an der Gesamtarchitektur des Flugzeugs haben Auswirkungen auf die Kabine. Dieser Einfluss muss in Modellen und frühen Phasen des Entwicklungsprozesses abgebildet und erkannt werden. Darüber hinaus liegen nur wenig Erfahrung oder Referenzsysteme für neuere Technologien vor, wodurch der konzeptionelle Entwurf mit Unsicherheiten behaftet ist. Zudem liegen teilweise noch keine Zertifizierungsspezifikationen seitens der Behörden vor [KBBT22]. Ein Beispiel ist die Einführung eines neuen Konzepts für die Klimatisierung in der Flugzeugkabine. Bei dem Flugzeugmodell 787-8 von Boeing (Dreamliner) wird für den Betrieb des Luftkreislaufsystems in der Kabine die erforderliche Druckluft von einem elektrisch angetriebenen Kompressor erzeugt [MBTdA20]. Vorher wurde die komprimierte Zapfluft des Motors aus den Triebwerken benötigt. Durch den entkoppelten Betrieb der Klimaanlage sinken das Systemgewicht, der Treibstoffverbrauch sowie die Betriebs- und die Wartungskosten [MBTdA20]. In den beiden folgenden Unterabschnitten werden die Klassifizierung der Kabinensysteme und die Sicherheitsanforderungen vorgestellt, die bei der Entwicklung verwendet werden.

#### ATA-Klassifizierung der Flugzeugsysteme

Für den grundlegenden Betrieb einer Flugzeugkabine werden verschiedene Systeme benötigt. Diese lassen sich wiederum in viele Untergruppen mit überlappenden Zugehörigkeiten unterteilen. Darüber hinaus sind die Wechselwirkungen und Abhängigkeiten untereinander und in Bezug auf den Passagier entscheidend für den Auslegungsprozess. So erfordern einige Systeme sowohl im Normal- als auch im Notfall eine direkte Zugänglichkeit und Erreichbarkeit durch den Fluggast vom Sitzplatz aus bei unterschiedlichen Körpergrößen. Für eine genaue Unterscheidung der Systeme wird eine numerische, technische Klassifizierung aller Systeme und deren Untersystemen vorgenommen. Entwickelt wurde diese von der ehemaligen Air Transport Association (ATA)[17], die mittlerweile in Airlines for America (A4A) umbenannt wurde [SKY23].

[17] ATA ist ein Dachverband amerikanischer Fluggesellschaften mit der Aufgabe als Interessenvertretung Regelungen für den Flugverkehr ggü. der US-Regierung durchzusetzen und Standards auszuarbeiten [Air23d].

Bei dieser Systematik werden alle technischen Einrichtungen des Flugzeuges in entsprechende Gruppen und Untergruppen eingeteilt. Die Unterteilung erfolgt auf Basis von 100 nummerierten Kategorien, die als „Kapitel“ zusammengefasst sind [SKY23]. Innerhalb der Kapitel gibt es weitere nummerierte Abschnitte und Unterabschnitte. Dabei fungieren die sogenannten ATA-Kapitel als eine generische, abstrakte Referenzarchitektur für Großflugzeuge.
Nachdem die Klassifizierung erstmals im Jahre 1956 eingeführt wurde, wurde diese branchenweit in der Flugzeugtechnik und -instandhaltung allgemein übernommen [SKY23]. Damit ermöglicht sie ein einheitliches Arbeiten und eine einfache Identifizierung der relevanten Komponenten und Systeme [FBBN22]. Insgesamt gibt es sechs Hauptgruppen für die Kabinensysteme. Diese sind die Systeme für Klimatisierung, Kommunikation, elektrische Versorgung, Beleuchtung, Sauerstoffversorgung und Wasser/Abwasser. Abbildung A.3 zeigt die Klassifizierung einiger ausgewählter Subsystemkomponenten am Beispiel der Flugzeugkabine nach den ATA-Kapiteln für die drei Gruppen Klimatisierung (Air Conditioning), Kommunikation (Communications) und Ausrüstung/Einrichtung (Equipment & Furnishings) [Air23a].

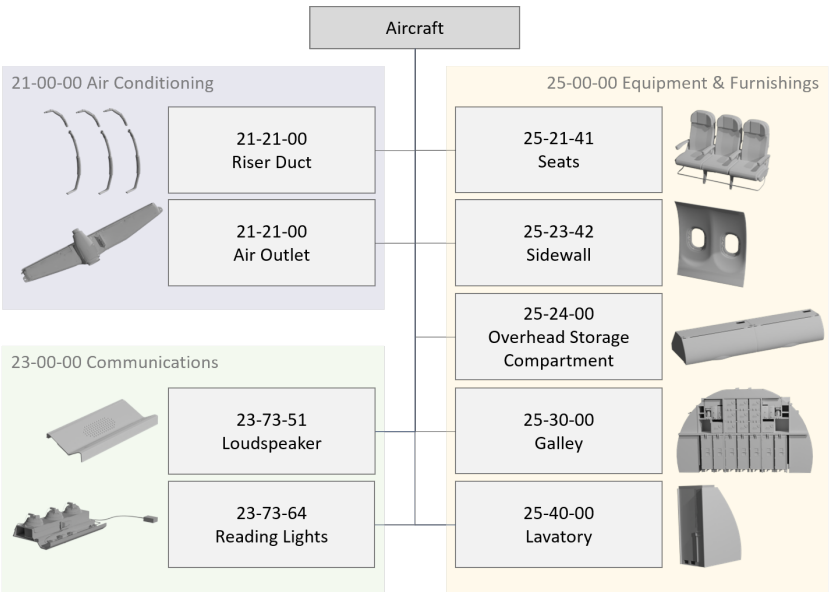



Abbildung A.3: Klassifizierung einiger Kabinenkomponenten nach den ATA-Gruppen- und Untergruppen, abgewandelt von [FBBN22].

Die ATA-Kapitel spiegeln im Wesentlichen die technischen und leistungsbezogenen Grundvoraussetzungen für das Flugzeugmodell wieder [RW17, S. 94]. Jedes Kapitel enthält entweder allgemeine Informationen wie Abmessungen oder technische Beschreibungen von beispielsweise den Triebwerken, der Kabinenausstattung, der Flugzeugstruktur oder den spezifischen Betriebsverfahren [RW17, S. 94].

**Sicherheitsanforderungen für Kabinensysteme**

Ein wichtiger Aspekt bei der Gestaltung der Kabinensysteme sind die Sicherheitsanforderungen. Diese sind als europäische Standards in einer Zulassungsvorschrift für Großflugzeuge (Certification Specifications for Large Aeroplanes, CS-25) der Agentur der Europäischen Union für Flugsicherheit (EASA) [Eur21] zusammengefasst. Die darin genannten Mindestanforderungen müssen für die Zulassung eines Flugzeugs dieser Klasse erfüllt werden. Im Auslegungsprozess werden die Sicherheitsanforderungen bei der Konzeption der Systeme berücksichtigt und anschließend mit verschiedenen und geeigneten Prüfmethoden auf ihre Konformität hin untersucht. Beispiele hierfür sind Einhaltung der Gangbreite vor den Notausgängen (CS 25.813 (a)) oder die Zugänglichkeit von Sauerstoffmasken für jeden sitzenden Passagier (CS 25.1447 (a)) [Eur21].

Um mögliche Einflüsse von neuen Konfigurationen nicht zu übersehen, müssen diese mit einem Referenzmodell gleicher Genauigkeit verglichen werden [WHPK19]. Der Entwicklungsprozess des Kabinensystems ist jedoch vom vorläufigen Flugzeugentwurf entkoppelt. Bei letzterem wird eine Grundkonfiguration des Flugzeugs ausgehend von einem begrenzten Satz von Top-Level-Anforderungen wie Reichweite, Reisegeschwindigkeit und Nutzlastgröße erstellt [CNL13]. Für die Auslegung der Kabinen und deren Systeme ist allerdings ein höherer Detaillierungsgrad z.B. der Rumpfstruktur erforderlich, als dieser im vorläufigen Entwurf bereitgestellt wird. In diesem werden häufig einfache Formen und vorläufige Positionierungen von Monumenten verwendet. Die Primärstruktur wird z.B. als einfache Hülle dargestellt ohne Angabe von z.B. Spanten. Dadurch fehlen die Informationen über Befestigungspunkte für die Platzierung der Gepäckablagen in der Kabine. Dementsprechend ist der Detaillierungsgrad des Rumpfes für die Bewertung und Platzierung von Kabinensystemen zu gering. Ein erster Ansatz wurde von Engelmann et al. vorgestellt, die den Parametersatz aus dem vorläufigen Entwurf zur Ableitung von Kabinenmodellen für die Untersuchung von Boarding-Simulationen verwenden [EKH20]. Boarding-Simulationen untersuchen den Prozess, wenn Passagiere die Flugzeugkabine betreten, falls vorhanden ihr Gepäck verstauen und ihre Sitze einnehmen. Damit können Einflüsse durch Änderungen am Kabinenentwurf auf die Leistungsfähigkeit eines Flugzeugs beim Einsteigen untersucht werden [EKH20]. Allerdings wird die Kabine auf einem niedrigeren Detaillierungsgrad modelliert und ihre Systeme nicht berücksichtigt.

Die Entscheidungsfindung beim Design ist bestimmend, um zu einem realisierbaren und rentablen Kabinendesign zu gelangen [HMWC13]. Entscheidet ein Flugzeughersteller Innovationen nur geringfügig zu berücksichtigen, führt dies dazu, dass möglicherweise veraltete Technologien genutzt werden und das Unternehmen gegenüber Konkurrenten am Markt zurückfällt [HMWC13]. In einem schnelllebigen Technologiesektor ist die Anpassungsfähigkeit an neue Technologien wichtig. Daher muss der Kabinen- und Systementwurf frühzeitig an den Flugzeugentwurf angebunden werden, so dass ein hochauflösendes Modell automatisch auf der Grundlage des Parametersatzes des vorläufigen Flugzeugentwurfs generiert werden kann. Dadurch können disruptive Konzepte untersucht und weiterführende Forschungsmethoden wie Produktionsplanung oder Komfortevaluierungen durchgeführt werden.

## A.2 Vorstellung der SysML-Elemente in der Modellierungsumgebung Cameo Systems Modeler

Dieser Abschnitt stellt die spezifischen Knoten- und Verbindungselemente der SysML vor. Zudem wird eine Einführung in die für die Arbeit angewandten Diagramme an Beispielen, generiert mit dem Cameo Systems Modeler, gegeben.

Die SysML setzt sich aus zwei Teilen zusammen. Diese sind die grafischen Knotenpunkte und die grafischen Pfade, die zusammengefasst die Sprachelemente darstellen. Abbildung A.4 zeigt den Zusammenhang zwischen den Elementen. Sowohl die grafischen Knotenpunkte als auch die Pfade sind Teil der Sprachelemente, dargestellt durch die Generalisierung (Pfeil mit geschlossener und nicht aufgefüllter Spitze). Diese gerichtete Beziehung erzeugt eine Eltern-Kind-Abhängigkeit zwischen den Elementen. Des Weiteren sind die Knotenpunkte mit den Pfaden über eine Spezialisierung/Komposition miteinander verknüpft (Pfeil mit ausgefülltem Diamant). Allgemein können grafische Knoten alleine existieren, während für einen Pfad mindestens zwei grafische Knoten existieren müssen, die dann wiederum über einen Pfad in Beziehung gestellt werden können.

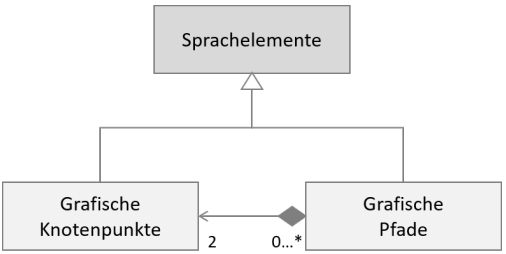



Abbildung A.4: SysML Notation dargestellt als Klassendiagramm.

Bei der Modellierung werden für jedes Diagramm unterschiedliche Elemente bereitgestellt. Dabei können auch diagrammfremde Elemente verwendet werden, sofern dadurch die Modelle in ihrer Klarheit nicht eingeschränkt werden [Abu12]. Ein Einblick in die wichtigsten SysML-Modellelemente dieser Arbeit wird im Folgenden gegeben.

#### Blöcke

Blöcke sind ein grundlegendes Sprachelement in der SysML. Sie werden als universelle Repräsentation von Objekten und für die Beschreibung statischer Systemstrukturen verwendet [Abu12]. Ein Block bietet ein einheitliches Konzept für die Beschreibung einer Entität [Fri10]. Das können ein System, eine Komponente, eine Person, Daten oder ein Verfahren sein. Die Definition und Visualisierung eines Blocks erfolgt im Blockdefinitionsdiagramm. Standardmäßig wird ein Block als Rechteck dargestellt mit einem eindeutigen Namen und dem Schlüsselwort mit dem benutzten Stereotypen (Standard «block»). Ein Stereotyp kann jedes Modellelement erweitern und ermöglicht die Definition eigener Metaklassen. Der Block ist eine modulare Einheit

und unterteilt sich in mehrere Fächer (Compartments), mit denen die Blockmerkmale beschrieben werden können (Abbildung A.5) [Obj22]. Zu diesen Attributen gehören Eigenschaften, Operationen, Beschränkungen, Zuweisungen von/zu anderen Modellelementen (z.B. Aktivitäten), Anforderungen die der Block erfüllt und benutzerdefinierte Fächer [Fri10].

