Studienarbeit

Konfigurationsmanagement von Systemen mit MBSE

für die Abschlussprüfung zum Certified Systems Engineers (GfSE)® Ebene A,

Oktober 2023

Hugo, Ormo

Adams-Lehman-Straße 51, 80797 München

München, 15.10.2023

Inhalt

1 Einleitung

Moderne Systeme werden aufgrund der steigenden Erwartungen an Komplexität und Interoperabilität sowie des wirtschaftlichen Drucks oft im Rahmen von Produktlinien[1] entstehen. Sie werden in verschiedenen Systemvarianten bzw. Ausprägungen vorgesehen, die durch unterschiedliche Kombinationen und Gestaltungen ihrer Bestandteile spezifische Lösungen für eine Vielfalt an Problemen oder Chancen bieten. Ohne den Ansatz des Systems Engineering und insbesondere des modellbasierten Systems Engineering (MBSE[2]) ist ihre Entstehung kaum mehr zu bewältigen[3] [4]. Dies entspricht in erster Linie dem Ansatz eines an die Schwierigkeit der Herausforderung angepassten Konfigurationsmanagements[5] [6]. Während des Lebenszyklus eines Systems erfolgt eine iterative Weiterentwicklung seiner Bestandteile, wodurch sich Produktänderungen[7] ergeben, die zu neuen Versionen einiger Entwicklungsartefakte führen. Diese Versionen werden durch Konfigurationseinheiten[8] verwaltet. Die Referenzkonfigurationen[9]

---

[1] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten 2017, Seite 262.
[2] Vgl. Lünnemann, Pascal, „https://www.ipk.fraunhofer.de/de/kompetenzen-und-loesungen/digital-engineering/modellbasiertes-systems-engineering/was-ist-model-based-systems-engineering.html“, 26.08.2023 (Stand des letzten Zugriffs)
[3] Vgl. Gausenmeier, Jörg, Roman Dumitrescu, Daniel Steffen, Anja Czaja, Olga Wiederkehr, Christian Tschirner, https://www.hni.uni-paderborn.de/fileadmin/Fachgruppen/Seniorprofessur_Gausemeier/systemsengineerings/Studie_Systems-Engineering.pdf“, 26. 08 2023 (Stand des letzten Zugriffs), Seite 10.
[4] Vgl. Lünnemann, Pascal, „https://www.ipk.fraunhofer.de/de/kompetenzen-und-loesungen/digital-engineering/modellbasiertes-systems-engineering/was-ist-model-based-systems-engineering.html“
[5] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 188.
[6] Vgl. Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Falls Church, VA, 1992, Seite 9.
[7] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 189.
[8] Vgl. ISO (the International Organisation for Standardization), ISO10007:2017, ISO, Genf 2017, Seite 1.
[9] Vgl. ISO (the International Organisation for Standardization), ISO10007:2017, ISO, Seite 1.

beschreiben Kombinationen von Konfigurationseinheiten, die zu einer bestimmten Zeit die gewünschten Lösungen darstellen. Die Referenzkonfigurationen müssen sorgfältig verwaltet werden, denn es muss zu jeder Zeit ersichtlich sein, welche Änderungen implementiert sind und welche noch nicht, welche alternative Lösungen noch gültig und welche schon obsolet sind, welche Verifikations- und Validierungsmaßnahmen welche Anforderungen verifizieren oder validieren und welche Systemelemente welche Anforderungen erfüllen. Daraufhin muss auch ersichtlich sein, mithilfe welcher unterstützenden Systeme, wie Prozesse, Methoden und Tools, das System entworfen, hergestellt, betrieben, gewartet und zuletzt stillgelegt wird. Diese Vielfalt an Artefakten sind figurativ miteinander durch einen Draht von gegenseitigen Abhängigkeiten verbunden: Eine Veränderung in einem der Artefakte wird eine Kaskade von Veränderungen im Verbund auslösen.

"The Digital Thread purely wants to connect all relevant artifacts that are part of the development of a system, trace between them, ensure their configuration and help organizations to identify what is affected in case of a change."[10] Im Kontext komplexer Systeme “There is no Digital Thread without MBSE”[11]: Das Ziel des MBSE ist „die formalisierte Anwendung von Modellierung [,] um Systemanforderungs-, -entwurfs-, -analyse-, -verifikations- und -validierungsaktivitäten in den frühen Konzeptphasen, der Entwicklung und späteren Lebenszyklusphasen zu unterstützen.“[12] [13] Beim MBSE geht es nicht nur darum, ein Systemmodell herzustellen, sondern dieses Systemmodell als Wegweiser der Daten und ihrer Semantik in einer datenzentrierten Unternehmensarchitektur festzulegen[14].

---

[10] Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, 29.07.2023 (Stand des letzten Zugriffs), Seite 33.
[11] Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 31.
[12] Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 291.
[13] International Council on Systems Engineering (INCOSE) in: SYSTEMS ENGINEERING VISION 2020, Ausgabe: 09/2007, Seite 15.
[14] Vgl. Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 31.

In dieser Studienarbeit werden in Kapitel 2 die aktuelle Problematik und die entsprechenden Bestrebungen vorgestellt. In Kapitel 3 wird die Forschungsfrage gestellt. In Kapitel 4 wird das Ergebnis der Studie vorgestellt. Dafür wird eine Ontologie[15] des Konfigurationsmanagements vorgeschlagen, entlang dessen die unterschiedlichen ontologischen Elemente definiert und kommentiert werden. Das Ergebnis wird in Kapitel 5 mit einem Beispiel ergänzt. Im Beispiel wird eine Vorgehensweise für ein nach der Object Oriented Systems Engineering Method (OOSEM)[16] mit der System Modeling Language (SysML)[17] in der Version 1.6 modelliertes Fahrzeugsystem dargestellt. Ein Ausblick zur Version 2.0 der SysML und die Anwendung von künstlich intelligenten Agenten (KI-Agenten) im Konfigurationsmanagement werden in Kapitel 6 diskutiert. Anschließend wird in Kapitel 7 zusammengefasst, wie das Konfigurationsmanagement, der Digital Thread und der Digital Twin zusammenhängen und warum sie in der Verwaltung komplexer Systeme relevant sind.

[15] Vgl. Balandi, Oguzahn, „https://kobra.uni-kassel.de/handle/123456789/13754“, 26.08.2023 (Stand des letzten Zugriffs), Seite 8-9.
[16] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Waltham, Ma USA, 2011, Seite 431-519.
[17] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/index.htm”, 26.08.2023 (Stand des letzten Zugriffs), Seite 7.

2 Stand der Technik

Die Verwaltung von Konfigurationseinheiten und Referenzkonfigurationen stellt für Unternehmen eine besondere Herausforderung dar. Denn Konfigurationseinheiten versionieren Artefakte, die verteilt über mehrere Datenbanken des Unternehmens aufbewahrt werden[18].

Diese Datenbanken werden von monolithischen Product Lifecycle Management (PLM) Lösungen „PLM monoliths“[19] gesteuert, die ihre Daten bedingt effizient abarbeiten, und kaum Interoperabilität mit anderen Datenbanken vorsehen[20].

Die PLM-Lösungen bieten Konfigurationsmanagementfunktionalitäten für ihre Daten und basieren deshalb auf einer Single-Source-of-Truth. Die Daten können ohne die Logik der PLM-Lösung, die den Herausgebern gehört, unbrauchbar werden und zu einem Vendor Lock-in führen[21].

Die PLM-Industrie richtet sich aktuell nach den Microservices[22] und den „Linked Data for Engineering“[23] neu aus. Dies soll durch die Umsetzung von „Data-Centric Architectures”[24] erfolgen, die den Bedarf nach „shared and authoritatively managed sets of data”[25] decken sollen.

---

[18] Vgl. Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 3-4.
[19] Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 3.
[20] Vgl. Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 5-6.
[21] Vgl. Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 7-11.
[22] Vgl. Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 10-11.
[23] Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 18-19.
[24]Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 28-30.
[25] McDermott, Tom; Kelly Alexander, Richard Wallace, "The Supra-Systemmodell." In: INCOSE INSIGHT Ausgabe: Volume 26/Issue 2, 06/2023, Seite 17-18.

Das Supra-Systemmodell[26] stellt ein allumfassendes Lebenszyklusmodell für die drei Lebenszyklusmodelle, die zum Ergebnis des Systems beitragen, dar:

1. Das Lebenszyklusmodell des Zielsystems selbst und die Systeme in seinem Kontext,
2. das Lebenszyklusmodell des Projektes,
3. das Lebenszyklusmodell der Entwicklung.

Entsprechend dieser Sichtweise erlebt das System als Teil des Supra-Systems einen iterativen Lebenszyklus mit drei Phasen: Lernen, Umsetzen und Überwachen, wie in Abbildung 1 dargestellt[27].

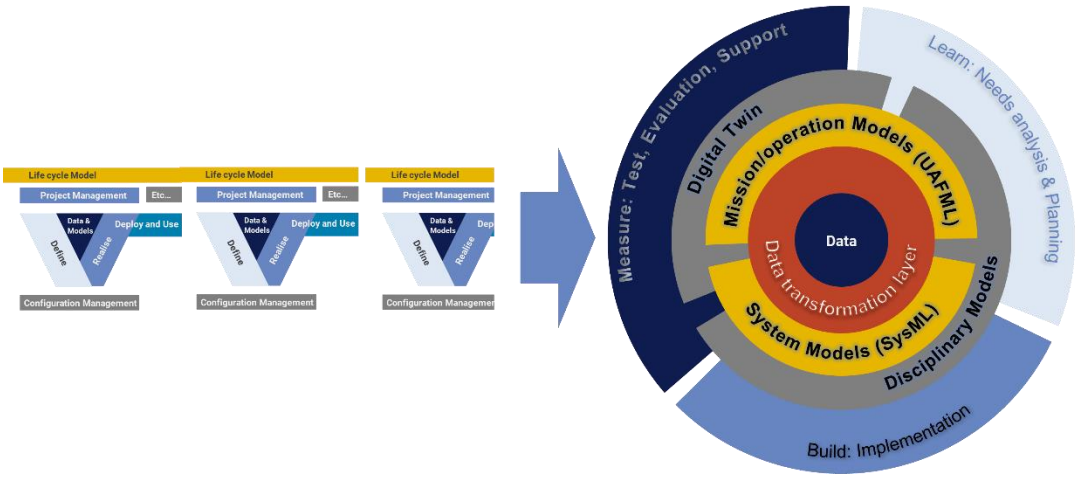

Abbildung 1: Im Gegensatz zu einer Reihenfolge von V-Modellen, stellt das zirkulare Modell die Daten, Transformationen und Modelle ersichtlicher im Zentrum des Kreises dar[28].

Rund um die Daten und die Transformationsebene befinden sich das Systemmodell und die strategischen und operativen Modelle. Die Daten sollen nun für den gesamten Lebenszyklus maßgeblich verwaltet werden und durch Transformationen interpretiert werden. „Digital artifacts may still be documents or presentable

[26] Vgl. McDermott, Tom; Kelly Alexander, Richard Wallace, "The Supra-Systemmodell." In: INCOSE INSIGHT, Seite 15.
[27] Vgl. McDermott, Tom; Kelly Alexander, Richard Wallace, "The Supra-Systemmodell." In: INCOSE INSIGHT, Seite 18-19.
[28] Vgl. McDermott, Tom; Kelly Alexander, Richard Wallace, "The Supra-Systemmodell." In: INCOSE INSIGHT, Seite 17.

views but should remain digitally connected to the underlying data and models from which they draw context and explainability.“[29]

Die aktuelle Arbeit bei The Object Management Group (OMG) zur Weiterentwicklung der SysML für die Modellierung der Systemarchitektur und der Unified Architecture Framework Modeling Language *(*UAFML v1.2)[30] für die Modellierung der strategischen und operativen Modelle soll den Aufbau des Kerns des Supra-Systemmodells ermöglichen. Die Meta Object Facility (MOF™) Versioning and Development Lifecycle Specification[31] spezifiziert die nötige Semantik, die die übergreifende Verwaltung von Versionen und das Konfigurationsmanagement ermöglicht. Die Daten und Semantik werden in der eXtensible Markup Language (XML)[32] erfasst und die Transformationen als XML Metadata Interchange (XMI®)[33] ermöglicht.

