Konzept zur Verhaltensmodellierung mit der Systems Modeling Language (SysML) zur Simulation varianten Systemverhaltens

David Arndt[1], Sylvia Melzer[1], Ralf God[1], Manfred Sieber[2]

[1]Technische Universität Hamburg-Harburg, Institut für Flugzeug-Kabinensysteme, Nesspriel 5, 21129 Hamburg, {david.arndt, sylvia.melzer, ralf.god}@tuhh.de
[2]Airbus Operations GmbH, Cabin Core System Architecture, Kreetslag 10, 21129 Hamburg {manfred.sieber}@airbus.com

Zusammenfassung: Die Entwicklung von Produktlinien bzw. von Produktfamilien erfordert häufig eine Modellierung von unterschiedlichen Systemvarianten. Obgleich die Systems Modeling Language (SysML) für die Entwicklung komplizierter Systeme entworfen und spezifiziert worden ist, weist die SysML keine expliziten Sprachelemente für die Variantenmodellierung auf. Infolge dessen wurden SysML-spezifische Methoden, wie z.B. die *Variant Modeling with SysML* (VAMOS) vorgeschlagen, die bei der Variantenmodellierung unterstützen. Bei VAMOS jedoch liegt der Fokus beim Entwurf varianter Systemstrukturen. Variantes Systemverhalten und dessen Modellierung stand bei VAMOS nicht im Vordergrund. Die Modellierung varianten Systemverhaltens gewinnt jedoch beim *Model-Based Systems Engineering* (MBSE) zunehmend an Bedeutung, weil bei einer modellbasierten Vorgehensweise häufig auch so genannte Simulationsmodelle zur Überprüfung des korrekten Systemverhaltens erstellt werden. Um Simulationsmodelle für variantes Systemverhalten erstellen zu können, wird folglich ein Konzept zur Verhaltensmodellierung mit der SysML zur Simulation varianten Systemverhaltens benötigt. Ein solches Konzept wird in diesem Beitrag vorgestellt.

1 Einleitung

Beim modellbasierten Systems-Engineerings (MBSE) erfolgt die Spezifikation eines Systems durch ein Modell, welches eine Abstraktion des realen Systems darstellt [Alt12]. Die Nutzung von Modellen ermöglicht es, die Komplexität bei der Entwicklung von Systemen während des gesamten Entwicklungsprozesses besser zu beherrschen, was zur Vermeidung von Ungenauigkeiten bei der Systembeschreibung hinsichtlich dessen Struktur und Verhalten beiträgt und damit auch Fehler reduziert bzw. ausschließt [Ga13]. Die Modellierungssprache *Systems Modeling Language* (SysML) wurde von der *Object Management Group* (OMG) [OM14] definiert, um die modellbasierte Entwicklung von Systemen in ihren drei wichtigen Phasen Systementwurf, Implementierung und Systemintegration zu unterstützen. Deshalb berücksichtigt ein SysML-Modell

die zwei zentralen und wesentlichen Systemaspekte, nämlich dessen Struktur und Verhalten. Die Struktur zeigt alle Elemente in einem System und repräsentiert die Beziehungen zwischen ihnen. Das Verhalten zeigt, wie sich ein System über die Zeit verhält.

Eine Herausforderung besteht heute darin, dass es einen Trend zur Individualisierung von Produkten bzw. Systemen für unterschiedliche Kunden gibt [VM14, HG12, LRZ06]. Dies führt zu einer steigenden Variantenvielfalt und macht infolge dessen die Modellierung von sogenannten Varianten erforderlich. Viele Methoden zum Systementwurf, darunter auch das MBSE mit der SysML, sind häufig nur für die Entwicklung eines bestimmten, d.h. singulären Produkts oder Systems konzipiert und die Entwicklung ganzer Produktlinien bzw. -familien mit ihrer Variantenvielfalt werden von diesen daher nur ungenügend unterstützt. Es bedarf daher einer Methode, welche diese Variantenvielfalt mit entsprechenden SysML-Systemmodellen abbilden kann [Me14]. Zur Repräsentation varianter Systemstrukturen ist beispielsweise die in [We16] beschriebene Variant-Modeling-with-SysML-Methode (VAMOS-Methode) bekannt. Die Darstellung varianten Systemverhaltens stand bei der Entwicklung von VAMOS nicht im Mittelpunkt. Gerade dieser wichtige Aspekt ist allerdings unverzichtbar, wenn Varianten beim Systemverhalten in der SysML durch Verhaltenselemente repräsentiert und zur Validierung simuliert werden sollen. Ziel der hier vorgestellten Arbeiten war es daher, ein solches Konzept zur Verhaltensmodellierung und zur Simulation varianten Systemverhaltens mit der SysML zu entwerfen und zu validieren.