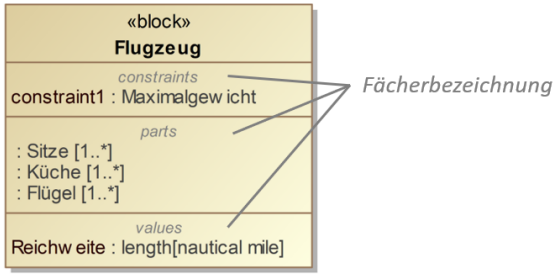



Abbildung A.5: Strukturelement Block.

Blöcke sind eine Definition oder ein Typ und können in verschiedenen Kontexten im Modell wiederverwertet werden. Die SysML gibt keine Beschränkung vor, welche Art von Systemen oder Systemelemente durch einen Block beschrieben werden können [Obj22].

#### Blockeigenschaften

Für jede Fächer-Kategorie werden bestimmte Informationen in den Attributen gespeichert. Nähere Informationen zum Aufbau eines Blocks sind in [Obj22] zu finden. Ein genauerer Einblick wird für die Kategorien Werteigenschaften (values) und Beschränkungen (contraints) gegeben.

Eine quantitative Eigenschaft des enthaltenden Blocks wird in der SysML als Werteigenschaft (value) klassifiziert. Beispiele für Werteigenschaften sind die Länge oder das Gewicht eines physischen Teils [BCS+20]. Der Datentyp (z.B. boolean, string, integer) eines Values ist frei wählbar. Der Cameo Systems Modeler bietet die Möglichkeit SI-Einheiten für die Bezeichnung der Eigenschaften im Projekt zu verwenden, sodass bei der Modellierung einheitliche Größen wie Kilogramm (kg) oder Meter (m) verwendet werden können. Zusätzlich kann ein vorgegebener Wert für die Values hinterlegt werden.

Mit Hilfe von Constraints können physikalische Eigenschaften eines Systems beschränkt werden. Die Constraints definieren Randbedingungen, mit denen kritische Leistungsparameter und ihre Beziehungen zu anderen Parametern ermittelt werden können [Obj22]. Diese Bedingungen können entweder ein beliebig komplexer mathematischer oder ein logischer Ausdruck sein [Obj22]. Ein Constraint setzt sich aus dem Beschränkungsausdruck und den benötigten Parametern für die Überprüfung des Ausdrucks zusammen. Die Werte für die Parameter erhält das Element von dem

eigenen Block oder über die Werteigenschaften von externen Blöcken. Über Constraints können Anforderungen verfeinert werden. Indem diese mit den Anforderungsblöcken verknüpft werden, kann die Überprüfung der Anforderung über die Berechnungsformel aus dem Constraint erfolgen.

### Beziehungen

Beziehungen zwischen Blöcken werden unter anderem durch Assoziation, Komposition und Generalisierung spezifiziert. Weitere Informationen zu Beziehungen zwischen Blocken sind [BCS+20] und [Obj22] zu entnehmen. Abbildung A.6 zeigt je ein Beispiel für die drei Beziehungsarten. Die Assoziation beschreibt die Verbindung zwischen zwei Blöcken, wobei nicht beschrieben wird, wie diese interagieren [BCS+20]. Die binäre Verbindung kennzeichnet sich durch eine gerichtete, offene Pfeilspitze. Im gezeigten Beispiel kennt der Server die Eigenschaften vom Client und kann dessen Anfragen verarbeiten. Der Client wiederum kennt die Eigenschaften vom Server nicht. Mit der Komposition wird ausgedrückt, dass eine Instanz eines Blocks Teil einer Instanz eines anderen Blocks ist [BCS+20]. Die Komposition wird durch einen Pfeil mit einer geschlossenen Raute am anderen Ende symbolisiert. Der Flügel ist ein Teil des Flugzeugs (Abbildung A.6) und wird als dieser im Fach parts im Block gespeichert (Abbildung A.5). Das Flugzeug wiederum hat mindestens einen Flügel. Die Zahl an der Pfeilspitze gibt die Multiplizität an. In dem gezeigten Beispiel kann es unbegrenzt viele aber es muss mindestens einen Flügel geben. Mit Hilfe der Multiplizität können bei Instanziierung von Objekten entweder mehrere oder eine genau definierte Anzahl an weiteren Teilinstanzen automatisch erzeugt werden. Bei jeder Art einer Verbindung in einem SysML-Modell kann die Kardinalität (Multiplizität) einer Eigenschaft spezifiziert werden, muss aber nicht [No 22b].

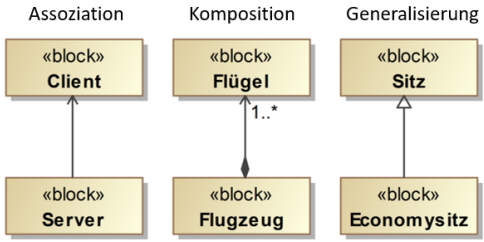



Abbildung A.6: Beziehungen zwischen Blöcken.

Zuletzt beschreibt die Generalisierung die Vererbung zwischen Modellelementen. Symbolisiert wird diese Verbindung durch einen Pfeil mit einer offenen Spitze. Die Vererbung ermöglicht die einmalige Definition eines Konzepts ohne Mehrfachdarstellung [BCS+20]. Das generalisierte Element, in Abbildung A.6 der Block Sitz, wird als Supertyp bezeichnet. Der Block Economysitz, der die Eigenschaften des Supertyps erbt, wird als Subtyp bezeichnet. Der Subtyp stellt einen spezifischeren Block des übergeordneten Konzepts dar.

### Instanzen

Eine Instanz ist eine realisierte Variante eines Blocks. Sie spezifiziert die Existenz einer Entität und beschreibt diese vollständig oder teilweise [No 22b]. Das Erstellen einer Instanz kann über die Simulation (z.B. Ausführung Aktivitätsdiagramm) des Modells erfolgen und wird Instanziierung genannt. Die Instanz wird durch ein Rechteck ähnlich dem Block dargestellt, bei dem der Name unterstrichen wird. Die Instanz enthält spezifische Werte für die Eigenschaften und verfügt über die gleichen Beziehungen und Attribute des zugehörigen Blocks. Abbildung A.7 zeigt eine Instanz vom Typ Economysitz im Cameo Systems Modeler einmal als Block und einmal zur Laufzeit im Workspace mit ausgefüllten Eigenschaften.

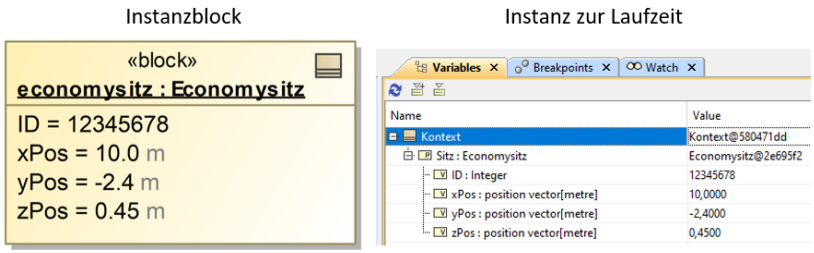



Abbildung A.7: Instanz eines Economysitzes als Block (links) und dargestellt zur Laufzeit (rechts) im Cameo Systems Modeler.

### Aktionselement

Eine Aktion ist das Hauptelement für das Aktivitätsdiagramm. Es stellt die grundlegende Einheit einer ausführbaren Funktionalität dar [No 22b]. Mit der Ausführung einer Aktion werden festgelegte Berechnungen oder Befehle durchgeführt, die eine Transformation oder Verarbeitung in dem modellierten System repräsentieren [No 22b]. Damit ist das Aktionselement ein wesentlicher Bestandteil für die Kopplung mit externen (heterogenen) Modellen. Der Fokus liegt auf vier Metaklassen. Abbildung A.8 zeigt Beispiele für jedes Aktionselement. Erstens, die Read Self Action, mit der ein Objekt aus dem Kontext der ausgeführten Aktion zurückgegeben wird. Damit können alle Instanzen während der Laufzeit einer Simulation gelesen und deren Eigenschaften verändert werden. Zweitens, die Leseaktion Read Structural Feature Action, mit der Werte eines Strukturmerkmals abgerufen werden. Teilinstanzen oder spezifische Eigenschaften (wie z.B. die ID) von Instanzen können mit diesem Element ausgelesen und im Diagramm weiter verarbeitet werden. Drittens, die Opaque Action, mit der implementierungsspezifische Aktionen ausgeführt werden können [No 22b]. Diese Aktion hat keine spezifische Notation. Die spezielle Eigenschaft (Body and Language) der Opaque Action ermöglicht die Implementierung von Programmierbefehlen (z.B. Erzeugen einer Instanz) oder die direkte Ansteuerung von Matlab (s. Abschnitt 5.2.2). Viertens, die Call Behavior Action, mit der ein Verhalten aufgerufen und ausgeführt werden kann. Zusätzlich können mit dieser Aktion auf Befehle der grundlegenden

virtuellen Maschine der UML (fUML)[18] zugegriffen und für die automatisierte Speicherung von Instanzen genutzt werden. Ein Einblick in die Verwendung der Elemente erfolgt in Abschnitt 7.4.2.

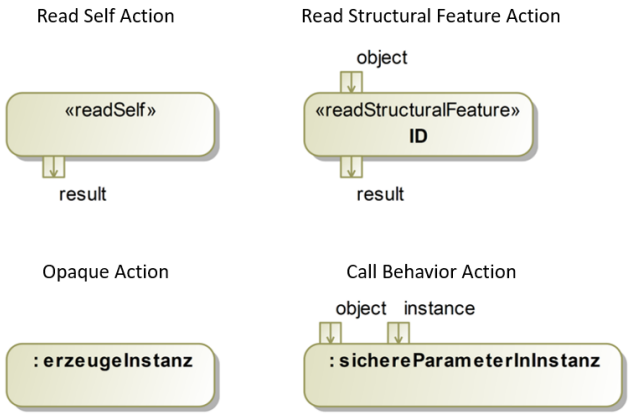



Abbildung A.8: Die Aktionselemente Read Self Action, Read Structural Feature Action, Opaque Action und Call Behavior Action im Cameo Systems Modeler.

**Vorstellung der angewandten SysML-Diagramme und -Matrizen**

Diagramme und Matrizen visualisieren und definieren einen bestimmten Teilaspekt eines Systems. Im Summe kann damit ein System beschrieben werden und ermöglicht dem Nutzer mehrere Systemansichten. Im Rahmen dieser Forschung werden vor allem Paket-, Blockdefinitions-, Interne Block-, Anwendungsfall-, Anforderungs- und Aktivitätsdiagramme verwendet. Die Diagrammtypen sind in Abbildung A.9 dargestellt. Darüber hinaus werden Anforderungstabellen und Matrizen (Satisfy-Requirement-Matrix) verwendet. Bei allen Diagrammen kann der Diagrammrahmen die Grenze des zu modellierenden Systemausschnitts kennzeichnen [Abu12]. Alle Elemente für die Beschreibung der Systemansicht befinden sich innerhalb dieser Grenze. Ausnahme bilden Knotenelemente, die als Ein- oder Ausgänge die Schnittstelle zu anderen Elementen in der Umgebung darstellen [Abu12]. Diese gehen über den Diagrammrahmen hinaus. Des Weiteren können verschiedene Diagramme und ihre Elemente miteinander verknüpft werden. Die generierten SysML-Modellelemente können durch Hyperlinks mehrfach im Modell und über die Diagrammgrenze hinaus genutzt werden. Zusätzlich können in einem Diagramm Links zu weiteren Diagrammen hinterlegt werden, sodass ein direktes Aufrufen dieser aus dem Diagramm heraus möglich ist. Dadurch kann der Modellierer sich durch das Modell navigieren und tiefergehende Einblicke in die Systemstruktur erhalten.