[29] McDermott, Tom; Kelly Alexander, Richard Wallace, "The Supra-Systemmodell." In: INCOSE INSIGHT, Seite 17.
[30] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/cgi-bin/doc?formal/22-07-10.pdf“, 26.08.2023 (Stand des letzten Zugriffs), Seite 3.
[31] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/mof/”, 26.08.2023 (Stand des letzten Zugriffs), Seite 1.
[32] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/spec/XMI/2.5.1/About-XMI/“, 26.08.2023 (Stand des letzten Zugriffs), Seite 5.
[33] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/spec/XMI/ISO/19509/PDF“, 26.08.2023 (Stand des letzten Zugriffs), Seite 1-2.

3 Forschungsfrage

Die Zahl der Konfigurationseinheiten und Referenzkonfigurationen wächst entlang der Lebenszyklusphasen eines Systems und ihre Verwaltung wird immer aufwändiger. Darüber hinaus wird es nicht ausreichen, wenn nur das zu entwickelnde System unter Konfigurationsmanagement gehalten wird; die interoperierenden Systeme und die Unterstützungssysteme müssen auch unter Konfigurationskontrolle gehalten werden. Durch die Wiederverwendung von Architekturen, Lösungen und Komponenten werden Konfigurationseinheiten über den Lebenszyklus des Zielsystems hinaus gelten.

Auf die Forschung im Supra-Systemmodell aufbauend stellt sich die Frage: Wie können die Konfigurationseinheiten und die Referenzkonfigurationen im Systemmodell und in den operativen und strategischen Modellen definiert und modelliert werden, so dass zu jedem Zeitpunkt der Digital Thread[34] ermöglicht wird?

[34] Vgl. Bleisinger, Oliver, et al, „https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 33.

4 Ergebnis

In dieser Studienarbeit wird der Ansatz des Konfigurationsmanagements mit MBSE anhand eines Beispiels in der SysML[35] vorgestellt. Das Systemmodell wird mit weiteren ontologischen Elementen ergänzt, die die Verwaltung der Konfigurationen im Systemmodell ermöglichen. Diese Vorgehensweise greift auf vorgelagerte Arbeiten zur Verwaltung von Modellen mit Modellen zurück.[36] [37]
Die Studienarbeit beschränkt sich auf den Umfang des Systemmodells. Die strategischen und operativen Modelle werden deshalb im Beispiel aufgrund des Umfangs der Studie nicht betrachtet.

4.1 Die ontologischen Elemente des Konfigurationsmanagements

Die ontologischen Elemente[38] [39], welche in dieser Studienarbeit für das Konfigurationsmanagement eine Rolle spielen, sind in Abbildung 2 dargestellt. Diese Ontologie bedient sich der Dienste der Meta Object Facility (MOF™) Versioning and Development Lifecycle Specification, um eine vollumfängliche Konfigurationsmanagementvorgehensweise nach den Richtlinien[40] des International Council on Systems Engineering (INCOSE) zu gewährleisten. Die wesentlichen Elemente der Ontologie werden nachstehend beschrieben.

---

[35] Vgl. The Object Management Group®(OMG®), "https://www.omg.org/index.htm"
[36] Vgl. Schulte, Tim, Marc Schneider, Udo Judaschke, Daniel Batz, Systemmodelle verwalten mit ConfigML – Motive, Grundlagen und erste Konzepte einer Sprache für das modellbasierte Konfigurationsmanagement, Tag des Systems Engineering 2016, München 2016, Seite 327.
[37] Vgl. Schulte, Tim, Samson Groß, Sebastian Langer, Lucas Kirsch, ConfigML – Erste prototypische Realisierung einer Verwaltung von Modellen mit Modellen im PLM, Tag des Systems Engineering 2017, München 2017, Seite 183.
[38] Nach Auskünften im Rahmen des Workshop for comprehensive & model-based knowledge management via ontologies, Prostep IVIP 2022, PROSTEP AG; Hofmann, Tamara.
[39] UWA System Health Lab, „https://ontology-explained.com/2020/obda1/", 02.09.2023 (Stand des letzten Zugriffs)
[40] Vgl. Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 190-193.

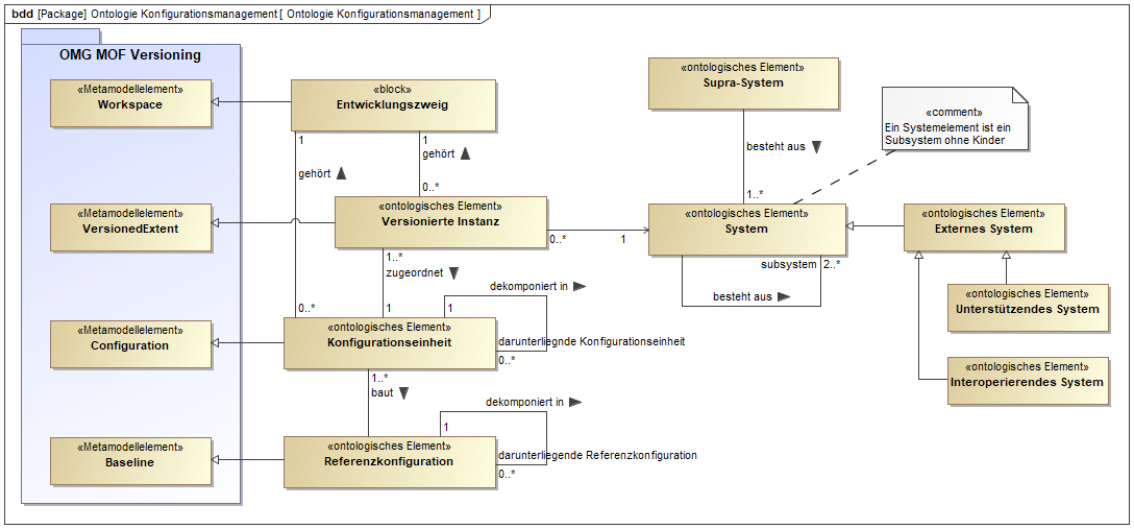

Abbildung 2: Die ontologischen Elemente des Konfigurationsmanagements in dieser Studienarbeit.

4.2 Die versionierten Instanzen

Die kleinsten Entwicklungsartefakte, die versioniert[41] werden, sind die versionierten Instanzen[42]. Neben den Systemen oder Systemteilen samt ihren Schnittstellen können auch Dokumente, Analysen, Simulationen oder Modelle versionierte Instanzen sein. Sie entsprechen einem feststehenden Zustand des Entwicklungsartefaktes zu einem bestimmten Zeitpunkt. Versionierte Instanzen werden Konfigurationseinheiten zugeordnet und bilden die Historie der Konfigurationseinheit (Developmental configuration)[43].

4.3 Die Konfigurationseinheiten und die Referenzkonfigurationen

Eine Konfigurationseinheit[44] ist eine „entity within a configuration that satisfies an end use function“[45]; Konfigurationseinheiten „sind die wesentlichen Einheiten, die

[41] Eine Version ist ein zeitlicher Zustand der Beschreibung eines Elements.
[42] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/mof/”, Kapitel 5.5 VersionedExtent, Seite 6.
[43] Vgl. Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Seite 11.
[44] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/mof/”, Kapitel 5.6.3 Configuration, Seite 10.
[45] ISO (the International Organisation for Standardization), ISO10007:2017, ISO, Seite 1.

einer streng formalen Kontrolle unterzogen werden."[46] Welche Konfigurationseinheiten es gibt und was sie beinhalten, wird am Anfang eines Projekts in der Konfigurationsmanagementplan[47] nach den Kriterien der Standards[48], der Erfahrung des Unternehmens, der Referenzarchitektur[49] und den besonderen Bedürfnissen des Projekts ausgerichtet. Konfigurationseinheiten dürfen aus anderen darunterliegenden Konfigurationseinheiten bestehen. Die Entwicklungsprozesse und -methoden, Betriebs-, Wartungs- und Stilllegungsverfahren sind keine Konfigurationseinheiten des Systemmodells, sondern des operativen Modells.

Eine Referenzkonfiguration[50] stellt eine „approved configuration information that establishes the characteristics of a product or service at a point in time that serves as reference for activities throughout the life cycle of the product or service"[51] indem sie eine spezifische Version einer Konfigurationseinheit festlegt. Die Referenzkonfigurationen erfassen zu jeder Zeit eine valide[52] Zusammenfassung von darunterliegenden Referenzkonfigurationen und versionierten Instanzen. Eine Referenzkonfiguration „bildet die Grundlage für die nächste Änderung."[53] Die Referenzkonfigurationen werden durch einen für diesen Zweck gebildeten Gremien, der Konfigurationsausschuss (Configuration Control Board), genehmigt.[54]

---

[46] Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 190.
[47] Vgl. Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Seite 18-19.
[48] Vgl. ISO (the International Organisation for Standardization), ISO10007:2017, ISO, Seite 3.
[49] Siehe Kapitel 5.3.
[50] Vgl. The Object Management Group®(OMG®), "https://www.omg.org/mof/", Kapitel 5.6.4 Baseline, Seite 10.
[51] ISO (the International Organisation for Standardization), ISO10007:2017, ISO, Seite 1.
[52] Im Sinne des Lösens des Problems oder der Chance der Unternehmung.
[53] Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 188.
[54] Vgl. Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Seite 8.

4.4 Die Entwicklungszweige

Entwicklungszweige[55] beinhalten versionierte Instanzen und Konfigurationseinheiten; allerdings nur eine einzige Version derselben versionierten Instanz oder Konfigurationseinheit. In einem Entwicklungszweig dürfen Versionen eingepflegt[56] werden. Solche Versionen können einzelne versionierte Instanzen sowie ganze Referenzkonfigurationen sein. Ein Entwicklungszweig kann auf der Grundlage einer Konfigurationseinheit erzeugt werden. Von einem Entwicklungszweig kann eine Kopie gemacht werden.

4.5 Die Variationen, Varianten und ihr Bezug zu den Referenzkonfigurationen

Bei der Gestaltung eines Systems werden aus vielerlei Gründen[57] Variationen eintreten. Eine Variation repräsentiert ein abstraktes System, Subsystem oder Systemelement, das in unterschiedlichen Ausprägungen, nämlich Varianten, gestaltet und umgesetzt wird[58]. Die Varianten müssen konsistent[59] mit der Variation, von der sie stammen, sein[60]. Eine Variation kann durch eine Referenzkonfiguration verwaltet werden, die ihre eigenen Varianten umfasst.
Varianten und Alternativen sind in dieser Studienarbeit nicht identisch: Alternativen stellen exklusive Lösungsmöglichkeiten dar, wobei letztendlich nur eine dieser Alternativen umgesetzt wird; sei es eine alternative Variante oder eine allumfassende alternative Gestaltung des Systems.

---

[55] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/mof/”, Kapitel 5.6.2 Workspace, Seite 8-9.
[56] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/mof/”, Kapitel 5.6.2.2 merge, Seite 9.
[57] Varianten entstehen grundsätzlich aus technischen - ‚Markt- und Kostengründen.
[58] Vgl. Bahns, Tammo, Sylvya Melzer, Ralf God, Dieter Krause, Ein modellbasiertes Vorgehen zur variantengerechten Entwicklung modularer Produktfamilien, Tag des Systems Engineering 2015, München 2015, Seite 149.
[59] Siehe Kapitel 5.9
[60] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/index.htm”, Seite 45-46.