2 Konzept zur Modellierung varianten Systemverhaltens

Nach einer Beschreibung der bekannten VAMOS-Methode zur Modellierung varianter Systemstrukturen und der in dieser Arbeit verwendeten Modellierungswerkzeuge werden in diesem Kapitel zwei mögliche SysML-konforme Konzepte zur Modellierung varianten Systemverhaltens vorgestellt und am Beispiel eines Kabinen-Handtelefons, welches im Flugzeug zur Kommunikation zwischen dem Kabinenpersonal, dem Cockpit und mit den Passagieren eingesetzt wird, illustriert. Im darauf folgenden Kapitel 3 wird gezeigt, wie mit einem dieser Konzepte zusätzlich auch die Simulation varianten Systemverhaltens gelingt. Im letzten Kapitel 4 wird die Validierung des entworfenen Konzepts anhand eines Anzeigesystems für Passagiere beschrieben.

2.1 Methode zur Modellierung varianter Systemstrukturen

Eine Variante ist durch ein Basis-Modell und differenzierende Anteile charakterisiert, wobei das Basis-Modell quasi den Kern des Systems und die differenzierenden Anteile die Unterscheidungen von Systemkomponenten repräsentieren [Me14]. Weilkiens beschreibt in [We16] die VAMOS-Methode, die zur Modellierung von Varianten herangezogen wird. Dazu werden in der SysML vorhandene Modellelemente um variantenspezifische Elemente erweitert. Bild 1 zeigt die vorhandenen Modellelemente *Package* und

NamedElement mit den Stereotyp-Erweiterungen Variante, Variation und Variationspunkt. Ein Stereotyp ist eine Erweiterung vorhandener Modellelemente. Eine Variante ist ein Stereotyp der Metaklasse Paket (engl. *Metaclass Package*), welche alle Elemente einer Variationsmöglichkeit enthält. Eine Variation ist auch ein Stereotyp der Metaklasse Paket, welche mehrere Variantenpakete enthält. Ein Variationspunkt ist ein Stereotyp der Metaklasse *NamedElement* [We16].

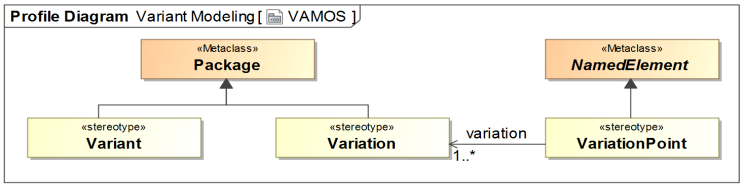

Bild 1. Semantik und Notation einer Variante (engl. *Variant*), einer Variation (engl. *Variation*) und eines Variationspunktes (engl. *VariationPoint*) [We16]

2.2 Verwendete Modellierungswerkzeuge

Rational Rhapsody Developer (Version 8.14) aus dem Hause IBM [IB17] ist ein aktuelles Modellierungswerkzeug, dessen Ursprung sich auf das ältere Werkzeug *Statemate* aus dem Jahr 1986 zurückverfolgen lässt. *Statemate* war damals für die Modellierung und Simulation von Zustandsmaschinen entwickelt worden. Mit Aufkommen der Spezifikation der *Unified Modeling Language* (UML) im Jahr 1996 entstand im selben Jahr daraus das Werkzeug *Rational Rhapsody*. Im Gegensatz zu *Statemate* basiert dieses Werkzeug auf einem objektorientierten Paradigma und wurde für die Entwicklung von Software mithilfe der UML weiterentwickelt. Für die Systementwicklung bietet *Rational Rhapsody* seit einigen Jahren auch die Möglichkeit, sich zu Beginn eines Projektes, alternativ zur Nutzung der UML-Elemente, auch für die später spezifizierten SysML-Sprachelemente entscheiden zu können. In dieser Arbeit wurde die SysML ausgewählt. Unter SysML bildet *Rational Rhapsody* in seiner Version 8.14 nicht alle Sprachelemente spezifikationskonform ab. Verhaltensdiagramme sind syntaktisch und semantisch sehr nah an der UML-Spezifikation [CD05]; dies gilt analog auch für die Verhaltensbeschreibungen mit der Auswahl SysML. Weiterhin lässt sich mit dem Werkzeug aus einem Modell Quellcode in verschiedenen Programmiersprachen (z.B. C/C++ oder Java) und für verschiedene Plattformen (z.B. Linux, QNX, Windows oder VxWorks) erzeugen und es lassen sich unter anderem Aktivitäts- und Zustandsdiagramme, ausgehend vom kompilierten Quellcode, simulieren.