[18] OMG Semantics of a Foundational Subset for Executable UML Models (fUML): https://jdocs.nomagic.com/190/CST/com/nomagic/magicdraw/simulation/fuml/fUMLHelper.html

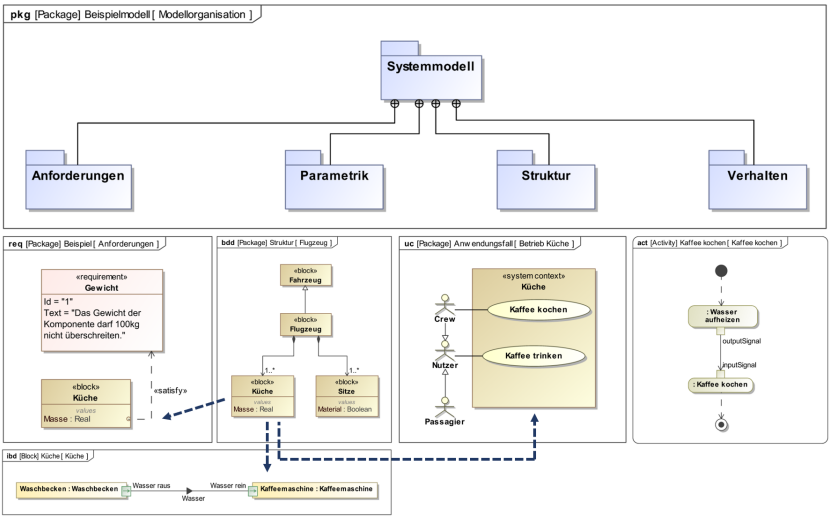



Abbildung A.9: Beispiele für die angewandten Diagrammtypen Paketdiagramm, Anforderungsdiagramm, Blockdefinitionsdiagramm, Anwendungsfalldiagramm, Aktivitätsdiagramm und Internes Blockdiagramm, abgewandelt von [Fri10].

### Paketdiagramm (pkg)

Das Paketdiagramm veranschaulicht die Struktur, die Gliederung und Organisation von SysML-Modellen (Abbildung A.9). Modellelemente werden in einzelne Paketelemente aufgeteilt, die entweder nach Systemhierarchie oder nach Diagrammtyp organisiert sein können [Fri10]. Abhängigkeiten zwischen den Paketen werden mit einem Fadenkreuz hergestellt, welches zum übergeordnetem Paket (Eltern-Kind Beziehung) zeigt. Dieses Organisationsprinzip unterstützt bei der eindeutigen Benennung von Modellelementen [Obj22].

### Blockdefinitionsdiagramm (bdd)

Im Blockdefinitionsdiagramm werden Bestandteile des betrachteten Systems sowie dessen Wechselwirkungen und Beziehungen dargestellt. Dabei wird ein System hierarchisch in seine Subsysteme und Bestandteile untergliedert (Abbildung A.9). Mit Hilfe von Blöcken werden die einzelnen Systemelemente definiert. Die SysML stellt dafür verschiedene Elemente bereit, um die strukturellen und funktionalen Charakteristiken modellieren zu können. Für die Beziehungen werden die Verbindungselemente Assoziation und Generalisierung verwendet.

**Internes Blockdiagramm (ibd)**

Das Interne Blockdiagramm veranschaulicht die innere Struktur eines Systembestandteils und wird meistens aus dem bdd abgeleitet (Abbildung A.9). Während ein Systembestandteil im Blockdefinitionsdiagramm nur als Black-Box abgebildet ist, werden dessen innere Strukturen als White-Box im ibd sichtbar. Im Internen Blockdiagramm werden Parts verwendet, die den Nutzen und die Verbindung eines Blocks in einem spezifischen Kontext darstellen. Besonders für die Darstellung von Schnittstellen und den Austausch von Signalen oder Stoff- und Energieflüssen eignet sich das Diagramm [Abu12].

**Anwendungsfalldiagramm (uc)**

Mit dem Anwendungsfalldiagramm können Szenarien oder Teilfunktionen des Systems und ihre Abhängigkeiten sowie Beziehungen zu den Anwendern dargestellt werden [Abu12]. Ziel ist die Darstellung aller Anwendungsfälle des Systems aus Sicht der Nutzer. Der Nutzer, auch Akteur genannt, wird als Strichmännchen veranschaulicht (Abbildung A.9). Weitere Elemente sind das Subjekt, stellvertretend für das zu entwickelnde System (Rechteck), die Anwendungsfälle (Oval) und die Beziehungen (Striche und Pfeile).

**Anforderungsdiagramm (req)**

Das Anforderungsdiagramm wird für die Darstellung und Gliederung der Systemanforderungen verwendet. Dabei können die Anforderungen noch verfeinert werden und die Beziehungen zu den anderen SysML-Modellelementen modelliert werden. Dadurch können die Anforderungen auf ihre Erfüllung hin überprüft und ihre Beziehungen zu den anderen Modellelementen verfolgt (siehe Matrizen) werden. Das Stereotyp «requirement» repräsentiert eine textbasierte Anforderung (Abbildung A.9). Es besteht aus einer ID und Texteigenschaften. Darüber hinaus kann es definierte Eigenschaften wie Verifikationsmethoden oder Anforderungskategorien (z.B. funktional, Leistung) besitzen [Fri10]. Beziehungen werden mit einer gestrichelten Linie und einem Pfeil am Ende symbolisiert. Zu den Beziehungstypen gehören satisfy, derive, trace, refine, copy and verify. In dieser Arbeit wird vorwiegend die satisfy-Beziehung genutzt. Diese zeigt die Abhängigkeit zwischen einem Modellelement und der zu erfüllenden Anforderung an.

**Aktivitätsdiagramm (act)**

Mit dem Aktivitätsdiagramm können einerseits systeminterne Funktionsabläufe dargestellt und andererseits Simulationen ausgeführt werden (Abbildung A.9). Dafür stellt das Diagramm Elemente bereit, mit denen mathematische Beschreibungen und Berechnungen von Systemparametern möglich sind. Das Element Aktivität beschreibt die Umwandlung eines Inputs in einen Output durch eine kontrollierte Abfolge von Aktionen [Fri10]. Neben Kontrollflüssen die eine Weitergabe der Ausführung an die nächsten Aktion darstellen, können mit Hilfe von Objektflüssen Objekte zwischen den

Aktionen übertragen werden. Zudem können mit dem Element Aktion externe Simulationsprogramme wie Matlab/Simulink angesteuert und für die Simulation des Systemverhaltens und der -auslegung genutzt werden.

### Tabellen und Matrizen

Zusätzlich zu den Diagrammen stellt die Modellierungsumgebung Matrizen und Tabellen bereit. Im Folgenden werden die Anforderungstabelle, die Übersichtstabelle und die Satisfy-Requirement-Matrix vorgestellt. Die Anforderungstabelle enthält Anforderungen und stellt deren Eigenschaften in einer Tabelle dar (Abbildung A.10a). Standardmäßig werden vier Spalten angezeigt. Diese zeigen eine fortlaufende Nummerierung der Anforderungen, die ID, den Namen und den Text. Ergänzend können weitere Spalten mit Eigenschaften, z.B. welches Element die Anforderung erfüllen muss (satisfied by), hinzugefügt werden. Mit einer Übersichtstabelle können eigene Tabellen für die Rückverfolgung von Modellelementen oder für die Überprüfung der Modellierung erstellt werden. Abbildung A.10b zeigt eine Traceability-Tabelle für die Analyse von Anwendungsfällen am Beispiel des Kabinensystems Passagierserviceeinheit (Passenger Service Unit, PSU) [FGA+22]. Die Tabelle zeigt die Vernetzung der einzelnen Modellelemente untereinander ausgehend von den definierten Anwendungsfällen der Subsysteme.

| # | Id | Name | Text | Satisfied By |
|---|---|---|---|---|
| 1 | 2 | 2 Pre-assembly | The Hatrack shall form a compact module together with the passenger function components and thus should be preassembled. | Modular concept |
| 2 | 3 | 3 Design | The Hatrack should visually integrate well with the cabin and the existing lining. | DesignAspect |
| 3 | 1 | 1 Accessibility OxygenMasks | The distance between the freely hanging Oxygen Masks and the Seat Reference Point incl. the shoulder height of the 5 Percentil asian woman shall not exceed 636.2mm. | distanceOxygenMask |

(a) Anforderungstabelle mit drei Beispielanforderungen.

| # | Id | Name | Applied Stereotype | Associated Actor | Top-Level Requirement | General Use Case |
|---|---|---|---|---|---|---|
| 1 | PSU-UC-AC-01 | Provide Passenger Individual Air Condition | «» FunctionalUseCase [UseCase]<br>«» PSU_UseCase [UseCase] | Passenger<br>ACS | TLCR.1 Air Outlets<br>FR.9 Individual Air | Provide Passenger Service Functionalities |
| 7 | PSU-UC-OX-01 | Provide Oxygen Outflow for Passengers | «» PSU_UseCase [UseCase]<br>«» SafetyUseCase [UseCase]<br>«» FunctionalUseCase [UseCase] | CMS<br>OxygenSystem<br>Passenger | TLCR.2 Oxygen Supply<br>FR.8 Oxygen Outflow | Provide Oxygen Masks for Crew and Passengers<br>Provide Passenger Service Functionalities |
| 15 | PSU-UC-CO-01 | Provide Call Flight Attendant functionality | «» FunctionalUseCase [UseCase]<br>«» PSU_UseCase [UseCase] | CMS<br>CabinCrew<br>Passenger | TLCR.3 Cabin Management System<br>FR.4 PAX Call | Provide Comfort<br>Provide Passenger Service Functionalities |

(b) Traceability-Tabelle für Anwendungsfälle eines Systems von [FGA+22].

Abbildung A.10: Tabellen für die Analyse von Anforderungen und Modellelementen.

Für die Erstellung von den Matrizen wird das SysML-Plugin benötigt [No 22a]. Mit Hilfe der Satisfy-Requirement-Matrix werden die Beziehungen zwischen den Anforderungen und den Modellelementen dargestellt. In den Spalten werden die Anforderungen dargestellt und in den Zeilen die Elemente (Abbildung A.11). Ein Pfeil symbolisiert eine Beziehung bzw. ordnet den Anforderungen die entsprechenden Elemente zu.

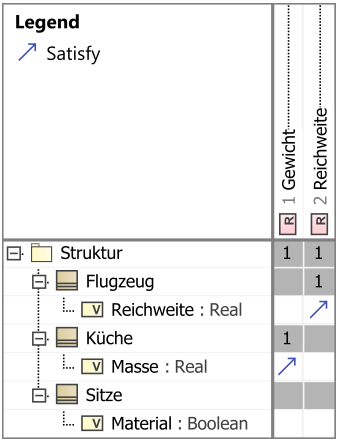



Abbildung A.11: Beispiel für eine Satisfy-Requirement-Matrix.

Sowohl mit den Tabellen als auch mit den Matrizen können die Beziehungen zwischen den Anforderungen und anderen Entwurfselementen erstellt, analysiert und geändert werden [No 22a]. Zusammenfassend können mit den Tabellen und Matrizen Anforderungen zu den SysML-Modellelementen zurückverfolgt und auf Vollständigkeit überprüft werden.

## A.3 Vorstellung der Matlabelemente

In diesem Abschnitt werden die beiden Elemente Klasse und Funktion vorgestellt. An zwei Beispielimplementierungen für Kabinenkomponenten werden der interne Aufbau und die Funktionsweise erklärt. Zudem wird das Funktionsskript für den Export der Parameter in die XML-Datei im Rahmen des Customizings vorgestellt.

### A.3.1 Aufbau einer Klasse in Matlab

Der folgende Code in Abbildung A.12 zeigt das Modellelement *Klasse* in Matlab. Anhand der Klasse *Seat* für die Kabinenkomponente Sitz sind einige Beispiel für Attribute und Methoden aufgeführt. Zusätzlich zu den spezifischen Eigenschaften (Pitch-Seat) erbt die Klasse Sitz weitere Attribute von der Basisklasse *Component*. Mit Hilfe der gezeigten Funktion obj = Seat() im Methodenabschnitt kann eine Instanz von der Klasse Seat erzeugt. Dabei werden die Parameter aus der Funktion den hinterlegten Attributen zugeordnet.

```
classdef Seat < Component
    %SEAT
    %Seat inherit properties like ID, name and type of class component
    %Seat has specific properties like mass_seat

    properties
        mass_seat
        PitchSeat = 0.762
        WidthSeat =0.4572
        AreaSeat
    end

    methods
        function obj = Seat(name,type,x,y,z,xcog,ycog,zcog,length,width,height,mass_seat)
            %SEAT Construct an instance of this class
                obj.name = name;
                obj.type = type;
                obj.x = x;
                obj.y = y;
                obj.z = z;
                obj.xcog = xcog;
                obj.ycog = ycog;
                obj.zcog = zcog;
                obj.length = length;
                obj.width = width;
                obj.height = height;
                obj.mass_seat = mass_seat;
        end

    end
end
```

Abbildung A.12: Matlab-Code für eine Objektklasse am Beispiel der Kabinenkomponente Sitz.