5 Veranschaulichung des Ergebnisses anhand eines Beispiels

Zusammen mit der Entwicklung des Systemmodells wird nachstehend anhand eines Beispiels eines Fahrzeugsystems[61] ein Baum von Referenzkonfigurationen[62] aufgebaut, der die Struktur des Systems nachbildet und ihre Verwaltung ermöglicht. Die genannten Elemente des Beispiels erhalten eine Kennzeichnung in Form dieser Notation <…>, wenn sie im Text auftreten.

5.1 Die Entstehung der Variationen

Die Variationen haben ihren Ursprung in den Bedarfen der operativen Ebene (z.B. Marketing, Vertrieb, Wartung). Die Bedarfe wurden auf der Grundlage von den Anwendungsfällen und Szenarien aus den operativen Lebenszykluskonzepten, die den Lösungsraum abstecken, erhoben[63] [64]. Dieses Set von genannten Stakeholder-Bedarfen wird in die Anforderungen der operativen Ebene überführt[65]: Sie bilden die Stakeholder-Anforderungen[66]. Die Stakeholder-Anforderungen werden in einer Spezifikation erfasst: der Stakeholder-Anforderungsspezifikation – Stakeholder Requirements Specification (StRS). Die Lebenszykluskonzepte, die Stakeholder-Bedarfe und die StRS sollen in Referenzkonfigurationen im operativen Modell oder in einem ergänzenden Modell verwaltet werden.

5.2 Die Variationen in der Systemanforderungsspezifikation

Eingeschränkt von den operativen Lebenszykluskonzepten, sowie den Stakeholder-Bedarfen und -Anforderungen werden die nötigen System-Lebenszykluskon-

---

[61] Nach Auskünften der Firma AUDI AG; Herr Alexander Dück (I/EZ-X).
[62] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Seite 173-176 und 492-496.
[63] Vgl. Wheatcraft, Lou, Tami Katz, Michael Ryan, Raymond B. Wolfgang, INCOSE Needs and Requirements Manual, San Diego, California USA, 2022, Seite 84-156.
[64] Vgl. Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 70.
[65] Vgl. Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 77.
[66] Vgl. Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 74.

zepte festgelegt und auf deren Grundlage die Systembedarfe zuletzt erhoben[67]. Unter anderen Aspekten führen die Bedarfe auch die Variationen in die Lösung ein. Aus den Systembedarfen werden die Systemanforderungen überführt und in der Systemanforderungsspezifikation – System Requirements Specification (SyRS) erfasst[68]. Die SyRS spezifiziert die Varianten des Systems, wie in Abbildung 3 dargestellt.

Die System-Lebenszykluskonzepte, die System-Bedarfe und die SyRS sollen in Referenzkonfigurationen im Systemmodell oder in einem ergänzenden Modell verwaltet werden.

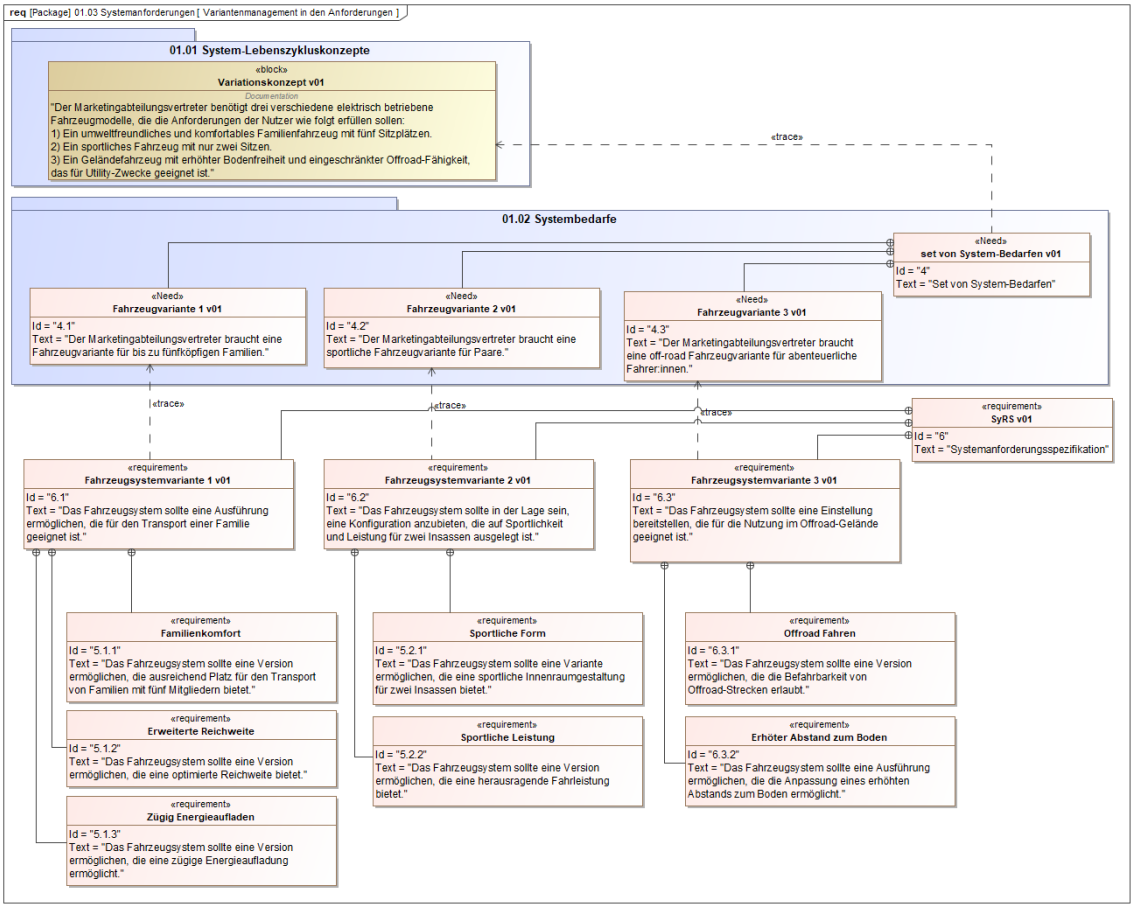

Abbildung 3: Variationskonzept, Systembedarfe und Systemanforderungen für drei Fahrzugvarianten. Typ, Status, Priorität und Verifikationsmethode der Systemanforderungen sind nicht dargestellt.

[67] Vgl. Wheatcraft, Lou, Tami Katz, Michael Ryan, Raymond B. Wolfgang, INCOSE Needs and Requirements Manual, Seite 33.
[68] Vgl. Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 85.

5.3 Die Variationen und Varianten des Systems in der Referenzarchitektur

Die Referenzarchitektur beschreibt eine abstrakte[69] Architektur eines Referenzsystems, die den Wissensbestand des Unternehmens erfasst, worauf neue Entwicklungen basieren[70]. Die Variationen und Varianten können, insbesondere bei der Entwicklung von Produktlinien, in der Referenzarchitektur bereits vorhanden sein.

In dem hier auf Basis eines Fahrzeugsystems dargestellten Beispiel handelt es sich um eine sogenannte Brownfield-Entwicklung. Im Gegensatz zu einer Greenfield-Entwicklung[71] werden in einer Brownfield-Entwicklung überwiegend bekannte Lösungen angewendet. In diesem Fall stellt oft die Referenzarchitektur bereits die logischen Knoten der logischen Architektur[72] dar. Das Diagramm in Abbildung 4 zeigt exemplarisch diese abstrakten[73] logischen Knoten der Variation <Antriebssystem> und seine Varianten <Antriebsystem Schnellladung> und <Antriebsystem Hochleistung>.

[69] Vgl. Delligatti, Lenny, SysML Distilled, A Brief Guide to Systems Modeling Language, Crawfordsville, Indiana, 2013, Seite 51.
[70] Vgl. Walden, Roedler, et al. GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 246.
[71] Vgl. Walden, David D., et al, INCOSE Systems Engineering Handbook. A guide for Systems Life Cycle Processes and Activities, Sab Diego, California USA, 2023, Seite 229-230.
[72] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SySML: The Systems Modeling Language, Seite 475-481.
[73] Vgl. Delligatti, Lenny, SysML Distilled, A Brief Guide to Systems Modeling Language, Seite 51.

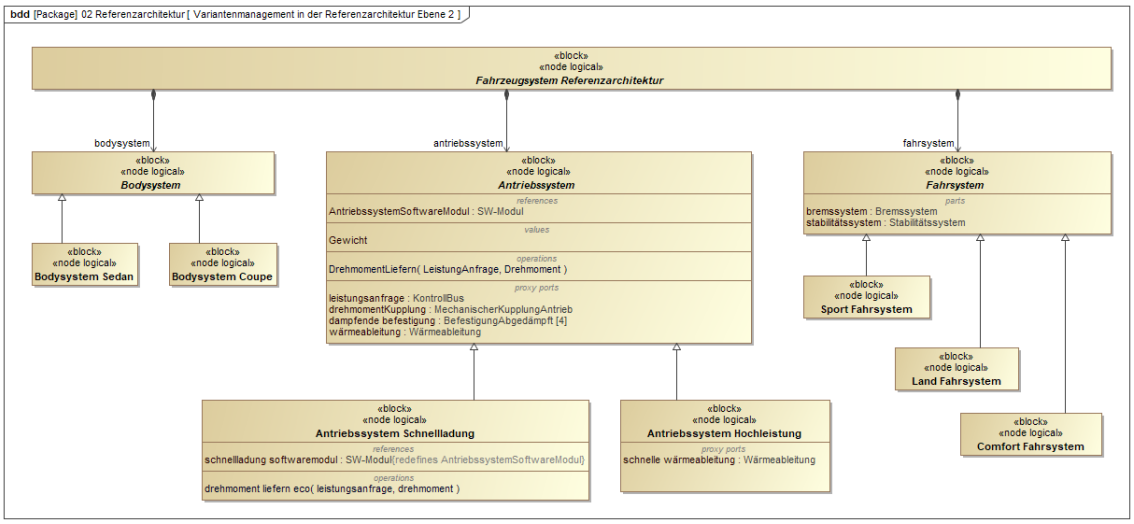

Abbildung 4: Abstrakte (siehe kursive Namen in den Blocks) Referenzarchitektur auf Systemebene 2 mit herausgehobenem Detail der Variationen (Variationen sind immer Abstrakt) und Varianten.

Die Struktur des Systems in der Referenzarchitektur befindet sich rekursiv auf jeder Systemebene. Siehe das Detail auf Systemebene 3 der Variationen <Bremssystem> und <Stabilitätssystem> in Abbildung 5.

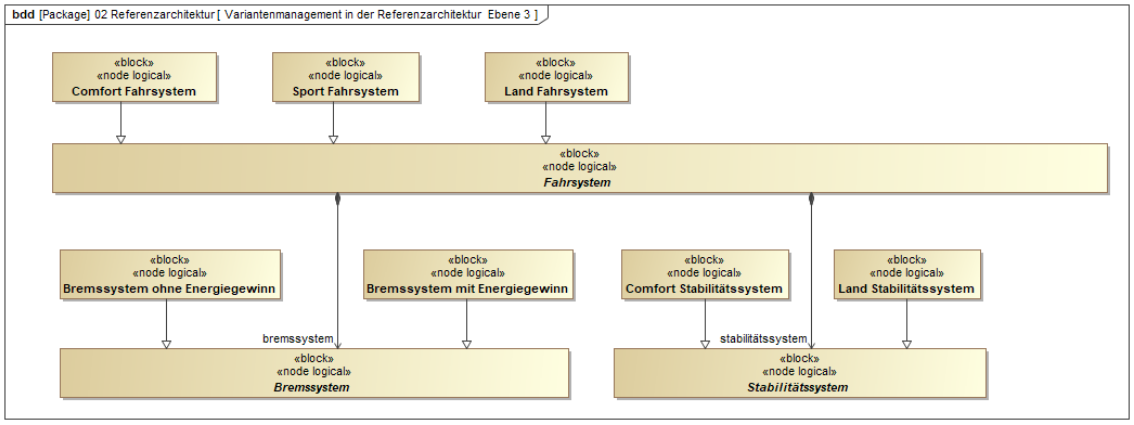

Abbildung 5: Abstrakte Referenzarchitektur des <Fahrsystem> auf Systemebene 3 mit Variationen (Abstrakt) und spezialisierten Varianten.