Cameo Systems Modeler (Version 18.4) aus dem Hause No Magic [NM14] wurde als Modellierungswerkzeug spezifisch für die erstmals im Jahr 2007 spezifizierte SysML entwickelt. Dieses Werkzeug bildet in seiner Version 18.4 überwiegend alle SysML-Sprachelemente spezifikations-konform ab [Me17]. Zur Simulation von Verhaltensdiagrammen nutzt *Cameo Systems Modeler* das

zusätzlich erforderliche Plug-In *Cameo Simulation Toolkit*. Dabei ist das Toolkit so aufgebaut, dass eine Teilmenge der UML-Elemente auf der OMG fUML (Foundation Subset for Executable Models) und W3C SCXML (State Chart XML) Standards basieren, sodass überhaupt eine Simulation eines Modells erstellt werden kann. Mit *Cameo Systems Modeler* lässt sich ein *Graphical User Interface* (GUI) erstellen. Dessen Simulation beruht auf *Java Swing*. Beim Starten einer solchen Simulation wird ein zur Laufzeit lauffähiges *Java Swing User Interface* generiert. Dies legt den Schluss nahe, dass auch für die Simulation der SysML-Verhaltensdiagramme zur Laufzeit lauffähige Java-Objekte generiert werden und bei der Animation involviert sind.

2.3 Modellierung varianten Systemverhaltens

Zur Ausrüstung von Verkehrsflugzeugen gehört ein Kabinenmanagementsystem mit einem Handtelefon (siehe Bild 2, rechts), welches der Kommunikation zwischen der Kabinencrew, der Cockpitcrew und mit den Passagieren dient. Die Art der Kommunikation wird über ein Bedienfeld (B) gewählt, z.B. *Passenger Address* (PA, Ansage an die Passagiere) und *Interphone* (Int, Telefongespräch zwischen den Crewmitgliedern). Da Verkehrsflugzeuge von verschiedenen Fluggesellschaften betrieben werden, gibt es je nach Betreiber unterschiedliche Anforderungen an die Umsetzung der Funktionen auf einem Kabinen-Handtelefon. Dies hat wiederum zur Folge, dass das Bedienfeld für die verschiedenen Kunden unterschiedlich umzusetzen ist. Zur Vermeidung einer Erstellung aufwendiger individueller Systemmodelle pro Kunde wird daher ein Konzept zur Modellierung (d.h. zur Spezifikation) von Varianten mit variantem Systemverhalten benötigt. Da für den Test dieser Spezifikation auch die Simulierbarkeit der Modelle möglich sein soll, wird für die Konzeptumsetzung nicht nur eine graphische Repräsentation der SysML-Sprachelemente gefordert, sondern auch die Unterstützung durch Modellierungswerkzeuge, welche die Simulation des modellierten Systemverhaltens ermöglichen.

Das Verhalten eines Kabinen-Handtelefons (engl. *handset*) lässt sich mit der in Bild 2 (links) dargestellten Zustandsmaschine (state maschine [stm]) beschreiben. Die darin beinhalteten Zustände repräsentieren die verschiedenen Betriebsmodi eines Kabinen-Handtelefons: ausgeschaltet (Zustand *handsetOff*), eingeschaltet (Zustand *handsetOn*), aufgehängt (Zustand *hookedOn*) und abgenommen (Zustand *hookedOff*). In dem Zustand *hookedOff* können mit dem Handtelefon weitere Funktionen bedient werden, z.B. die Passagieransage (Zustand *PAActive*) oder ein Telefongespräch mit anderen Crewmitgliedern (Zustand *interphoneActive*).

Für die Erstellung eines Variantenmodells, das um eine zweite Verhaltensbeschreibung für eine andere Fluggesellschaft erweitert werden soll, bietet sich die auf Generalisierung/Spezialisierung basierende Methode VAMOS an, die SysML-konform angewendet werden kann. Die SysML 1.4 ist jedoch hinsichtlich des Verhaltens semantisch nicht ausreichend für ein simulierbares Modell mit variantem Systemverhalten spezifiziert, wie der folgende Absatz beschreibt. Aus diesem Grund kommt es dazu, dass bei verschiedenen SysML-Werkzeugen die SysML-Verhaltenselemente teilweise unterschiedlich eingesetzt

werden müssen und dass dadurch dann die Simulationen anders ausgeführt werden. Das hat wiederum zur Folge, dass hinsichtlich der Anforderung eines simulierbaren Verhaltensmodells, je nach verwendetem Werkzeug, unterschiedliche SysML-1.4-konforme Konzepte entworfen werden müssen.