### A.3.2 Beispiel für ein Funktionsskript in Matlab

Der Matlabcode in Abbildung A.13 zeigt den Aufbau eines Funktionsskript für die Platzierung der Verkleidung (Lining). Eine Funktion besteht aus einem Output, dem Funktionsnamen und dem Input. Der Funktionsname des gezeigten Beispiels ist *placeLining()*. Der Input sind die Objektinstanzen für die Sitze, die Spanten und die Notausgänge. Die Attribute der Instanzen werden als Parameter für den Aufbau des Bauraums verwendet, da z.B. die Seitenwandpaneele an den Spanten befestigt werden müssen. Der Output sind Objektinstanzen für die Verkleidungskomponenten. Dies umfasst die Seitenwandpaneele (Sidewall), die Fußraumverkleidung bzw. das Dadopanel, die Lichtabdeckung (Light Cover) und die Deckenwandpaneele (Ceiling). In dem Funktionsskript werden zuerst die Parameter der Input-Objektinstanzen ausgelesen und daraus die erforderliche Anzahl der Verkleidungselemente kalkuliert. Anschließend erfolgt die parametrische Platzierung und Instanziierung der Komponenten.

```
function [sidewall,dadopanels,lightCovers,ceilings] = placeLining(seats,frames,exitDoors)
%PLACESIDEWALL Place the Sidewall in an area
global params
for i=1:length(seats)
    dummy(1,i) = seats(1,i).Midpoint_x;
end
[x,idx] = max(dummy);
xEnd = seats(1,idx).Midpoint_x+seats(1,idx).Length*0.5*params.i2mm;
for i=1:length(frames)
    dummy2(1,i) = frames(1,i).Midpoint_x;
    dummy3(1,i) = frames(1,i).Midpoint_x-params.cabin.xStart;
    dummy4(1,i) = frames(1,i).Midpoint_x-(exitDoors(1,1).Midpoint_x+exitDoors(1,1).Length*0.5*params.i2mm);
end
%find closest frame as xStart/xLauf var after exit
%[x,idx] = min(dummy3);
x = min(dummy4(dummy4>0));
idx = find(dummy4(1,:)==x);
xLauf = dummy2(1,idx);
sidewallNumber = round((xEnd-xLauf)/(params.sidewall.length*params.i2mm)) + 1;
```

```
zStart = params.cabin.zMidpoint;
yStart = sqrt((params.cabin.radius)^2-(zStart)^2); %in mm
%Sidewalls
sidewall = [];
for j = 1:sidewallNumber
    %last Sidewall needs to be shortened
    if j==sidewallNumber && xLauf + params.sidewall.length*params.i2mm > xEnd + 100
        newLength = (xEnd+100-xLauf)*params.mm2i;
        v1 = Sidewall(newLength,params.sidewall.width,params.sidewall.height,xLauf+0.5*newLength*params.i2mm,...
            yStart-0.5*params.sidewall.width*params.i2mm,zStart+0.28*params.sidewall.height*params.i2mm,j,'RH');
        v2 = Sidewall(newLength,params.sidewall.width,params.sidewall.height,xLauf+0.5*newLength*params.i2mm,...
            -yStart+0.5*params.sidewall.width*params.i2mm,zStart+0.28*params.sidewall.height*params.i2mm,j,'LH');
        sidewall = [sidewall,v1];
        sidewall = [sidewall,v2];
    else
        v1 = Sidewall(params.sidewall.length,params.sidewall.width,params.sidewall.height,xLauf+0.5*params.sidewall....
            length*params.i2mm,...
            yStart-0.5*params.sidewall.width*params.i2mm,zStart+0.28*params.sidewall.height*params.i2mm,j,'RH');
        v2 = Sidewall(params.sidewall.length,params.sidewall.width,params.sidewall.height,xLauf+0.5*params.sidewall....
            length*params.i2mm,...
            -yStart+0.5*params.sidewall.width*params.i2mm,zStart+0.28*params.sidewall.height*params.i2mm,j,'LH');
        sidewall = [sidewall,v1];
        sidewall = [sidewall,v2];
    end
    xLauf = xLauf + params.sidewall.length*params.i2mm;
    if j ==1
       params.deuA.xStart = xLauf;
    end
end
%Dadopanels
dadopanels = [];
xLauf = dummy2(1,idx);
for j = 1:sidewallNumber
    %last Dadopanel needs to be shortened
    if j==sidewallNumber && xLauf + params.dadopanel.length*params.i2mm > xEnd + 100
        newLength = (xEnd+100-xLauf)*params.mm2i;
        v1 = Dadopanel(newLength,params.dadopanel.width,params.dadopanel.height,xLauf+0.5*newLength*params.i2mm,...
            yStart-0.5*params.dadopanel.width*params.i2mm,0.5*params.dadopanel.height*params.i2mm,j,'RH');
        v2 = Dadopanel(newLength,params.dadopanel.width,params.dadopanel.height,xLauf+0.5*newLength*params.i2mm,...
            -yStart+0.5*params.dadopanel.width*params.i2mm,0.5*params.dadopanel.height*params.i2mm,j,'LH');
        dadopanels = [dadopanels,v1];
        dadopanels = [dadopanels,v2];
    else
        v1 = Dadopanel(params.dadopanel.length,params.dadopanel.width,params.dadopanel.height,xLauf+0.5*params....
            dadopanel.length*params.i2mm,...
            yStart-0.5*params.dadopanel.width*params.i2mm,0.5*params.dadopanel.height*params.i2mm,j,'RH');
        v2 = Dadopanel(params.dadopanel.length,params.dadopanel.width,params.dadopanel.height,xLauf+0.5*params....
            dadopanel.length*params.i2mm,...
            -yStart+0.5*params.dadopanel.width*params.i2mm,0.5*params.dadopanel.height*params.i2mm,j,'LH');
        dadopanels = [dadopanels,v1];
        dadopanels = [dadopanels,v2];
    end

    xLauf = xLauf + params.dadopanel.length*params.i2mm;
end
%lightCovers
lightCovers = [];
xLauf = dummy2(1,idx);
%y and z calculating between Sidewall and PSC
yMidpoint = yStart-0.5*params.lightCover.width*params.i2mm;
zMidpoint = 0.5*params.lightCover.height*params.i2mm+sidewall(1,1).Midpoint_z+params.sidewall.height*params.i2mm*0.5;
for j = 1:sidewallNumber
    if j==sidewallNumber && xLauf + params.lightCover.length*params.i2mm > xEnd + 100
        newLength = (xEnd+100-xLauf)*params.mm2i;
        v1 = LightCover(newLength,params.lightCover.width,params.lightCover.height,xLauf+0.5*newLength*params.i2mm,...
            yMidpoint,zMidpoint,j,'RH');
        v2 = LightCover(newLength,params.lightCover.width,params.lightCover.height,xLauf+0.5*newLength*params.i2mm,...
            -yMidpoint,zMidpoint,j,'LH');
        lightCovers = [lightCovers,v1];
        lightCovers = [lightCovers,v2];
    else
        v1 = LightCover(params.lightCover.length,params.lightCover.width,params.lightCover.height,xLauf+0.5*params....
            lightCover.length*params.i2mm,...
            yMidpoint,zMidpoint,j,'RH');
        v2 = LightCover(params.lightCover.length,params.lightCover.width,params.lightCover.height,xLauf+0.5*params....
            lightCover.length*params.i2mm,...
            -yMidpoint,zMidpoint,j,'LH');
        lightCovers = [lightCovers,v1];
        lightCovers = [lightCovers,v2];
    end
    xLauf = xLauf + params.lightCover.length*params.i2mm;
end
%Ceiling
ceilings = [];
yMidpoint = 0;
zMidpoint = 2203.3; %mm
ceilingNumber = 33;
xLauf = params.cabin.xStart;
for j = 1:ceilingNumber
    v1 = Ceiling(params.ceiling.length,params.ceiling.width,params.ceiling.height,xLauf+0.5*params.ceiling.length*...
        params.i2mm,...
        yMidpoint,zMidpoint,j,'RHLH');
    ceilings = [ceilings,v1];

    xLauf = xLauf + params.ceiling.length*params.i2mm;
end
end
```

Abbildung A.13: Matlab-Code für ein Funktionsskript am Beispiel der Verkleidungsplatzierung.

### A.3.3 Funktionsskript in Matlab für den Parameterexport beim Customizing

Beim Customizing wird die Kabinenauslegung einer untersuchten Flugzeugvariante geändert. Zum Ende der Auslegung der neuen Kabinenkonfiguration werden die Ergebnisparameter wieder in eine XML-Datei exportiert. Dafür werden sowohl neue Parameter für den Baumknoten *cabin* hinzugefügt, als auch Baumknoten weiterer Subsysteme aus der Basisvariante verwendet. Das Customizing bezieht sich lediglich auf die Kabinenkomponenten, wodurch keine Änderung der anderen Subsysteme angedacht ist. Das zugehörige Funktionsskript ist in einer verkürzten Form in Abbildung A.14 dargestellt. Zuerst wird die als Ausgangsbasis dienende XML-Datei der Variante als Strukturarray (adapterData) eingelesen. Anschließend wird eine neue Baumstruktur für die XML-Datei der Customizingvariante erzeugt (Zeilen 13ff.). Anschließend werden die neuen Parameter der Kabinenkomponenten an den Baumknoten angehängt. Dabei werden über die Collection auf die erzeugten Objektinstanzen zugegriffen und die gewünschten Parameter wie ID, Länge oder Name ausgelesen. Ab Zeile 19 ist dies verkürzt am Beispiel des Parameters ID dargestellt. Im letzten Schritt werden alle weiteren Baumknoten aus der Basisvariante ausgelesen, die nicht den Namen Cabin tragen und dessen Parameter ebenfalls in die neue XML-Baumstruktur übertragen.

```
1  function exportAdapter(xmlAdapterName)
2  global collection
3  global params
4  global totalPowerCabin

6  %read data from variant xml via structure
7  rng(0,'twister');
8  docNode = com.mathworks.xml.XMLUtils.createDocument('Master');
9  root = docNode.getDocumentElement;
10 root.setAttribute('version', '1.0');
11 adapterData = xml2struct(xmlAdapterName);

13 %create new xml
14 header = docNode.createElement('header');
15 root.appendChild(header);
16 aircraft = docNode.createElement('aircraft');
17 root.appendChild(aircraft);

19 %add new cabin parameters as new tree knot
20 cabin_parent = aircraft.appendChild(docNode.createElement("cabin"));
21 curr_parent = cabin_parent.appendChild(docNode.createElement("components"));
22 collectionKeys = collection.keys;

24 for i=1:length(collectionKeys)
25     tempKey = string(collectionKeys(i));
26     tempObj = collection(tempKey);

28     if isa(tempObj, "Seat") == 1  isa(tempObj, "Galley") == 1  isa(tempObj, "Lavatory") == 1

30         new_node = docNode.createElement('component');
31         curr_node = new_node.appendChild(docNode.createElement('ID'));
32         curr_node.appendChild(docNode.createTextNode(num2str(tempObj.ID)));
33         ...
34         curr_parent.appendChild(new_node);
35   end
36 end

38 %% find other systems data from old xml and write to new XML
39 subsystemNames = fieldnames(adapterData.Master.aircraft);

41  for i = 1:length(fieldnames(adapterData.Master.aircraft))-3
42      subsystemName = convertCharsToStrings(subsystemNames{i+3,1});
43      subsystem_parent = aircraft.appendChild(docNode.createElement(subsystemName));
44      curr_parent = subsystem_parent.appendChild(docNode.createElement("parameters"));
```

```
1       %components
2       curr_parent = subsystem_parent.appendChild(docNode.createElement('components'));
3       for l = 1:length(adapterData.Master.aircraft.(subsystemName).components.component)
4           curr_node2 = curr_parent.appendChild(docNode.createElement('component'));
5           for k = 1:length(fieldnames(adapterData.Master.aircraft.(subsystemName).components.component{1,l}))
6               nameFields = convertCharsToStrings(fieldnames(adapterData.Master.aircraft.(subsystemName).components....
                    component{1,l}));
7               a = nameFields(k,1);
8               b = convertCharsToStrings(adapterData.Master.aircraft.(subsystemName).components.component{1,l}.(...
                    nameFields(k,1)).Text);
9               curr_node3 = curr_node2.appendChild(docNode.createElement(a));
10              curr_node3.appendChild(docNode.createTextNode(b));

12              curr_parent.appendChild(curr_node2);
13          end
14      end
15  end
16  ...

18  xmlwrite('XML_AdapterNew.xml',docNode);

20  end
```

Abbildung A.14: Verkürzter Matlab-Code für das Funktionsskripts zur Erzeugung einer neuen XML-Datei nach dem Customizing (XML_AdapterNew.xml) mit Übertragung von Parametern aus der Basisvariante (xmlAdapterName).

## A.4 Dateien für die Visualisierung einer Flugzeugkonfiguration

In diesem Abschnitt werden die Python-Steuerungsdatei für die automatische Ansteuerung des Blender-Modells für die 3D Visualisierung und die JSON-Konfigurationsdatei für die Zuordnung der detaillierten 3D Einzelmodelle für eine Visualisierung der Flugzeugkonfiguration vorgestellt. Zudem wird ein Einblick in den XML-Dateiaufbau für die Matlab-Blender-Anbindung gegeben. Aus dieser werden Informationen für die 3D Visualisierung der einzelnen Flugzeugkabinenkomponenten ausgelesen, die nicht im XML-Adapter hinterlegt sind.