5.4 Die Varianten in der logischen Architektur

Die Referenzarchitektur gilt als Vorlage für die logische Architektur. Ein Entwicklungszweig der Referenzarchitektur wird für das aktuelle Entwicklungsprojekt vorgelegt. Die erforderlichen logischen Varianten des Fahrzeugsystems sind in Abbildung 6 durch eine Spezialisierung[74] aufgebaut.

[74] Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Seite 169-170.

Durch die Spezialisierung erben[75] die logischen Knoten die nötigen Eigenschaften[76] der Referenzelemente. Manche Eigenschaften werden in der aktuellen Entwicklung unverändert beibehalten und manche besonderen Eigenschaften werden für dieses bestimmte Projekt angepasst, entfernt oder ergänzt. Die SysML bietet diese Möglichkeit durch das SysML-redefine[77].

Falls nicht in der Referenzarchitektur vorhanden, werden nötige Systemelemente neu entworfen und einem logischen Knoten zugeordnet. Sie werden im <Fahrzeugsystem> modelliert, und damit sowohl für alle Varianten des Fahrzeugsystems verfügbar gemacht als auch für einen möglichen Rückfluss in der Referenzarchitektur, wenn der Konfigurationsausschuss (Configuration Control Board, CCB) es für nötig hält.

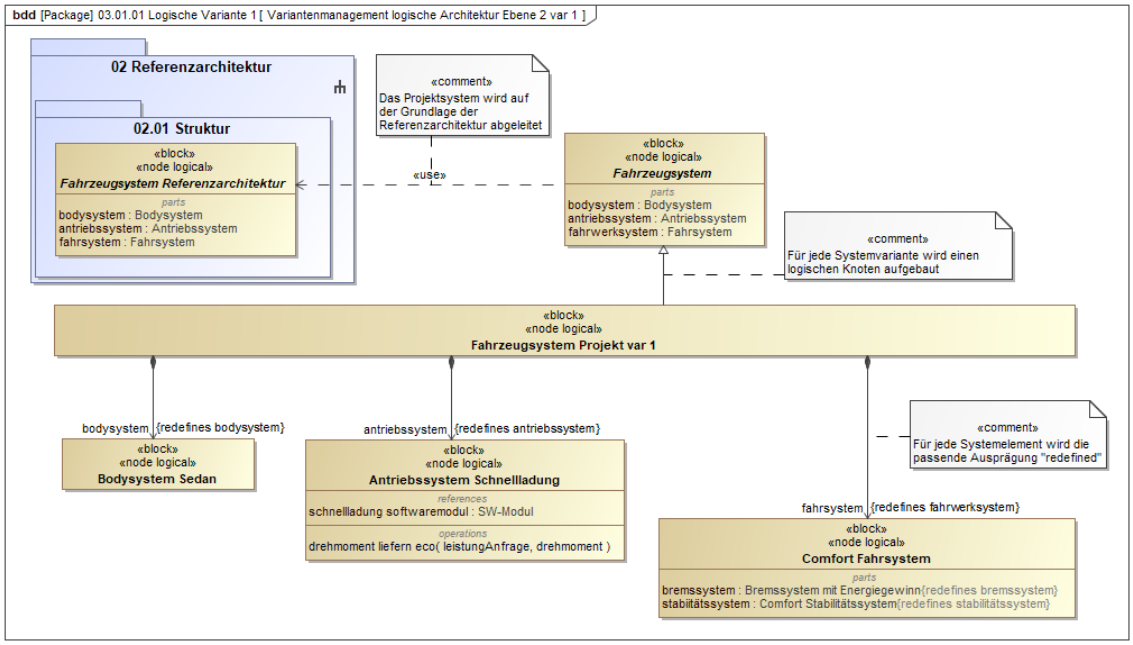

Abbildung 6: Systemarchitektur des <Fahrzeugsystem> in der Variante "Var 1" auf Ebene 2 mit den entsprechenden Varianten jedes darunterliegenden logischen Knotens.

Diese Vorgehensweise findet rekursiv auf jeder Systemebene statt, wie in Abbildung 7 dargestellt.

[75] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Seite 168-169.
[76] In der SysML lassen sich die Teileigenschaften, die Werteigenschaften und die Verhaltenseigenschaften modellieren.
[77] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Seite 169-170.

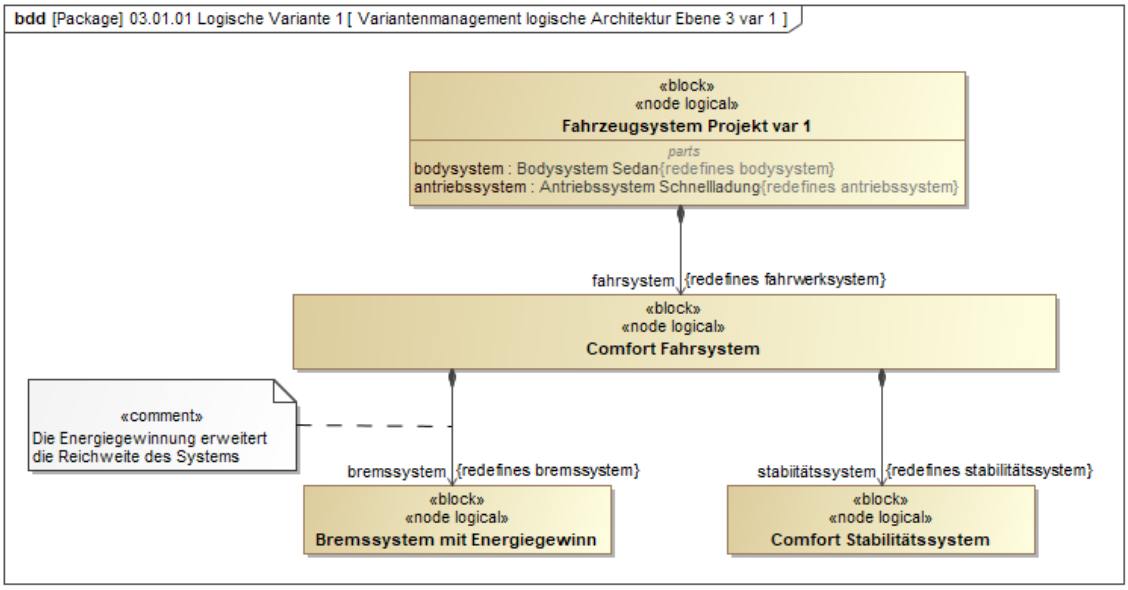

Abbildung 7: Systemarchitektur der Comfort-Variante des <Fahrsystem> auf Ebene 3 mit den Ausprägungen jedes darunterliegenden logischen Knotens.

Eine zweite Fahrzeugsystemvariante wird für eine bessere Nachvollziehbarkeit des Beispiels analog zur Variante 1 in Abbildung 8 dargestellt.

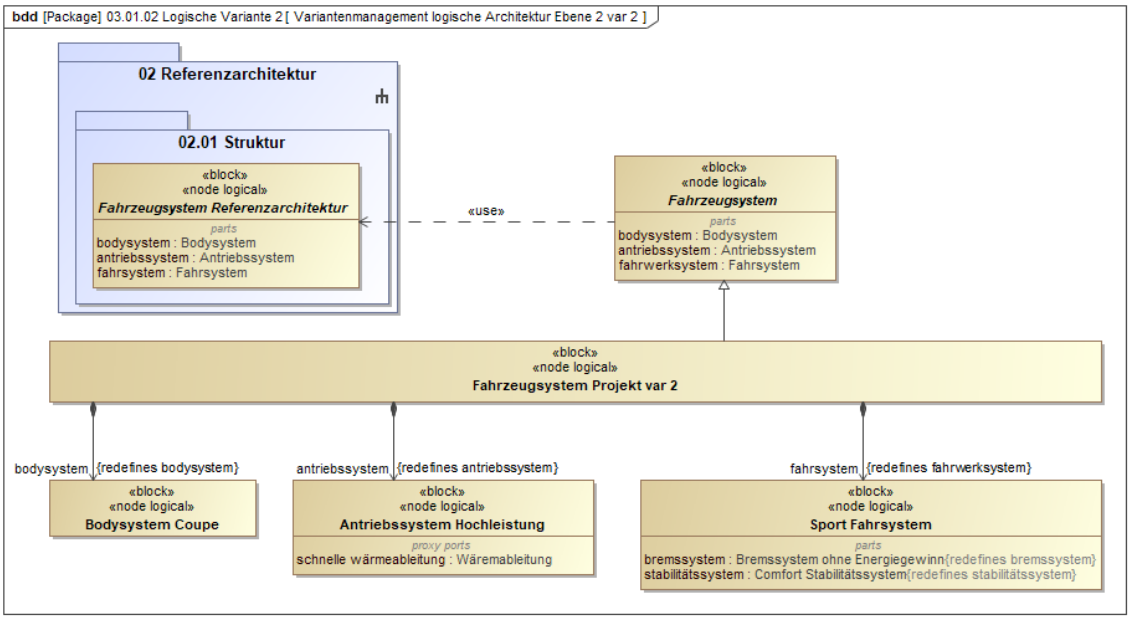

Abbildung 8: Systemarchitektur des <Fahrzeugsystem> in der Variante "Var 2" auf Ebene 2 mit den entsprechenden Varianten jedes darunterliegenden logischen Knotens.

Begründungen[78] werden für alle Anpassungen an die Referenzarchitektur dokumentiert und auf die entsprechenden logischen Knoten, Eigenschaften oder Assoziationen[79] bezogen, wie in Abbildung 9 dargestellt.

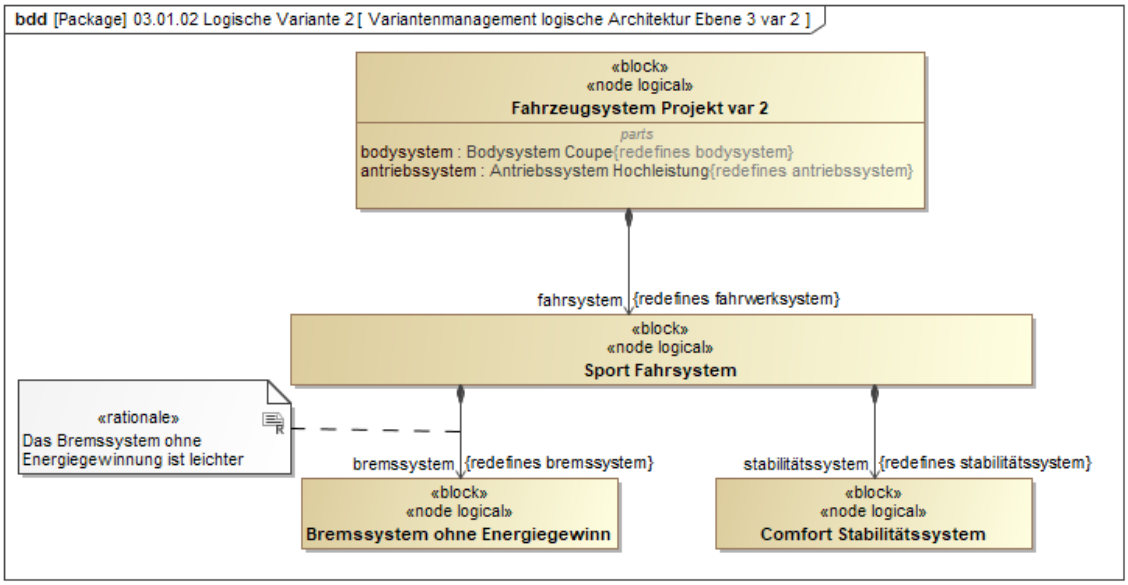

Abbildung 9: Systemarchitektur der Variante <Sport Fahrsystem> auf Ebene 3 mit den entsprechenden Varianten jedes darunterliegenden logischen Knotens.

Die in diesem Beispiel in den Fokus gestellten variantenbezogenen Systemanforderungen werden von den logischen Knoten jeder Variante des Fahrzeugsystems auf jeder Systemebene erfüllt. Dies wird mithilfe einer Matrize wie in Abbildung 10 visuell dargestellt.

[78] Vgl. Delligatti, Lenny, SysML Distilled, A Brief Guide to Systems Modeling Language, Kapitel 11.7, Seite 213.
[79] Vgl. Delligatti, Lenny, SysML Distilled, A Brief Guide to Systems Modeling Language, Kapitel 3.5, Seite 44-46.

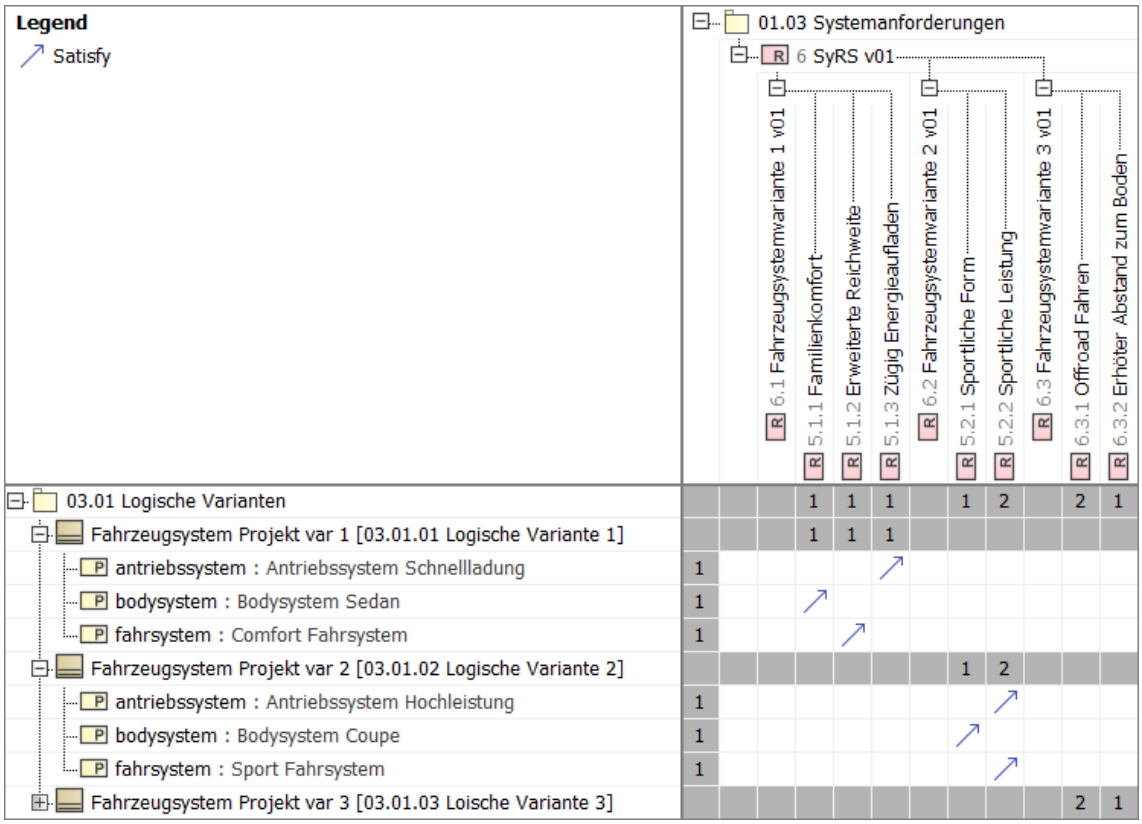

Abbildung 10: Erfüllungsmatrize zwischen den logischen Knoten und den variantenbezogenen Systemanforderungen. Die Variante 3 wird in dieser Studienarbeit nicht dargestellt, sondern aus Vollständigkeitsgründen in der Erfüllungsmatrize zusammengefaltet.

5.5 Die Varianten werden detailliert

Mithilfe der unterschiedlichen Systemanalyse und -simulationen (nicht im Beispiel durchgeführt) werden immer weitere Eigenschaften der logischen Knoten auf jeder Systemebene weiter verfeinert. Die Verfeinerungen werden prinzipiell für alle Systemvarianten in dem <Fahrzeugsystem> geltend modelliert und anschließend durch das SysML-redefine selektiv für jede entsprechende Variante des Fahrzeugsystems ausgewählt. Zuletzt sind die Varianten des Fahrzeugsystems ausreichend detailliert, wie exemplarisch in Abbildung 11 dargestellt.

Die Detaillierung der Systeme, Subsysteme und Systemelemente wird oft im Rahmen der Gestaltung der Architektur und des Entwurfs in bestimmten Architektursichten durchgeführt. Diese Architektursichten fokussieren sich auf bestimmte Anliegen und werden durch einen Entwicklungszweig je Architektursicht verwaltet. Solche Entwicklungszweige werden abschließend in einen einzelnen Entwicklungszweig eingepflegt, der das Systemmodell vervollständigt.

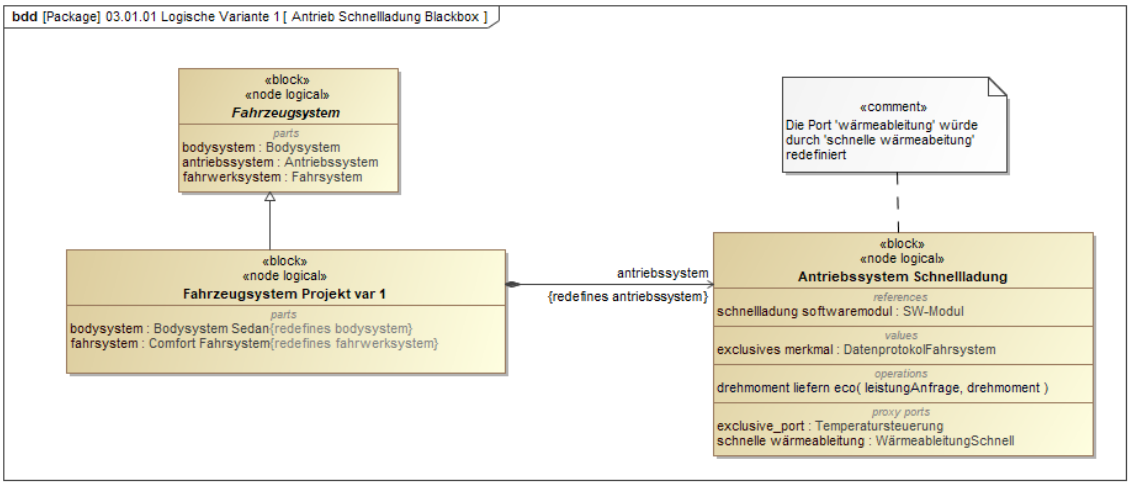

Abbildung 11: Die logischen Knoten aller Varianten werden durch die unterschiedlichen Architektursichten je nach Anliegen nach dem Wissen der Disziplinfachrichtungen abgearbeitet. Alle exemplarischen Bestimmungen in der <WärmeAbleitungSchnell> betreffen daher nur das <Antriebssystem Schnellladung>.

5.6 Die alternativen logischen Architekturen

Alternative logische Systemarchitekturen[80], die alternativen Lösungen des Problems oder der Chance des Systems entsprechen, werden untersucht und bewertet. Eine alternative logische Systemarchitektur ist nur gültig, wenn alle Systemanforderungen erfüllt sind. Die Begründungen in Bezug auf die Gewichtung und Metrik jedes Bewertungskriteriums[81] sowie die Auswahl der endgültigen Alternative werden dokumentiert (nicht im Beispiel durchgeführt). Für jede alternative logische Systemarchitektur wird ein Entwicklungszweig erstellt. Die nicht ausgewählten Entwicklungszweige werden abgebrochen und nicht weiterentwickelt.

Die logischen Knoten aus der ausgewählten Architektur werden jetzt entworfen, beschafft oder wiederverwendet.

[80] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 94.

[81] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 98.

5.7 Die physischen Knoten

Die logischen Systeme und Systemelemente sind ein Hilfsmittel, um die Funktionen des Systems abzuarbeiten, zu partitionieren und damit die Knoten der physischen Architektur bestimmen zu können. Die Konfigurationseinheiten und Referenzkonfigurationen werden erst in der physischen Architektur relevant. Dies schließt eine Verwaltung der logischen Systeme, Systemelemente und logischen Knoten durch Konfigurationseinheiten nicht aus, wird aber in dieser Studienarbeit nicht durchgeführt.

Die physischen Knoten geben die logischen Knoten im Verhältnis (1:1) wieder und werden für alle Varianten bestimmt. Die physische Architektur ist damit für jede Produktvariante, wie in Abbildung 12 und Abbildung 13 dargestellt, aufgebaut.

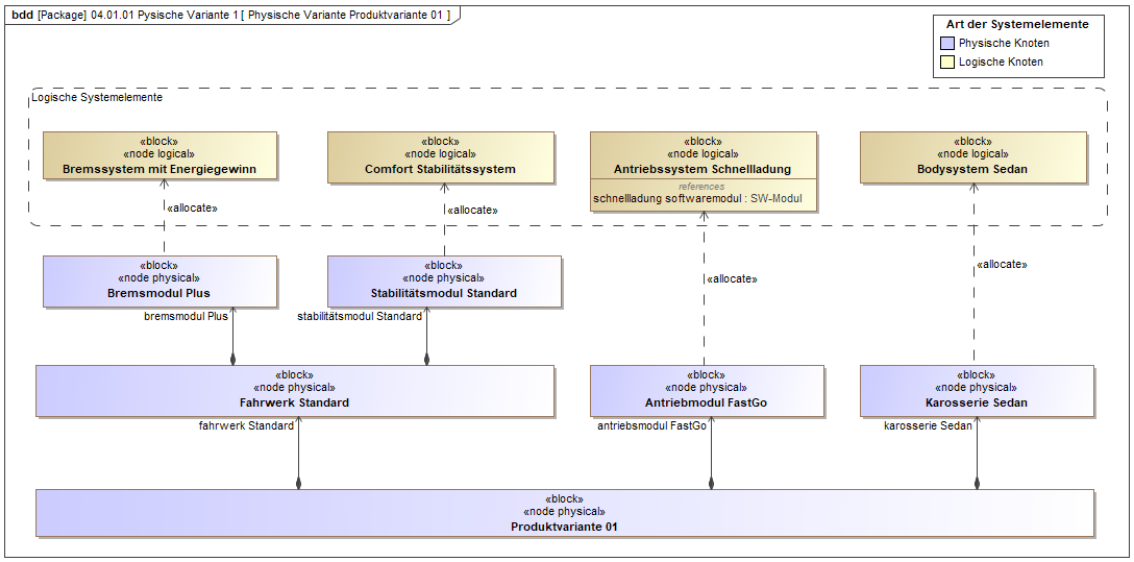

Abbildung 12: Die logischen Knoten des <Fahrzeugsystem Projekt var 1> werden in den physischen Knoten der <Produktvariante 01> wiedergegeben mittels Abhängigkeiten des Typs <allocate>.