### A.4.1 Python-Steuerungsdatei für die Benutzeroberfläche in Blender

Die automatische Ansteuerung der Benutzeroberfläche in Blender erfolgt über ein Python-Skript. Abbildung A.15 gibt einen verkürzten Einblick in den ersten Teil des Python-Codes. In dem gezeigten Abschnitt wird das ElementTree-Packet verwendet, um die XML-Datei mit den Parametern für die 3D Visualisierung aus der Matlab-Blender-Schnittstelle auszulesen. In Zeile 33 ff. wird die JSON-Konfigurationsdatei für die Zuordnung der 3D Einzelmodelle zu den Kabinenkomponenten eingeladen. Die Zuordnung der XML-Datei (output.xml) findet in den Zeilen 39 ff. statt. In den Zeilen 45 und 46 werden der Speicherort für das generierte 3D-Modell definiert und das Dateiformat fbx zugewiesen.

Der zweite Abschnitt in Abbildung A.16 zeigt den Bereich, in dem die Zuordnung der Werte aus der XML-Datei zu den 3D Einzelmodellen erfolgt und die Dimensionen angepasst werden. Für eine bessere Übersicht wurde der Abschnitt ab Zeile 28 für die Erzeugung eines einfachen geometrischen Objekts ausgeblendet. Die Zuordnung der

Werte und die Anpassung der geometrischen Maße erfolgt ähnlich zu der vorgestellten Methode bei den detaillierten fbx-Objekten.

```
1  import xml.etree.ElementTree as etree
2  import os
3  import bpy
4  import json
5  import math
6  import pathlib
7  from shutil import copyfile
8  import sys

10 # fetch arguments from command line and make them accessible
11 try:
12     args = list(reversed(sys.argv))
13     idx = args.index("--")
14 except ValueError:
15     params = []
16 else:
17     params = args[:idx][::-1]
18 # if -d argument is given, save decimateRatio
19 decimateRatio = False
20 fileFormat = "fbx"
21 createFuselage = True

23 for i in range(len(params)):
24     if params[i] == "-d":
25         decimateRatio = float(params[i + 1])
26     if params[i] == "-f":
27         fileFormat = params[i + 1]
28     if params[i] == "-fuselage":
29         if params[i + 1] == "false":
30             createFuselage = False

32 # load config from json-file
33 configFile = os.path.expanduser("config.json")

35 with open(configFile) as json_file:
36     config = json.load(json_file)

38 # load xml-file and parse elements
39 xmlFile = "output.xml"
40 xml_root = etree.parse(xmlFile).getroot()

42 # set filepath for blender
43 export_path = os.path.expanduser("./Generated Files/Cabin." + fileFormat)
44 pathlib.Path("Generated Files/").mkdir(parents=True, exist_ok=True)
```

Abbildung A.15: Python-Code für die Ansteuerung der Benutzeroberfläche in Blender - Teil 1.

In Zeile 1 wird ein Befehl ausgeführt, um alle instanziierten Flugzeugkabinenkomponenten in der XML-Datei unter dem Baumknoten *component* zu finden. Für jedes Element werden dann die folgenden Schritte ausgeführt. Zuerst werden die Werte für bestimmte Attribute ausgelesen. Dazu gehören die x-, y- und z-Positionen, die Länge, die Breite, die Höhe, die ID und der Typ. Im nächsten Schritt wird die JSON-Konfigurationsdatei angesteuert und geschaut, ob ein Eintrag für die Zuordnung des aktuellen Kabinenkomponentenobjekts hinterlegt ist (siehe Abschnitt A.4.2). Anschließend wird entsprechend des Eintrags oder wenn keiner gefunden wurde entsprechend der Standardeinstellung die Objektform (meshtype) ausgelesen. In den Zeilen 31 ff. wird die Anpassung eines 3D Objekts für einen gefundenen Eintrag und die Objektform fbx durchgeführt. Dabei wird zuerst das entsprechende 3D Einzelmodell aus der hinterlegten Objektbibliothek importiert und an seiner vorgesehenen Position platziert. Da die Einheit in Blender Meter ist und im Matlab-Modell die Einheit Millimeter verwendet wird, werden die x-, y- und z-Werte durch den Faktor 1000 geteilt (Zeile 35). Wenn eine Anpassung des 3D Objekts durchgeführt werden soll, werden in den Zeilen 44 ff. die Werte für die Länge, Höhe und Breite in Meter umgerechnet, da diese im Matlab-Modell in der Einheit inch abgespeichert wurden. Der Grund hierfür ist, dass in der Luftfahrt viele Angaben in der Einheit inch angegeben sind. In den Zeilen 61 ff. wird, sofern Werte für eine zusätzliche Verschiebung oder Rotation in der Konfi-

```
componentItems = xml_root.findall(".//component")
# create cabin elements
for i in componentItems:
    itemType = i.findtext("Type")
    x = i.findtext("X")
    y = i.findtext("Y")
    z = i.findtext("Z")
    length = i.findtext("Length")
    width = i.findtext("Width")
    height = i.findtext("Height")
    id = i.findtext("ID")

    # load config for elements and check conditions if needed
    if itemType in config:
        itemConfig = config[itemType]["default"]
        if "check" in config[itemType]:
            for case in config[itemType]["check"]:
                condition = case["condition"]
                conditionField, conditionValue = condition.split("=")
                if i.find(conditionField) is not None:
                    if i.findtext(conditionField) == conditionValue:
                        itemConfig = case
                        break
    else:
        itemConfig = config["default"]

    # create cube elements
    if itemConfig["meshType"] == "cube": ...

    # import fbx elements
    elif itemConfig["meshType"] == "fbx":
        bpy.ops.import_scene.fbx(filepath=itemConfig["meshSource"])
        bpy.ops.object.origin_set(type="ORIGIN_CURSOR")
        bpy.ops.transform.translate(
            value=(float(x) / 1000, float(y) / 1000, float(z) / 1000)
        )
        if decimateRatio:
            obj = bpy.context.selected_objects[0]
            dm = obj.modifiers.new("Decimate", "DECIMATE")
            dm.ratio = decimateRatio
        bpy.context.selected_objects[0].name = id
        bpy.context.selected_objects[0].data.name = id
        # if adjustSize is true in config, size of fbx is adjusted according to object data
        if "adjustSize" in itemConfig:
            if itemConfig["adjustSize"]:
                # adjust size based on data from Matlab
                if itemType == "SupplyDuct":
                    bpy.context.selected_objects[0].dimensions = (
                        float(height) / 1000 * 25.4,
                        float(width) / 1000 * 25.4,
                        float(length) / 1000 * 25.4,
                        )
                else:
                    bpy.context.selected_objects[0].dimensions = (
                        float(length) / 1000 * 25.4,
                        float(width) / 1000 * 25.4,
                        float(height) / 1000 * 25.4,
                    )
        ov=bpy.context.copy()
        ov['area']=[a for a in bpy.context.screen.areas if a.type=="VIEW_3D"][0]
        if "transform" in itemConfig:
            if itemConfig["transform"] is not None:
                for transform in itemConfig["transform"]:
                    if transform["transformType"] == "rotation":
                        angle = transform["rotationAngle"] * math.pi / 180
                        axis = transform["rotationAxis"]
                        bpy.ops.transform.rotate(ov, value=angle, orient_axis=axis)
                    if transform["transformType"] == "translation":
                        if "translationX" in transform:
                            translationX = transform["translationX"] / 1000
                        else:
                            translationX = 0
                        if "translationY" in transform:
                            translationY = transform["translationY"] / 1000
                        else:
                            translationY = 0
                        if "translationZ" in transform:
                            translationZ = transform["translationZ"] / 1000
                        else:
                            translationZ = 0
                        bpy.ops.transform.translate(
                            value=(translationX, translationY, translationZ)
                        )
    bpy.context.object.name = id
    bpy.context.object.data.name = id
```

Abbildung A.16: Python-Code für die Ansteuerung der Benutzeroberfläche in Blender -Teil 2.

gurationsdatei hinterlegt sind, das Objekt transformiert. Zuletzt wird der Name des 3D Einzelobjektes mit der ID überschrieben. Dadurch ist im 3D Modell eine eindeutige Identifizierung der 3D Objekte mit den Parametern in den XML-Dateien gewährleistet.

Damit ist die Platzierung und Anpassung eines detaillierten 3D Objekts abgeschlossen und der Prozess beginnt für den nächsten Objekteintrag von vorne.

### A.4.2 JSON-Konfigurationsdatei für die Zuordnung der 3D Einzelmodelle

Der JSON-Quellcode in Abbildung A.17 zeigt einen Ausschnitt der Konfigurationsdatei für die Zuordnung der Kabinenkomponenten zu den 3D Einzelmodellen für ein Verkehrsflugzeug. Ein detaillierter Einblick in die Zuordnung ist in den Zeilen 54 bis 94 für die Gepäckablage gegeben. In der Python-Steuerungsdatei wird in der XML-Datei nach dem Type des aktuellen instanziierten Objekts gesucht. Stimmt dieser mit der Bezeichnung *OHSC* überein, wird ein Check durchgeführt. Eine Konditionsabfrage überprüft den Wert des Objektattributs Spec. Dieser gibt ab, auf welcher Flugzeugseite sich das Objekt befindet. Aufgrund der Symmetrie des Flugzeugs können viele Komponenten in ihrer Ausrichtung gespiegelt werden, wodurch die detaillierten 3D Objekte nicht doppelt hinterlegt werden müssen. Stimmen die Werte mit der Bezeichnung *RH* (right hand, rechte Flugzeugseite) überein, wird die nachfolgende Zuordnung durchgeführt. Dabei werden die Größe des Objekts angepasst, das 3D Modell ohscENL.fbx als Objekt ausgewählt und eine Transformation durchgeführt. Letzteres wird durchgeführt, da die detaillierten 3D Modelle im Vergleich zu den einfachen Geometrien im Matlab-Modell eine leicht verschobene Position ihres geometrischen Mittelpunkt haben. Stimmen die Werte mit der Spec nicht überein, wird die Standardeinstellung verwendet. Dabei wird ebenfalls eine Transformation des 3D Objekts inkl. einer Rotation des Objekts durchgeführt und die Größe angepasst. Für alle anderen Objekttypen verhält sich die Zuordnung identisch. Ist ein Objekttyp nicht explizit aufgeführt, wird diese in der Standardeinstellung *default* als Rechteck (cube) dargestellt (Zeilen 1125 ff.).


Abbildung A.17: JSON-Code für die Zuordnung der 3D Einzelmodelle zu den Kabinenkomponenten in Blender.

### A.4.3 XML-Dateiaufbau für die Matlab-Blender-Kopplung

Wie bereits in den vorherigen Abschnitten vorgestellt, wird als Grundlage für die 3D Visualisierung der Kabinenkonfigurationen eine XML-Datei mit den Parametern aus der Auslegung mit dem Matlab-Modell verwendet. Abbildung A.18 gibt einen Einblick in den Aufbau der XML-Datei am Beispiel der generierten Datei für Variante 1b-1. Diese orientiert sich an dem Aufbau der XML-Adapterdatei für die Kopplung der Partialmodelle. Allerdings enthält diese Datei weitere Informationen über die Kabinenobjekte, die für eine Visualisierung bzw. die Platzierung detaillierter 3D Einzemodelle benötigt werden. Die Datei wird bei jeder Variante mit der Bezeichnung output.xml versehen. Unter dem Baumknoten *parameters* sind allgemeine Parameter hinterlegt, die z.B. für die Positionierung einer Flugzeughülle verwendet werden. Der Baumknoten *components* enthält alle instanziierten Kabinenobjekte. Jedes Objekt erhält einen eigenen Eintrag. Unter dem Baumknoten *component* werden die zugehörigen Attribute und deren Werte hinterlegt. Diese umfassen die ID, den Typ, den Namen, die Abmaße, die Position, das Material und das ATA-Chapter. Zusätzlich werden die Attribute Spec (Specification) und Tag abgespeichert. Mit der Spec wird angegeben, auf welcher Seite der Flugzeugkabine die Komponente angeordnet ist. Die Bezeichnungen RH und LH definieren jeweils die rechte oder linke Flugzeugseite, während RHLH eine mittige bzw. seitenunabhängige Platzierung klassifiziert. Dadurch können für die symmetrische Anordnung die identischen 3D Einzelmodelle für die Platzierung auf der rechten oder linken Flugzeugseite, unter Zuhilfenahme der Rotation, genutzt werden. Mit dem Attribut Tag wird die zugehörige übergeordnete Systemgruppe angegeben. Damit können für Darstellungen z.B. alle Komponenten einer Subsystemgruppe farblich gekennzeichnet und hervorgehoben werden.

```xml
<?xml version="1.0" encoding="utf-8"?>
<cabin version="2.0">
  <parameters>
    <aircraft>
      <type>A320</type>
      <xStart>2540</xStart>
      <length>37573.2</length>
    </aircraft>
    <cabin>
      <xStart>5794</xStart>
      <yStart>0</yStart>
      <zStart>0</zStart>
      <dia>3680</dia>
      <radius>1885</radius>
      <zMidpoint>650</zMidpoint>
      <length>25986</length>
      <rowsBC>0</rowsBC>
      <rowsEC>27</rowsEC>
      <nPaxBc>0</nPaxBc>
      <nPaxEc>162</nPaxEc>
      <seatPerRowsEC>6</seatPerRowsEC>
      <seatPerRowsBC>0</seatPerRowsBC>
      <rows>27</rows>
      <aisleNumber>1</aisleNumber>
      <aisleWidth>481</aisleWidth>
      <passenger>162</passenger>
    </cabin>
  </parameters>
  <components>
    <component>
      <ID>10022154</ID>
      <Type>fillerPanel</Type>
      <Name>FillerPanel68RH</Name>
      <Length>19</Length>
      <Width>5.4</Width>
      <Height>1.6</Height>
      <X>12877.8</X>
      <Y>948.58</Y>
      <Z>1670.0009</Z>
      <Spec>RH</Spec>
      <Ata>25-25-11</Ata>
      <Material>PVC</Material>
      <Tag>lining</Tag>
    </component>
```

Abbildung A.18: Aufbau der XML-Datei von Variante 1b-1 für die 3D Visualisierung der Flugzeugkabine in Blender.