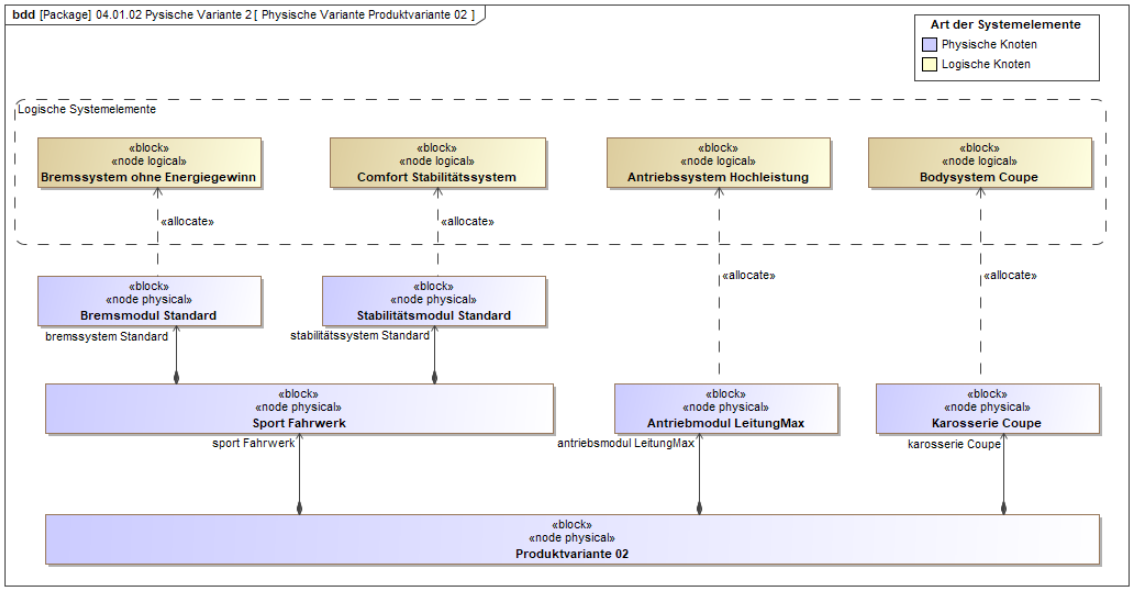

Abbildung 13: Die logischen Knoten der <Fahrzeugsystem Projekt var 2> werden in den physischen Knoten der <Produktvariante 02> wiedergegeben mittels Abhängigkeiten des Typs <allocate>.

Ein physischer Knoten gibt jeden nicht weiter dekomponierten logischen Knoten der logischen Architektur wieder. Eine Matrize in Abbildung 14 visualisiert diese Abhängigkeiten des Typs <allocate>

Es wird in diesem Beispiel nicht dargestellt, wäre aber möglich, bestimmten logischen Elementen eines logischen Knotens die physischen Elemente eines physischen Knotens zuzuordnen, zum Beispiel bei Sensoren oder Softwaremodulen.

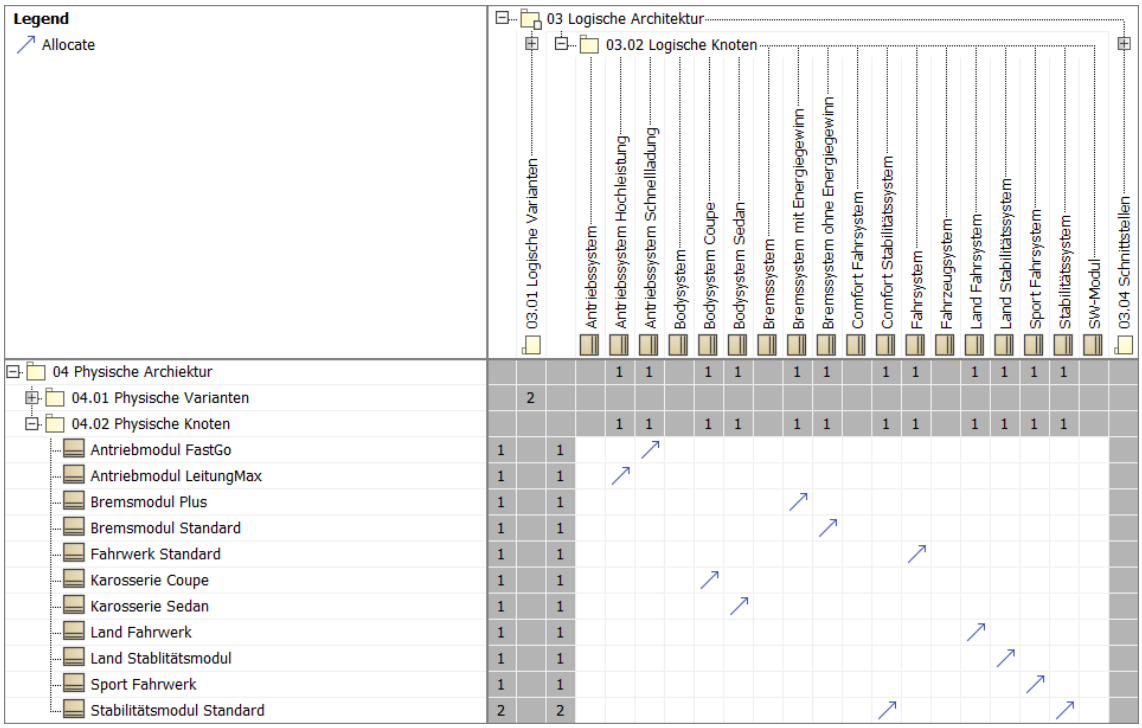

Abbildung 14: Eine Matrize macht die Wiedergabe der logischen Knoten in physischen Knoten ersichtlich.

5.8 Die physischen Elemente

Technische Lösungsalternativen für jeden physischen Knoten werden jetzt untersucht. Alternative Konfigurationen von physischen Elementen, welche einen physischen Knoten aufbauen, werden in sämtlichen trade-off studies[82] analysiert mit dem Ziel, eine abgewogene Konfiguration zu definieren, deren technische Leistungskennzahlen den Leistungskennzahlen der logischen Knoten gerecht werden. Demzufolge wird für jeden resultierenden physischen Knoten eine Referenzkonfiguration festgelegt.

Zu diesem Zeitpunkt werden die Handelsprodukte, nämlich die commercial-of-the-shelf (COTS)[83], betrachtet mitsamt ihrer dazugehörigen Dokumentation. Dafür ist für jeden COTS eine Konfigurationseinheit erforderlich[84]. Die COTS

[82] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 106.
[83] Vgl. Walden, David D., et al, INCOSE Systems Engineering Handbook. A guide for Systems Life Cycle Processes and Activities, Seite 106 und 231-232.
[84] Das, was wiederverwendet werden soll, ist nicht ein System oder Systemelement, sondern dessen Konfigurationseinheit. Es sollen neben dem System auch dessen Anforderungen, Prüfverfahren, Begründungen, Systemanalyse, geometrisches Modell usw. wiederverwendet werden.

werden, wenn nötig, im Systemmodell nachmodelliert, und zwar so detailliert wie für die Systemanalyse erforderlich wird.
Die physischen Elemente befinden sich aggregiert in ihren entsprechenden physischen Knoten[85]. Ihre spezifizierten und zu erwarteten Emergenzen sind dort ebenso verortet.

5.9 Das Prinzip Design to Abstraction[86] in der Gestaltung von Variationen und ihren Varianten

Bei der Gestaltung der Variationen und Varianten müssen die Varianten konsistent mit der Variation, von der sie stammen, sein[87]. SysML bietet mithilfe von Spezialisierungen (SysML-generalisation[88]) die Möglichkeit, die Gestaltung und Aufrechterhaltung konsistenter Varianten zu erleichtern.
Auf diese Weise lassen sich Gestaltungen erst nach bestimmten Normen und Regularien oder Standards allgemeingültig abstrakt gestalten und je nach Produktlinie und spezifischem Produkt ergänzend anpassen. Siehe gelber Gestaltungsbereich in Abbildung 15. Solche abstrakten Gestaltungen eignen sich als Konfigurationseinheiten.

[85] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Seite 482-491.
[86] Vgl. Delligatti, Lenny, SysML Distilled, A Brief Guide to Systems Modeling Language, Seite 50-51.
[87] Siehe Kapitel 4.5
[88] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Seite 167-168.

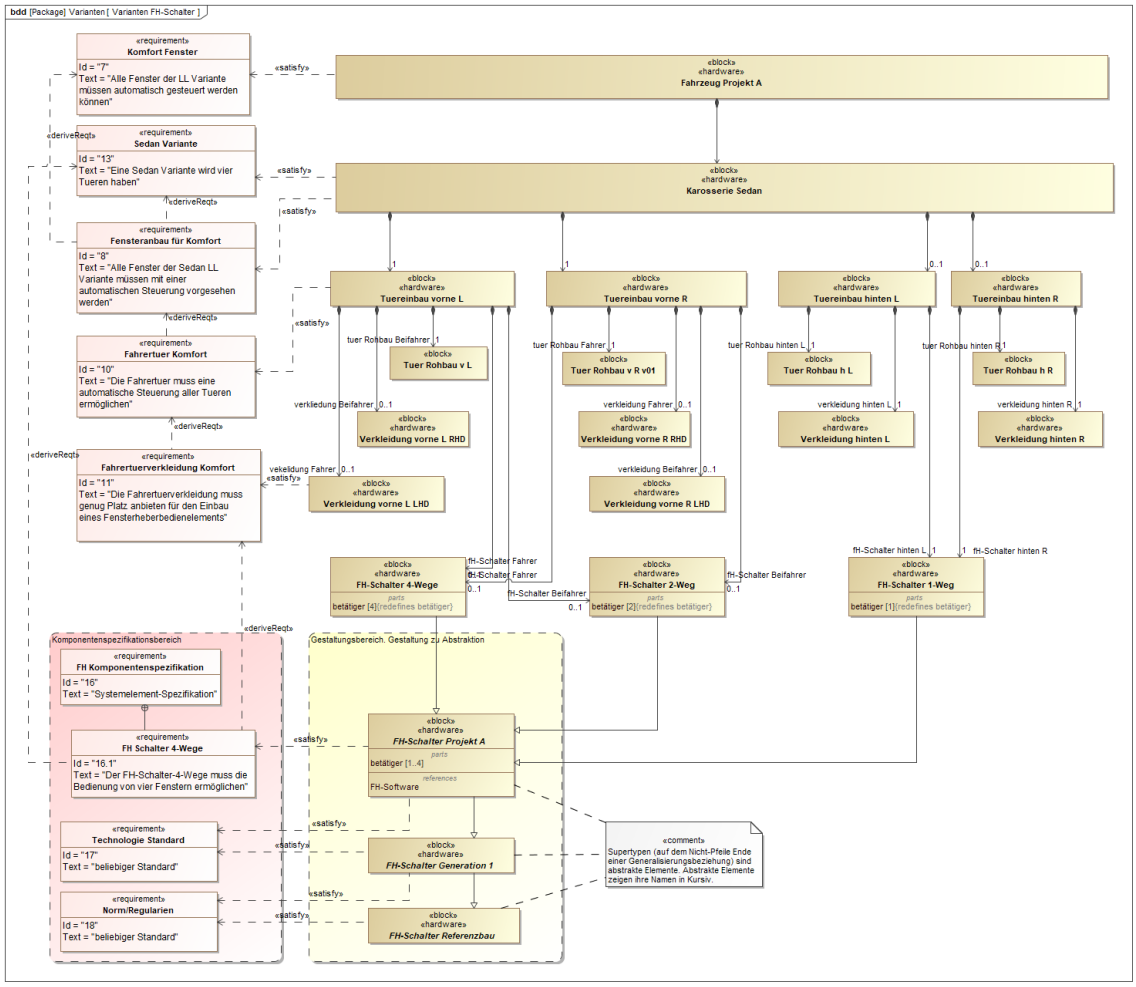

Abbildung 15: Beispiel von Design-to-Abstraction in der Gestaltung einer <Karosserie Sedan> mit verschiedenen Varianten von Fensterheber-Bedienelementen.

5.10 Von den physischen Knoten in die Stückliste

Die physischen Elemente sind nun vollständig detailliert. Jedes physische Element wird eine eindeutige (1:1) Beziehung zu einem Bestandteil der Stückliste haben. Spätere Änderungen in einem Bestandteil der Stückliste müssen erst in der physischen Architektur verifiziert und validiert werden, um mögliche Zusammenhänge mit anderen Teilen des betroffenen physischen Knotens sowie geänderte Interaktionen zwischen Knoten sicherzustellen.

Die Bestandteile der Stückliste liegen jenseits des Umfangs des Systemmodells und deshalb auch des Umfangs dieser Studienarbeit, werden aber mithilfe von Konfigurationseinheiten ebenso verwaltet. Diese Konfigurationseinheiten müssen mit den Konfigurationseinheiten der physischen Architektur rückverfolgbar sein. Dafür muss eine semantische Verbindung zwischen jeder Konfigurationseinheit der Stückliste und der Konfigurationseinheit des Systemmodells

hergestellt werden[89]. Diese Verbindungen werden den Digital Thread zwischen den Modellen ermöglichen[90] [91].

5.11 Die Referenzkonfigurationen und die versionierten Geschäftsobjekte im Systemmodell

Parallel zur Gestaltung der physischen Knoten aller Varianten werden die entsprechenden Referenzkonfigurationen definiert und aufgebaut. Für jeden physischen Knoten[92] wird eine Referenzkonfiguration erzeugt. Weitere darunterliegende Referenzkonfigurationen können unterhalb der physischen Knoten dekomponiert aufgebaut werden, wenn sie für die Verwaltung der versionierten Bestandteile des Knotens erforderlich oder vorteilhaft sind[93]. Sie werden einen Baum aufbauen, wie in Abbildung 16 exemplarisch dargestellt. Die Elemente dieses Baums werden zueinander mit einer SysML-reference association[94] nach einer (n:n) Regel einbezogen. Die Referenzkonfigurationen und die versionierten Bestandteile können, wie in diesem exemplarischen Beispiel, durch Blocks (SysML-block[95]) dargestellt werden. Die Blocks beinhalten die Versionsnummer in ihren Namen. Sie können effizienter als Baseline-Datenobjekt und VersionedExtent-Datenobjekt verwaltet werden, wenn eine mit der Meta Object Facility

---

[89] Vgl. Bleisinger, Oliver, et al, „https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, Seite 34.
[90] Vgl. McDermott, Tom, Kelly Alexander, Richard Wallace, "The Supra-Systemmodell." In: INCOSE INSIGHT, Seite 17-18.
[91] Vgl. Walden, David D., et al, INCOSE Systems Engineering Handbook. A guide for Systems Life Cycle Processes and Activities, Seite 90.
[92] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 187.
[93] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SySML: The Systems Modeling Language, Seite 532.
[94] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SySML: The Systems Modeling Language, Seite 131.
[95] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SySML: The Systems Modeling Language, Seite 119-120.

(MOF) Versioning and Development[96] übereinstimmende Entwicklungsumgebung verfügbar ist[97].

Die Anwendung der Methode VAriant Modeling with SysML (VAMOS)[98] ermöglicht die Nutzung von SysML-constraints, die die Validierung der Kombinationen von Referenzkonfigurationen unterstützen. Diese Möglichkeit wurde in diesem Beispiel nicht angewendet.

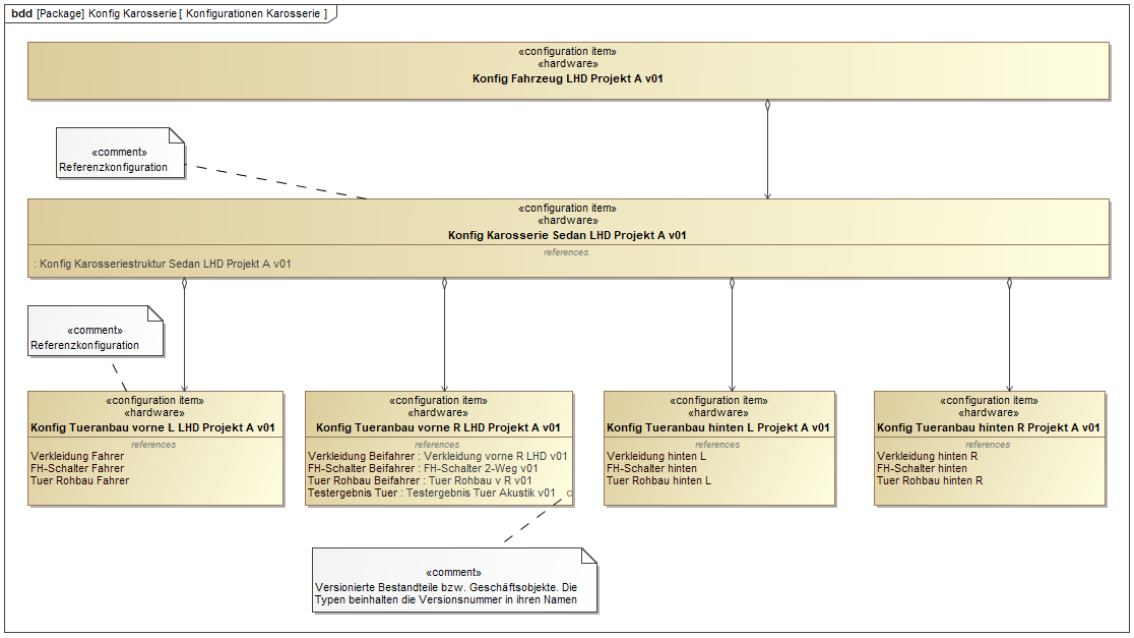

Abbildung 16: Versionierte Instanzen der Türanbauten und die jeweilige Referenzkonfigurationen durch SysML-Blocks (block) mit dem Stereotyp <<configuration item>>[99] dargestellt.

Änderungen treten aufgrund neuer Systembedarfe, Technologien, der Reduzierung der Entwicklungs-, System- und Wartungskosten oder der Erhöhung der Zuverlässigkeit des Systems[100] auf. Änderung werden durch Änderungsanträge (ECP)[101] gesteuert und durch neue Referenzkonfigurationen unter

[96] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/mof/”, Seite 1.
[97] Apache Software Foundation, „https://subversion.apache.org/“, 06.09.2023 (Stand des letzten Zugriffs)
[98] Vgl. Wilkiens, Tim, SYSMOD - the Systems Modeling Toolbox, 2020, Seite 228-229.
[99] Vgl. Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SysML: The Systems Modeling Language, Seite 174-175, 492.
[100] Vgl. Walden, Roedler, et al, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, Seite 188-189.
[101] Vgl. Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Seite 11.

Konfigurationskontrolle[102] verwaltet, wie in Abbildung 17 dargestellt. Eine neue Referenzkonfiguration wird aktualisierte versionierte Geschäftsobjekte aufweisen, wie eine aktualisierte Gestaltung des Systems, aktualisierte Anforderungen, eine aktualisierte Geometrie, aktualisierte Verifikationsvorgehen oder aktualisierte Dokumente aller Art.

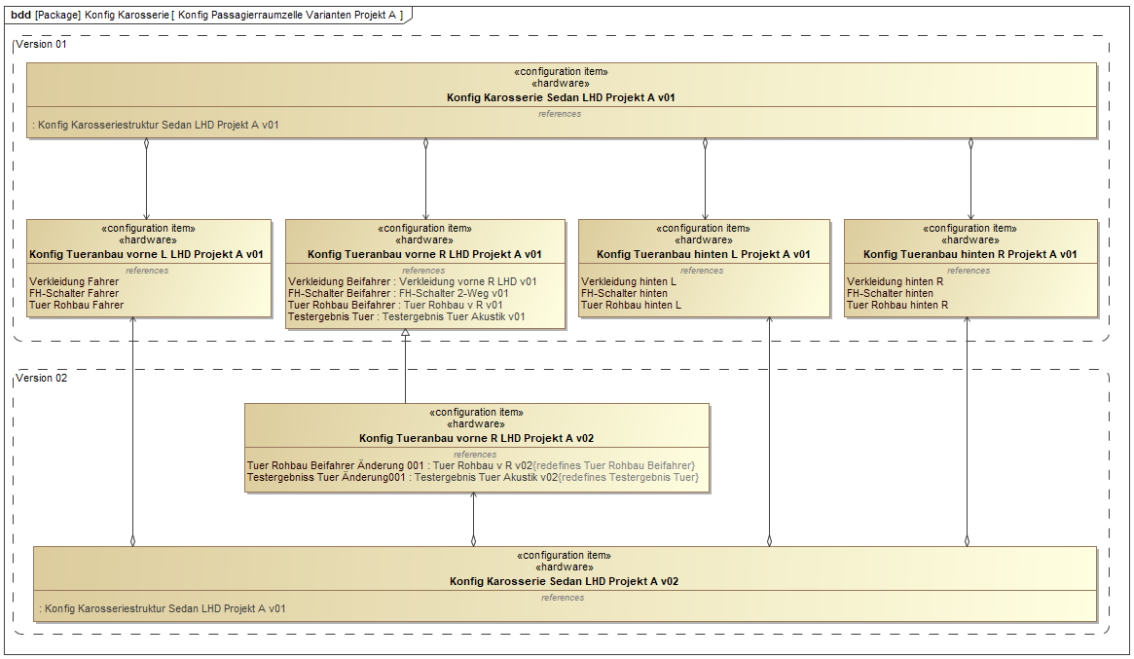

Abbildung 17: Die Referenzkonfiguration der <Karosserie> wird wegen einer Änderung in der <Tür Rohbau Beifahrer> eine neue Referenzkonfiguration (baseline) aufweisen.

In einer Referenzkonfiguration werden die entsprechenden versionierten Geschäftsobjekte zusammengefasst: System oder Systemelement, Anforderungen, Prüfverfahren, Prüfberichte, Begründungen, Systemanalyse, geometrische Modelle usw.[103] Diese Geschäftsobjekte sind im Systemmodell durch einen stellvertretenden Block modelliert, der Verweise auf die Quelldokumente beinhaltet, wie in Abbildung 18 dargestellt.

[102] Vgl. Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Seite 33.
[103] Vgl. ISO (the International Organisation for Standardization), ISO10007:2017, ISO, Seite 3.