## A.5 Programmcode des Elements Opaque Action in den SysML-Modellen

Im folgenden Abschnitt sind die Programmcodes der einzelnen Opaque Aktionen aufgeführt. Diese werden in den SysML-Modellen für das Einlesen, das Schreiben und

Verändern der XML-Adapterdatei-Parameter verwendet.

### A.5.1 Einlesen der Parameter aus der XML-Datei

Der Quellcode in Abbildung A.19 zeigt das Auslesen der Parameter aus der XML-Datei. Programmiert ist der Code in Python. Zuerst wird die XML-Datei eingelesen. Dafür kommt das Modul ElementTree (ET) zum Einsatz, welches eine einfache und effiziente API zum Parsen und Erstellen von XML-Daten bietet[19]. Anschließend wird innerhalb der Baumstruktur nach allen Powerkomponenten gefiltert und nach dem Typ jeder Komponente gesucht. Je nach Typ der Komponente, werden unterschiedliche Werte für die Länge des Wasserstofftanks und die Leistung rausgesucht. Final werden die beiden Parameter zusammen mit dem Typ der Powerkomponente an die Instanz im Modell vom Typ Aircraft übergeben.

```
#lesen der xml
import xml.etree.ElementTree as ET
Filename='D:/HSU/Doktorarbeit_Studie_IndustriellerKontext/Matlab_und_Cameo/XML_Adapter.xml'
tree = ET.parse(Filename)
xml_root = tree.getroot()

#findet nur Powerkomponenten
test = xml_root.findall(".//powersystem//component")

for i in test:
  type_xml = i.findtext("type")
  print(type_xml)
  if type_xml == "HydrogenTank":
    xLength_xml = i.findtext("length")
    print(xLength_xml)
    obj1 = ALH.getValue(input, "Aircraft")
    ALH.setValue(obj1,"LengthHydrogenTank",xLength_xml)
    ALH.addValue(self,"Aircraft",obj1)
    ALH.setValue(obj1,"PowerComponentType","HydrogenTank")
    ALH.addValue(self,"Aircraft",obj1)
  else:
    obj1 = ALH.getValue(input, "Aircraft")
    ALH.setValue(obj1,"LengthHydrogenTank",0)
    ALH.addValue(self,"Aircraft",obj1)

for i in test:
  type_xml = i.findtext("type")
  if type_xml == "FuelCell":
    power_xml = i.findtext("powerfc")
    obj1 = ALH.getValue(input, "Aircraft")
    ALH.setValue(obj1,"Power",power_xml)
    ALH.addValue(self,"Aircraft",obj1)

for i in test:
  type_xml = i.findtext("type")
  if type_xml == "APU":
    power_xml = i.findtext("power")
    obj1 = ALH.getValue(input, "Aircraft")
    ALH.setValue(obj1,"Power",power_xml)
    ALH.addValue(self,"Aircraft",obj1)
    ALH.setValue(obj1,"PowerComponentType","APU")
    ALH.addValue(self,"Aircraft",obj1)
```

Abbildung A.19: Python-Code für das Auslesen von Parametern aus einer XML-Datei.

### A.5.2 Export der Parameter in die XML-Datei

Für den Export der Parameter aus den erzeugten Instanzen im SysML-Modell des Cameo Systems Modeler in die XML-Datei werden zwei Opaque Aktionen definiert. Die

[19] https://docs.python.org/3/library/xml.etree.elementtree.html

zwei dargestellten Programmcodes zeigen jeweils ein Minimalbeispiel für die Komponenten der Systemgruppe Flugzeugkabine.

### Opaque Aktion createNewComponent(inputValue)

Die erste Aktion mit dem Namen createNewComponent ermöglicht das Auslesen der Instanzparameter und der anschließenden Erzeugung eines neuen Elements in der XML-Datei. Abbildung A.20 zeigt den zugehörigen Python-Code. Auf gleicher Weise wie beim Import wird die XML-Datei mit dem Modul ElementTree(ET) ausgelesen. Anschließend wird die Baumstruktur durchsucht, bis die passende Ebene für die Zuordnung der Systemkomponenten gefunden wird. Für jedes Partialmodell, welches ein Subsystem beschreibt, wird ein eigener zugewiesener Baumknoten verwendet. Im gezeigten Beispiel wird für die Kabinenkomponenten der Knoten .//cabin//components gesucht. Im letzten Schritt wird ein neuer Knoten erzeugt, mit den Parametern aus der eingelesenen Instanz (inputValue) aufgefüllt und abschließend als neues *component*-Element an den Baumknoten *components* angehängt. Der Prozess wird für alle Instanzen vom selben Typ wiederholt durchlaufen.

```
#lesen der Instanz
count1= len(inputValue)
print(count1)

import xml.etree.ElementTree as ET
Filename='D:/HSU/Doktorarbeit_Studie_IndustriellerKontext/Matlab_und_Cameo/XML_Adapter.xml'
tree = ET.parse(Filename)
xml_root = tree.getroot()

#schreibt neues "component"
for i in range(count1):
  loc_path = xml_root.findall(".//cabin//components")
  new_node=ET.Element("component")

  type_cameo = ET.Element("type")
  new_node.append(type_cameo)
  new_node.find('type').text = str(inputValue[i].get("Type"))
  id_cameo = ET.Element("ID")
  new_node.append(id_cameo)
  new_node.find('ID').text = str(inputValue[i].get("ID"))
  mass_cameo = ET.Element("mass")
  new_node.append(mass_cameo)
  new_node.find('mass').text = str(int(inputValue[i].get("Mass")))
  xLocal_cameo = ET.Element("x")
  new_node.append(xLocal_cameo)
  new_node.find('x').text = str(inputValue[i].get("x"))
  yLocal_cameo = ET.Element("y")
  new_node.append(yLocal_cameo)
  new_node.find('y').text = str(inputValue[i].get("y"))
  zLocal_cameo = ET.Element("z")
  new_node.append(zLocal_cameo)
  new_node.find('z').text = str(inputValue[i].get("z"))
  length_cameo = ET.Element("length")
  new_node.append(length_cameo)
  new_node.find('length').text = str(inputValue[i].get("Length"))
  width_cameo = ET.Element("width")
  new_node.append(width_cameo)
  new_node.find('width').text = str(inputValue[i].get("Width"))
  height_cameo = ET.Element("height")
  new_node.append(height_cameo)
  new_node.find('height').text = str(inputValue[i].get("Height"))

  for subElement in loc_path:
    subElement.append(new_node)
tree.write(Filename)
```

Abbildung A.20: Python-Code für die Opaque Aktion createNewComponent(inputValue).

**Opaque Aktion overwriteValueXML(inputValue)**

Die zweite Aktion mit dem Namen overwriteValueXML ermöglicht die Suche eines bestimmten Baumknotens und das anschließende Überschreiben des vorhandenen Werts in der XML-Datei. Abbildung A.21 zeigt den Python-Code. Zuerst wird die XML-Datei eingelesen. Anschließend wird der Baumknoten gesucht, in dem die Werte überschrieben werden sollen. In dem gezeigten Minimalbeispiel werden durch das Partialmodell der Kabine die beiden allgemeingültigen Kabinenparameter Länge und Anzahl der Passagiere überschrieben. Anschließend erfolgt in einem weiteren Schritt das Hinzufügen des Partialmodellnamens in den Header. Dadurch ist eine Rückverfolgbarkeit der ausgeführten Partialmodelle möglich. Im Header werden nach erfolgreicher Ausführung die Namen der Partialmodelle hinterlegt, mit denen die Daten in der XML-Datei generiert wurden.

```
#lesen der XML-Datei
import xml.etree.ElementTree as ET
Filename='D:/HSU/Doktorarbeit_Studie_IndustriellerKontext/Matlab_und_Cameo/XML_Adapter.xml'
tree = ET.parse(Filename)
xml_root = tree.getroot()

#lesen der Passagierinstanz
paxInstance = ALH.getValue(inputValue, "pax")

test2 = xml_root.findall(".//cabin//parameters")
for i in test2:
  x = i.findtext("length")
  i.find('length').text = str(inputValue.get("Length"))
  x = i.findtext("passenger")
  i.find('passenger').text = str(paxInstance[0].get("NumberofPax"))
  x = i.findtext("power")
  i.find('power').text = str(inputValue.get("ElectricalPower"))
tree.write(Filename)

#schreibt Partialmodellnamen in den Header
loc_path2 = xml_root.findall(".//header")
modelName=ET.Element("datasource")
modelName.text = str("cabinSystem")
for subElement in loc_path2:
  subElement.append(modelName)

tree.write(Filename)
```

Abbildung A.21: Python-Code für die Opaque Aktion overwriteValueXML(inputValue).

## A.6 Anwendung der Methodik im industriellen Kontext

In diesem Abschnitt werden Artefakte aus der Anwendung der Methodik im industriellen Kontext vorgestellt. Dazu gehören die Exceltabelle mit den Anforderungen an die Flugzeugkabine, die beiden Partialmodelle für das Wasserstofftanksystem und das Brennstoffzellensystem sowie die XML-Dateien der Fallstudien. Zusätzlich werden die Messungen der Prozesszeiten und die ermittelten Mittelwerte und Standardabweichungen vorgestellt.

### A.6.1 Ausschnitt der Exceltabelle mit den Anforderungen

Abbildung A.22 zeigt einen Ausschnitt der Exceltabelle, die bei der Durchführung der Fallstudien verwendet wurde. Die Liste enthält Vorgaben oder Annahmen an das Design sowie Sicherheitsregularien und Vorschriften aus der CS-25.

| ID | Name | Text | Documentation | Target | Unit | Classification | Date |
|---|---|---|---|---|---|---|---|
| 0 | **Requirements Aircraft Level** | | | | | | |
| 0.1 | **Top Level Aircraft Requirements** | | | | | | |
| 0.1.2 | Design range | Design range shall be at least 2500 nm | below A320 family | 2500 | nm | SHALL | 2023-02-02 |
| 0.1.5 | Cruise mach number | Cruise mach number shall be 0,78 | A320 AC-AAMP | 0,78 | | SHALL | 2023-02-02 |
| 0.1.6 | max. operating Mach number | Max. operating mach number shall be cruise mach number +0.02 | | 0,78+0,02 | | SHALL | 2023-02-02 |
| 0.1.7 | Max. operating altitude | Max. operating altitude shall be 40000 ft. | | 40000 | feet | SHALL | 2023-02-02 |
| 0.1.8 | Take-Off Field Length TOFL@ ISA+0, SL | TOFL @ISA+0, SL shall be shorter than 2200m | A320 AC-AAMP | <2200 | m | SHALL | 2023-02-02 |
| 0.1.9 | Rate of Climb at Topsl Climb (TOC) | Rate of climb at TOC shall be greater than 300 ft/ min | A320 AC-AAMP | >300 | ft/min | SHALL | 2023-02-02 |
| 0.1.10 | Approach Speed | Approach Speed shall be 136 Kcas | | 136 | kcas | SHALL | 2023-02-02 |
| 0.1.11 | Alternate Distance | Alternate Distance shall be 200 nm | A320 AC-AAMP | 200 | nm | SHALL | 2023-02-02 |
| 0.1.12 | Holding Time | Holding time MUST be 30 min | A320 AC-AAMP | >=30 | min | MUST | 2023-02-02 |
| 0.2 | **Operations** | | | | | | |
| 0.3 | **Operations - General Requirements** | | | | | | |
| 0.4 | **Performance** | | | | | | |
| 0.5 | **Emissions** | | | | | | |
| 1 | **Physical** | | | | | | |
| 1 | **Cabin Requirements** | | | | | | |
| 1 | **Cabin - General Requirements** | | | | | | |
| 1.0.0 | Passenger transport | The cabin shall provide a comfortable area for transporting people and provide services. | | | | SHALL | 2023-02-02 |
| 1.1.1 | Max. Number of passenges (Single Class) | The number of passenges shall be 180. | A320 AC-AAMP | 180 | | SHALL | 2023-02-03 |
| 1.2.1 | Design Payload | The Design payload shall be less than 25000 kg. | | <25000 | kg | SHALL | 2023-02-02 |
| 1.2.2. | Hatrack Modularity | The cabin design shall enable a modular assembly of the hatracks | | | | SHALL | 2023-02-02 |
| 1.2.3 | Cabin altitude | Cabin altitude shall be less than 8000 ft. | CS 25 841 Pressurised cabins | 8000 | feet | SHALL | 2023-02-02 |
| 1.2.4 | numbers of flight attendants | There must be one crew member per 50 passengers each. | FAR 91.533 A320 AC-AAMP | 1 | | MUST | 2023-02-02 |
| 1.2.5 | Mass per Pax | Mass of single passenger shall be 95 kg. | nach Annahme Airbus | 95 | kg | SHALL | 2023-02-02 |
| 1.3.1 | Total Cabin Energy Demand | The total energy of the cabin and furnishings, provided by the power system, shall be less than 75000W. | | 75000 | W | SHALL | 2023-02-02 |
| 1.4 | **Cabin Space Requirements** | | | | | | |
| 1.4.1 | Passenger Comfort | The accessibility of the passenger service functions shall be less than 0.715 m. | Anthropometrische Daten | <0.715 | m | SHALL | 2023-02-14 |
| 1.4.2 | Min. Width of Aisle | The width of the aisle is minimum 380mm if the seat height is less than 640mm from floor. | CS-25-815 | >=380 | mm | MUST | 2023-02-14 |
| 1.4.3 | Min. Width of Aisle upper part | The width of the aisle is minimum 510mm if the seat height is equal to 640mm or more from floor. | CS-25-815 | >=510 | mm | MUST | 2023-02-14 |
| 1.4.4 | Max. seats abreast for single aisle | The number of seats abreast in a single aisle cabin MUST NOT exceed 3 on each side of the aisle. | CS-25-817 | <=3 | | MUST | 2023-02-14 |
| 1.4.5 | Min. quantity Lavatory | The minimum quantity of lavatories shall be 4 for 180 passengers. | Airbus Modell | 4 | | SHALL | 2023-02-14 |
| 1.4.6 | Lavatory dimensions | The lavatory shall have a length of 36inch and a width of 36inch. | Lavatory 3D Modell | 36 | inch | SHALL | 2023-02-14 |
| 1.4.7 | Galley dimensions | The after galley has a length of 0,82m and a width of 3,7m. | Galley 3D Modell | | m | SHOULD | 2023-02-14 |
| 1.4.8 | Quantity of trolleys | The galleys shall store 14 trolleys. | Airbus Modell | 14 | | SHALL | 2023-02-14 |
| 1.4.9 | Min. quantity Galley | The cabin shall have two galleys. | Airbus Modell | 2 | | SHALL | 2023-02-14 |
| 1.4.10 | Number of classes | The cabin shall have a seat layout with one class. | | 1 | | SHALL | 2023-02-14 |
| 1.4.11 | Number of aisles | The number of aisle shall be one. | | 1 | | SHALL | 2023-02-14 |
| 1.4.12 | Seat pitch Economy | The seat pitch in the economy class shall be 29-34inch. | A320 Lufthansa | 29-34 | inch | SHALL | 2023-02-14 |
| 1.4.13 | Seat pitch Business | The seat pitch in the business class shall be 36-38inch. | A320 Lufthansa | 36-38 | inch | SHALL | 2023-02-14 |
| 1.4.14 | Seat dimensions | The width of an economy seat is between 475-525mm. | A320 Recaro | >=475, <=525 | mm | MUST | 2023-02-14 |
| 1.4.15 | Passenger dimensions | The passenger is 1.90m high. | 95% Amerikaner männlich | 1,95 | m | MUST | 2023-02-14 |
| 1.4.16 | Crossection inner diameter | The inner diameter of the crossection shall be 3770mm. | A320 | 3770 | mm | SHALL | 2023-02-14 |
| 1.4.17 | Crossection outer diameter | The outer diameter of the crossection shall be 3950mm. | A320 | 3950 | mm | SHALL | 2023-02-14 |
| 2 | **Propulsion System Requirements** | | | | | | |

Abbildung A.22: Ausschnitt der Exceltabelle mit den Anforderungen.

### A.6.2 Vorstellung des Partialmodells für das Wasserstofftanksystem

In der Fallstudie I wird für die Auslegung der Variante 1b ein Wasserstofftanksystem ausgelegt. Das Partialmodell für dieses Subsystem wird vom Projektpartner Centerline Design bereitgestellt. Diese Arbeit übernimmt keine Verantwortung für die Richtigkeit oder Vollständigkeit des Modells. Der folgende Abschnitt gibt einen kurzen Einblick in den Aufbau des Modells.
Das Partialmodell des Wasserstofftanksystems besteht aus einem SysML-Modell und einem Python-Modell. Mit dem SysML-Modell wird die Tanksystemarchitektur modelliert und die Systemanforderungen überprüft. Die interne Struktur erfolgt nach dem EVA-Prinzip. Abbildung A.23 zeigt die Ordnerstruktur des Wasserstofftanksystemmodells im Cameo Systems Modeler. Zuerst werden im Input-Ordner die Anforderungen aus der Exceltabelle eingelesen und Parameter, wie z.B. das benötigte Tankvolumen, aus der XML-Datei importiert.

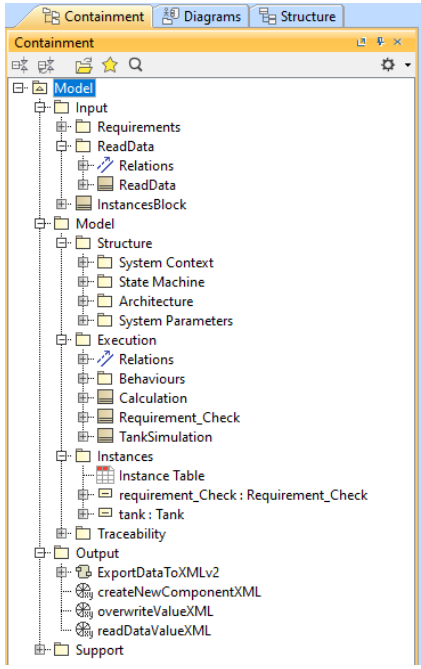



Abbildung A.23: Interne Ordnerstruktur des Wasserstofftanksystemmodells im Cameo Systems Modeler.

Im Ordner Model sind die Diagramme und Elemente für die Modellierung des Tanksystems hinterlegt. Für die Modellierung der Systemarchitektur wird unter anderem ein Blockdefinitionsdiagramm verwendet (Abbildung A.24). Das Flüssigwasserstoffspeichersystem besteht aus den Komponenten Rohren, Pumpen, Tank, Sicherheitsventilen, Entlüftungsventilen und Sensoren [FAF+23]. Mit dem Beziehungselement Kom-

position sind die einzelnen Blöcke in Beziehung gesetzt. Die Modellierung fokussiert die Auslegung des Wasserstofftanks. Weitere Systemkomponenten wie die Pumpen oder die Sensoren werden vereinfacht als Black-Box betrachtet [FAF+23].

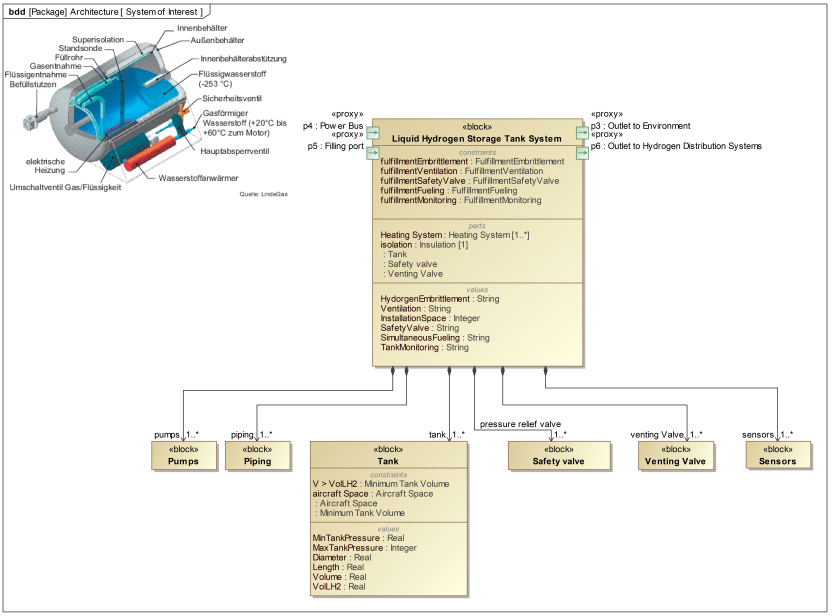



Abbildung A.24: Blockdefinitionsdiagramm für die Modellierung der Systemarchitektur des Wasserstofftanksystems im Cameo Systems Modeler, modifiziert von [FAF+23].

Die geometrische und funktionale Auslegung und Optimierung des Wasserstofftanksystems erfolgt mit einem Python-Modell. Die Ansteuerung des externen Python-Modells erfolgt mit der Opaque Aktion TankOptimizationPython in einem Aktivitätsdiagramm (Abbildung A.25). Das Optimierungsmodell ist ebenfalls nach dem EVA-Prinzip strukturiert. In diesem findet eine Dimensionierung und überschlägige Auslegung des Tanks, ausgehend vom verfügbaren Bauraum, von Materialkennwerten und von Betriebsparametern statt [FAF+23]. Das Ziel der Auslegung ist es, den Wärmeeintrag in den Flüssigwasserstoff möglichst gering zu halten [FAF+23]. Als Zielfunktion der Optimierung wird daher die Oberfläche der Tankwand minimiert, wobei die Wärmeisolation und die äußere Tankwand als konstante Größen betrachtet werden [FAF+23]. Weitere berechnete Ergebnisse umfassen das Systemgewicht, den Massenschwerpunkt und die Boil-off Rate. Während der Optimierung wartet das SysML-Modell auf die finalen Werte. Dies wird im Aktivitätsdiagramm über eine 5 s Warteschleife realisiert. Sobald die Optimierung abgeschlossen ist, werden die Ergebnisse für die Anforderungsüberprüfung in die Instanzen des Cameo Systems Modeler zurück geschrieben.

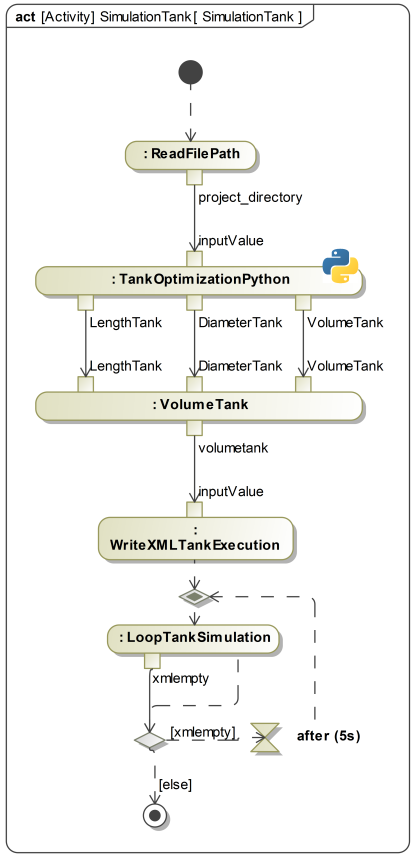



Abbildung A.25: Aktivitätsdiagramm für die Simulation des Wasserstofftanksystems und Ansteuerung des Optimierungsmodells in Python, modifiziert von [FAF+23].

Abschließend werden im Output-Ordner mit der Ausführung eines Aktivitätsdiagramm die Ergebnisse der Auslegung an die XML-Adapterdatei übergeben. Weitere Einblicke in den Aufbau der Modellierung können [FAF+23] entnommen werden.

### A.6.3 Vorstellung des Partialmodells für das Brennstoffzellensystem

In der Fallstudie I wird für die Auslegung der Variante 1b ein Brennstoffzellensystem ausgelegt. Das Partialmodell des Brennstoffzellensystems wird von Meier [Mei23] vom Projektpartner HAW Hamburg bereitgestellt. Die Verantwortung für die inhaltliche Richtigkeit und Vollständigkeit des verwendeten Modells liegt nicht bei der Verfasserin dieser Forschungsarbeit. Der folgende Abschnitt gibt einen kurzen Einblick in den Aufbau des Modells.

Das Partialmodell des Brennstoffzellensystems besteht aus einem SysML-Modell und einem Matlab/Simulink-Modell. Die interne Struktur des SysML-Modells erfolgt ebenfalls nach dem in dieser Arbeit vorgestellten EVA-Prinzip. Abbildung A.26 zeigt die Ordnerstruktur im Cameo Systems Modeler. Im Input-Ordner werden für die Auslegung des Systems die Missionsparameter aus der XML-Adapterdatei eingelesen. Dies beinhaltet unter anderem Werte für die Dauer der einzelnen Flugphasen, die maximale Flughöhe und die erforderliche Leistung jeder Phase [FAF+23]. Zudem werden die Anforderungen an das Subsystem aus der Exceltabelle eingelesen.