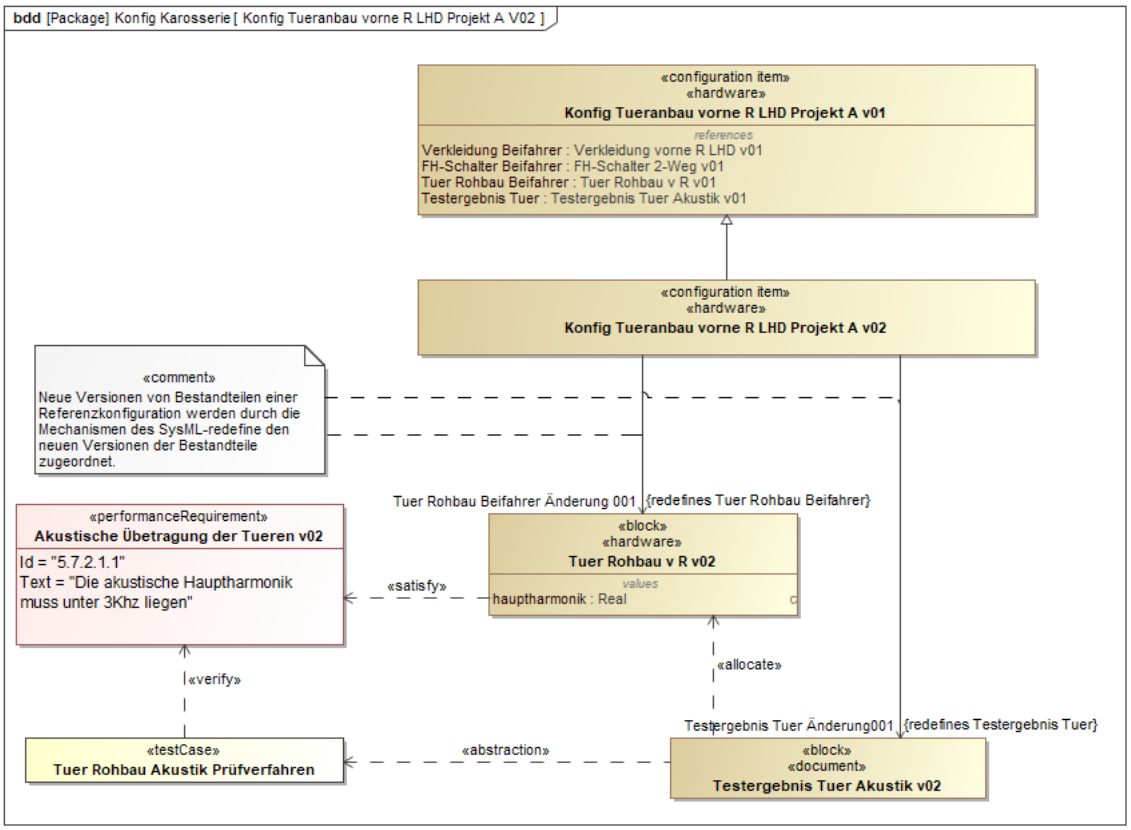

Abbildung 18: Die Referenzkonfiguration in SysML aufgebaut und im Systemmodell verwaltet, ermöglichen die Verwaltung aller Daten, die das System oder Systemelement beschreiben und dokumentieren.

Die Durchführung des Änderungsmanagements ist auf Grundlage der Referenzkonfigurationen dokumentiert und nachvollziehbar. Jede Referenzkonfiguration erbt durch diese Vorgehensweise alle Bestandteile ihrer vorherigen Version.

Das, was geändert wurde ist durch ein SysML-redefine in der neuen Referenzkonfiguration auf der neuen Version aktualisiert. Siehe die neue versionierte Instanz <Tuer Rohbau v R v02> und die entsprechende neue versionierte Instanz <Testergebnis Tuer Akustik v02> in Abbildung 18.

Die Durchführung von Konfigurationsaudits[104], die in dieser Studienarbeit nicht detailliert wird, lässt sich mit Automatisierungen im Modellierungstool umsetzen.

[104] Vgl. Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Seite 22.

6 Ausblick

In dieser Studienarbeit wurde ein Beispiel in SysML (Version 1.6) dargestellt. Die SysML (Version 2.0[105]), aktuell noch nicht veröffentlicht, wird die hier vorgestellten Vorgehensweise auch unterstützen[106]. Das in der SysML (Version 2.0) eingeführte Konzept der Variation und Variante[107] wurde in dieser Studienarbeit bereits betrachtet. Der Aufbau des Digital Threads wird mit der SysML Version 2.0 durch das API unterstützt[108]. Die gleichzeitige Verwendung von grafischen und textbasierten Modellen in SysML (Version 2.0) ermöglicht die Nutzung von Git/GitHub-Kollaborationsworkflows[109] zur Versionierung. Dennoch müssen die in dieser Studie getroffenen Festlegungen hinsichtlich der Definition der Konfigurationseinheiten übertragen und genauer spezifiziert werden. Eine spätere Weiterentwicklung dieser Studie nach den künftigen Möglichkeiten von SysML (Version 2.0) wird notwendig.

In Anbetracht der Entwicklungen in der künstlichen Intelligenz zeigt sich ein steigendes Interesse an möglichen Anwendungen von KI-Agenten im Umfeld des MBSE. Zum Zeitpunkt des Schreibens dieser Studienarbeit gibt es aber noch keine Anwendung bezüglich des Konfigurationsmanagements[110]. Diese Studienarbeit kann als Grundlage dienen für die Untersuchung von möglichen Anwendungen der KI-Agenten im Konfigurationsmanagement[111].

---

[105] The Object Management Group®(OMG®), “https://www.omg.org/index.htm”, Seite 35-41.

[106] Seidewitz, Ed, „https://github.com/Systems-Modeling/SysML-v2-Release/blob/master/sysml/src/training/35.%20Variability/Variation%20Configuration.sysml“, 30.08.2023 (Stand des letzten Zugriffs), Siehe Variation Configuration.sysml

[107] Vgl. The Object Management Group®(OMG®), “https://www.omg.org/index.htm”, Seite 45-46.

[108] Bajaj, Manas, Sanford Friedenthal, Ed Seidewitz, „Systems Modeling Language (SysML v2) Support for Digital Engineering“ in: INCOSE INSIGHT, Ausgabe: 03/2022, Seite 22.

[109] Vgl. GitHub Docs, “https://docs.github.com/en/get-started/quickstart/contributing-to-projects”, 14.10.2023 (Stand des letzten Zugriffs)

[110] Vgl. Fraunhofer-Institut für Entwurfstechnik Mechatronik IEM, „https://www.selive.de/ai-in-mbse/“, 20.08.2023 (Stand des letzten Zugriffs)

[111] Vgl. Dukino, Claudia, „https://blog.iao.fraunhofer.de/was-ist-kuenstliche-intelligenz-eine-definition-jenseits-von-mythen-und-moden/“, 26.08.2023 (Stand des letzten Zugriffs)

7 Zusammenfassung

Die Referenzkonfigurationen im Systemmodell sind nicht auf den Lebenszyklus eines bestimmten Systems begrenzt: Sie existieren oft vor dem System und werden durch die Referenzarchitektur wiederverwendet. Auch wenn der Baum von Referenzkonfigurationen sich redundant zur Struktur des Zielsystems erweisen könnte, findet erstere ihre Berechtigung in ihrem erweiterten Inhalt und langem Lebenszyklus. Die Referenzkonfigurationen in den strategischen und operativen Modellen werden auch in den meisten Fällen über den Lebenszyklus eines Systems hinaus gelten. Beide gehören in den Baum von Referenzkonfigurationen des Supra-Systemmodells, das alle Lebenszyklen umfasst. Der Aufbau eines solchen Baumes, der Modellelemente von SysML und UAFML beinhaltet, kann in einer künftigen Studie erarbeitet werden. Zu wissen, welche Version eines physischen Knotens welche Version einer Anforderung erfüllt, nach welcher Version eines Prüfverfahrens sie verifiziert wird oder welcher Version einer Simulation sie entspricht, ist nicht ausreichend; es gehört auch dazu, zu wissen, nach welcher Version eines Entwicklungsprozesses sie gestaltet wird, nach welcher Version eines Wartungsprozesses sie gewartet wird oder nach welcher Version eines Stilllegungsverfahrens sie abgemustert wird sowie unter welcher Version der Organisationsführung sie entwickelt, gewartet und stillgelegt wird.

Der Digital Thread kann die Integration von MBSE-Tools und der Digital Twin ermöglichen. Gleichzeitig kann das MBSE als Grundlage für den Aufbau des Digital Threads dienen[112]. Ein bestimmtes System ist eine Instanz, die nach einer Referenzkonfiguration entstanden ist und durch alle darunterliegenden Referenzkonfigurationen bis auf das letzte Detail verwaltet wird. Da alle Daten und ihre Semantik im Zentrum des Supra-Systemmodells liegen, ist es jederzeit möglich, auf das gesamte digitale Vermögen dieses spezifischen Systems zuzugreifen. Dies ermöglicht die Anwendung dieses Vermögens zur Erstellung des Digital Twins.

[112] Vgl. Madni, Azad M., Carla C. Madni, Scott D. Lucero, „https://www.mdpi.com/2079-8954/7/1/7“, 01.10.2023 (Stand des letzten Zugriffs), Seite 8.

8 Literaturverzeichnis

8.1 Bücher und Beiträge aus Sammelwerken

Bahns, Tammo, Sylvya Melzer, Ralf God, Dieter Krause, Ein modellbasiertes Vorgehen zur variantengerechten Entwicklung modularer Produktfamilien, Tag des Systems Engineering 2015, München 2015

Delligatti, Lenny, SysML Distilled, A Brief Guide to Systems Modeling Language, Crawfordsville, Indiana, 2013

Department of Defense, United States of America, MIL-STD-973 Military Standard Configuration Management, Falls Church, VA, 1992

Friedenthal, Sanford, Alan Moore, Rick Steiner, A Practical Guide to SySML: The Systems Modeling Language, Waltham, Ma USA, 2011

ISO (the International Organisation for Standardization), ISO10007:2017, ISO, Genf 2017

Schulte, Tim, Marc Schneider, Udo Judaschke, Daniel Batz, Systemmodelle verwalten mit ConfigML – Motive, Grundlagen und erste Konzepte einer Sprache für das modellbasierte Konfigurationsmanagement, Tag des Systems Engineering 2016, München 2016

Schulte, Tim, Samson Groß, Sebastian Langer, Lucas Kirsch, ConfigML – Erste prototypische Realisierung einer Verwaltung von Modellen mit Modellen im PLM, Tag des Systems Engineering 2017, München 2017

Walden, David D., Garry. J. Roedler, Kevin J. Forsberg, R. Douglas Hamelin, und Thomas M. Shortell, GfSE (2017) INCOSE Systems Engineering Handbuch - Ein Leitfaden für Systemlebenszyklus-Prozesse und -Aktivitäten, 2017

Walden, David D., et al, INCOSE Systems Engineering Handbook. A guide for Systems Life Cycle Processes and Activities, Sab Diego, California USA, 2023

Wheatcraft, Lou, Tami Katz, Michael Ryan, Raymond B. Wolfgang, INCOSE Needs and Requirements Manual, San Diego, California USA, 2022

Wilkiens, Tim, SYSMOD - the Systems Modeling Toolbox, 2020

8.2 Artikel aus Zeitschriften

Bajaj, Manas, Sanford Friedenthal, Ed Seidewitz, „Systems Modeling Language (SysML v2) Support for Digital Engineering.“ in: INCOSE INSIGHT, Ausgabe: 03/2022

International Council on Systems Engineering (INCOSE) in: SYSTEMS ENGINEERING VISION 2020, Ausgabe: 09/2007

McDermott, Tom; Kelly Alexander, Richard Wallace, "The Supra-Systemmodell." In: INCOSE INSIGHT Ausgabe: Volume 26/Issue 2, 06/2023

8.3 Internetquellen

Apache Software Foundation, “https://subversion.apache.org/“, 06.09.2023 (Stand des letzten Zugriffs)

Balandi, Oguzahn, „https://kobra.uni-kassel.de/handle/123456789/13754“, 26.08.2023 (Stand des letzten Zugriffs)

Bleisinger, Oliver, et al, „ https://publica-rest.fraunhofer.de/server/api/core/bitstreams/a09d6709-63ad-48de-b819-efad5387c2ab/content“, 29.07.2023 (Stand des letzten Zugriffs)

Dukino, Claudia, „https://blog.iao.fraunhofer.de/was-ist-kuenstliche-intelligenz-eine-definition-jenseits-von-mythen-und-moden/“, 26.08.2023 (Stand des letzten Zugriffs)

Fraunhofer-Institut für Entwurfstechnik Mechatronik IEM, „https://www.selive.de/ai-in-mbse/“, 20.08.2023 (Stand des letzten Zugriffs)

Gausenmeier, Jörg, Roman Dumitrescu, Daniel Steffen, Anja Czaja, Olga Wiederkehr, Christian Tschirner, https://www.hni.uni-paderborn.de/fileadmin/Fachgruppen/Seniorprofessur_Gausemeier/systemsengineerings/Studie_Systems-Engineering.pdf, 26.08.2023 (Stand des letzten Zugriffs)

GitHub Docs, “https://docs.github.com/en/get-started/quickstart/contributing-to-projects”, 14.10.2023 (Stand des letzten Zugriffs)