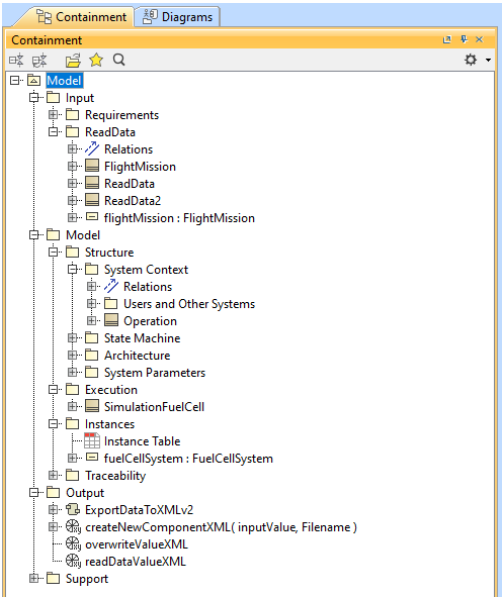



Abbildung A.26: Interne Ordnerstruktur des Brennstoffzellensystemmodells im Cameo Systems Modeler.

Im Modellordner sind die Diagramme und Elemente für die Modellierung der Brennstoffzellensystemarchitektur hinterlegt. In der Arbeit von Meier wird eine Polymer Elektrolyt Membran (PEM) Brennstoffzelle modelliert. Abbildung A.27 zeigt das Interne Blockdiagramm für die Darstellung der internen Stoff-, Informations- und Energieflüsse innerhalb des Brennstoffzellensystems. Als Verbindungspunkte zwischen den einzelnen Komponenten werden Ports verwendet (Block mit Pfeil). In diesen werden auch die Flussrichtungen gekennzeichnet. Weitere Diagramme wie Blockdefinitionsdiagramme veranschaulichen den Systemkontext, die -architektur und mögliche Zustände in den Betriebsphasen [FAF+23].

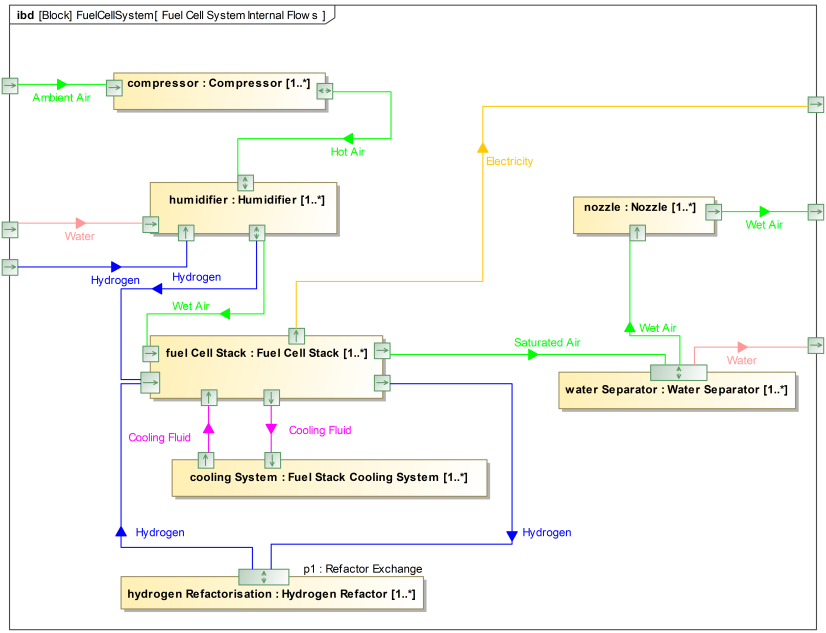



Abbildung A.27: Internes Blockdiagramm für das Brennstoffzellensystem im Cameo Systems Modeler, modifiziert von [FAF+23, Mei23].

Die Auslegung und Simulation des Brennstoffzellensystems inklusive des Verdichters und des Befeuchter-Wärmetauscher-Verbunds erfolgt mit Matlab/Simulink. Dafür wird die interne Schnittstelle des Cameo Systems Modelers für die Ansteuerung eines Matlab/Simulink-Modells genutzt. Das Aufrufen des Matlab/Simulink-Modells erfolgt über die Funktion *run startmatlabmodelcameo2.m* im Opaque-Element (Abbildung A.28). Im Matlab/Simulink-Modell wird eine Optimierung durchgeführt, die innerhalb festgelegter Grenzen die benötigten Luft- und Wasserstoffmassenströme für die geforderte Stackleistung unter Berücksichtigung der parasitären Verdichterleistung ermittelt [Mei23]. Im Anschluss an die Optimierung werden die Parameter für die verfügbare Leistung, das erforderliche Volumen des Wasserstoffs und die Gesamtmasse an das Modell im Cameo Systems Modeler zurückgegeben. Abschließend werden im Output-Ordner mit der Ausführung eines Aktivitätsdiagramm die Ergebnisse der Auslegung an die XML-Adapterdatei übergeben. Vertiefende Einblicke in den Aufbau des Modells und die Modellierung können [Mei23] entnommen werden.

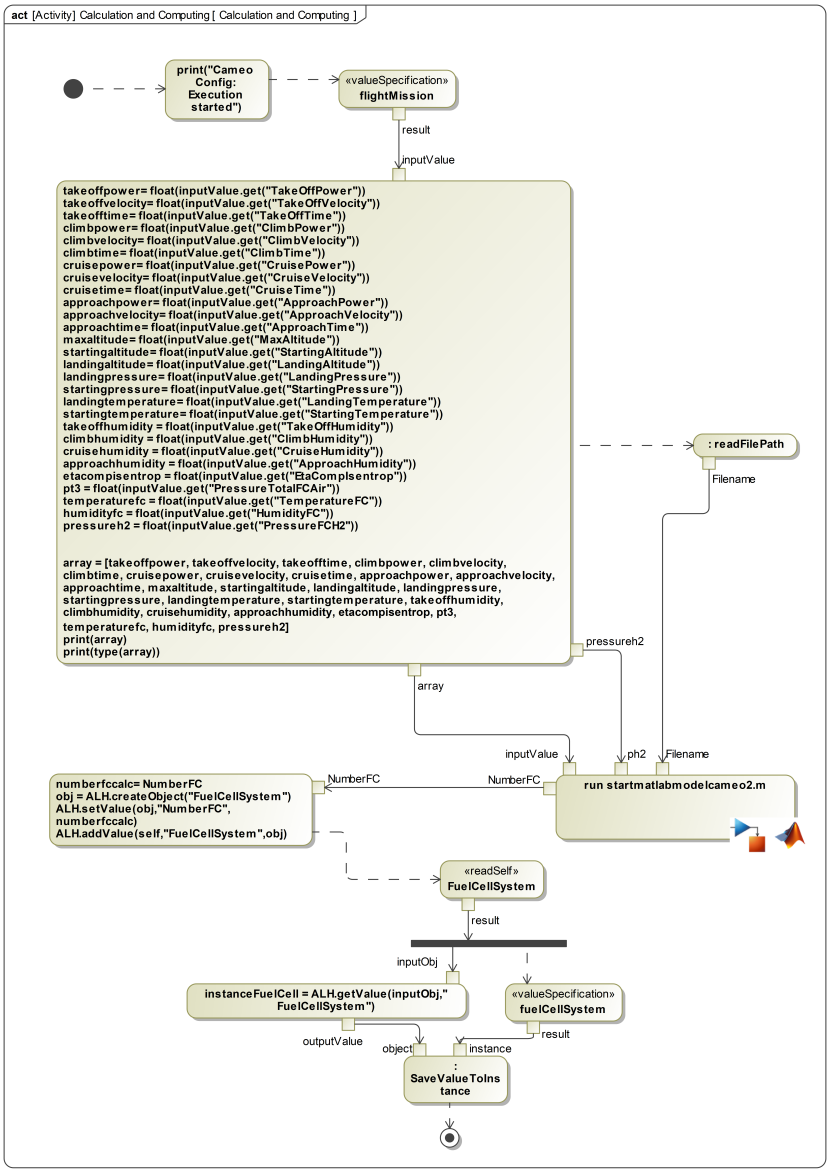



Abbildung A.28: Aktivitätsdiagramm für die Auslegung des Brennstoffzellensystems und Ansteuerung des Optimierungsmodells in Matlab/Simulink.

### A.6.4 Messungen der Prozesszeiten

Für die Bestimmung der durchschnittlichen Durchlaufzeiten der Prozesse für die Auslegung der Varianten wurden Messungen durchgeführt. Insgesamt wurde der Auslegungsprozess jeder Variante zehn Mal wiederholt durchlaufen. Im Cameo Systems Modeler wurde die Animationsgeschwindigkeit für die Simulation auf den Wert 95/100 eingestellt. Die Zeiten wurden manuell mit einer digitalen Stoppuhr gemessen. Die Ergebnisse der Messungen für die Varianten 1a und 1b sowie die Customizingsvarianten 1b-1 und 1b-2 sind in Tabelle A.1 zusammengetragen. Dabei zeigen die Messungen jedes Prozessschrittes minimale Standardabweichungen im Bereich 0,25 s bis 2,18 s auf, die auf Messungenauigkeiten beim manuellen Stoppen zurückzuführen sein könnten. Die schnellsten Prozesszeiten treten bei der Ausführung der Matlab-Modelle auf. Mit Mittelwerten im Bereich 18,07 s und 28,37 s benötigen die Modelle für ihre Berechnungen die kürzesten Zeiten. An zweiter Stelle folgen die Ausführungen des SysML Modells im Cameo Systems Modeler. Am meisten Zeit benötigt die Ausführung des Blender-Modells. Der Mittelwert für die Ausführung des Blender-Modells für Variante 1b-2 beträgt 851,16 s. Der deutliche Unterschied zu der Ausführung bei Variante 1b-1 liegt in den detaillierteren 3D-Modellen der Systemgruppe der Gepäckablage (Typ XL) und in der damit verbundenen geforderten höheren Rechenkapazität des Computers. Bei den Varianten 1a und 1b reduzieren sich die Prozesszeiten nochmal deutlich für das Blender-Modell. Der Grund hierfür ist, dass bei der 3D Visualisierung nur das Kabinen-LOPA bestehend aus den Sitzen, den Küchen, den Waschräumen und das jeweilige Stromversorgungssystem visualisiert werden. Im Gegensatz zu den Customizingvarianten werden keine weiteren Systemgruppen und deren Komponenten dargestellt.

Tabelle A.1: Messergebnisse der einzelnen Kabinenprozessschritte von zehn Durchläufen für Variante 1a und 1b sowie die Customizingvarianten 1b-1 und 1b-2. (CSM: Cameo Systems Modeler)

| | *Variante 1a* | | | *Variante 1b* | | | *Variante 1b-1* | | *Variante 1b-2* | |
|---|---|---|---|---|---|---|---|---|---|---|
| | *Modell CSM* | *Modell Matlab* | *Modell Blender* | *Modell CSM* | *Modell Matlab* | *Modell Blender* | *Modell Matlab* | *Modell Blender* | *Modell Matlab* | *Modell Blender* |
| **Durchlauf** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** | **Zeit [s]** |
| 1 | 84,33 | 28,67 | 106,36 | 71,55 | 27,12 | 95,27 | 18,01 | 446,44 | 19,51 | 851,74 |
| 2 | 83,92 | 29,24 | 106,47 | 70,80 | 27,36 | 97,09 | 18,09 | 448,14 | 18,78 | 854,51 |
| 3 | 85,14 | 27,96 | 107,10 | 70,71 | 26,90 | 94,84 | 18,04 | 451,60 | 19,44 | 852,38 |
| 4 | 83,77 | 29,48 | 109,10 | 71,49 | 27,16 | 96,10 | 17,60 | 451,81 | 19,64 | 848,91 |
| 5 | 84,10 | 27,32 | 104,98 | 70,60 | 26,88 | 95,75 | 18,11 | 446,92 | 19,10 | 851,71 |
| 6 | 84,05 | 28,20 | 110,01 | 72,34 | 28,11 | 96,42 | 17,79 | 446,88 | 19,41 | 849,92 |
| 7 | 84,76 | 28,44 | 103,53 | 70,95 | 27,24 | 96,17 | 18,35 | 450,36 | 19,51 | 849,57 |
| 8 | 83,98 | 27,68 | 105,61 | 72,10 | 27,60 | 95,70 | 18,36 | 448,55 | 18,96 | 854,29 |
| 9 | 84,91 | 28,62 | 103,50 | 71,38 | 26,98 | 96,40 | 17,97 | 452,15 | 19,49 | 848,18 |
| 10 | 83,20 | 28,08 | 104,50 | 70,67 | 28,34 | 95,35 | 18,35 | 449,06 | 19,34 | 850,21 |
| Mittelwert | 84,22 | 28,37 | 106,11 | 71,26 | 27,27 | 95,91 | 18,07 | 449,19 | 19,32 | 851,16 |
| Standard-abweichung | 0,58 | 0,61 | 2,18 | 0,62 | 0,37 | 0,66 | 0,25 | 2,17 | 0,28 | 2,17 |