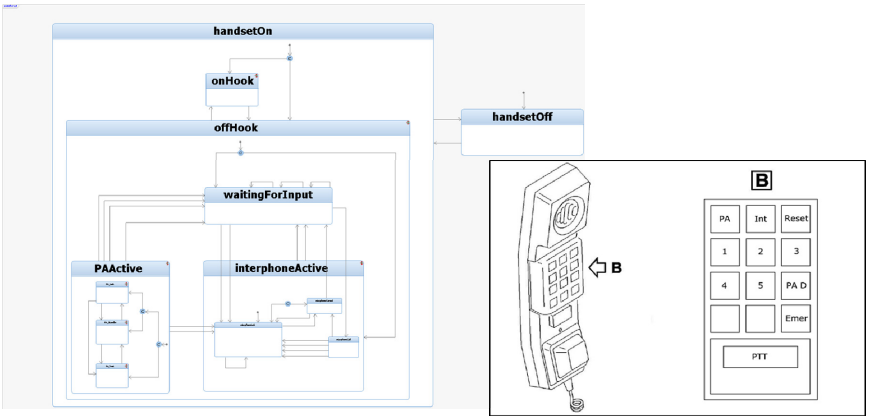

Bild 2: Kabinen-Handtelefon (rechts) [A330] mit einem Ausschnitt des Bedienfeldes und Modellierung des Systemverhaltens in Form eines Zustandsdiagramms mit dem Modellierungswerkzeug *Rational Rhapsody* [IB17] (links)

Ein mit der SysML 1.4 konformes Konzept zur Variantenmodellierung ist links in Bild 3 syntaktisch und semantisch korrekt formuliert und graphisch repräsentiert. Mit dem SysML-Werkzeug *Rational Rhapsody* lässt sich dieses Konzept jedoch weder syntaktisch noch semantisch in einem Blockdefinitionsdiagramm umsetzen und mit dem SysML-Werkzeug *Cameo Systems Modeler* ist lediglich eine syntaktische Anwendung der Spezialisierung einer Zustandsmaschine in einem Blockdefinitionsdiagramm möglich. Hinsichtlich einer werkzeugtechnischen Umsetzung der SysML besteht daher der Bedarf – so wie es auch ein klassischer SysML-Ansatz vorsieht – eine Verhaltensbeschreibung einem Block zuzuordnen. Dementsprechend wurde, getrieben durch die werkzeugtechnischen Restriktionen, nach einem anderen Konzept gesucht, welches mit beiden hier verwendeten Werkzeugen von IBM [IB17] und No Magic [NM14] umsetzbar ist.

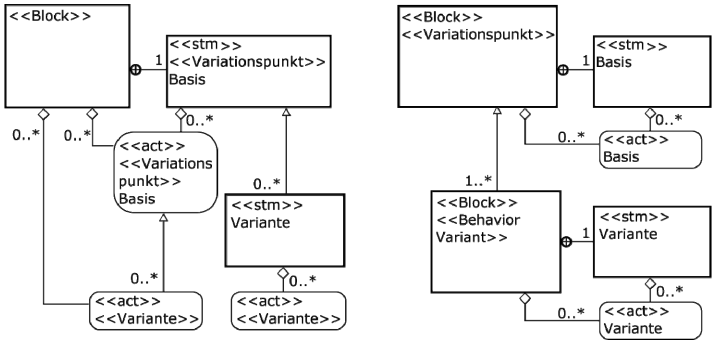

Bild 3: SysML-konformes, jedoch mit den verwendeten Werkzeugen nicht umsetzbares Konzept zur Verhaltensmodellierung (links) und angepasstes, werkzeugtechnisch umsetzbares Konzept (rechts)

In der rechten Hälfte von Bild 3 ist dieses für beide Werkzeuge angepasste Konzept dargestellt. Hier ist eine einfache Unterscheidung von Varianten hinsichtlich der Basis und den differenzierenden Anteilen auch bei komplizierten Systemen möglich. Ein Block mit dem Stereotyp *VariationPoint* enthält die Basis-Elemente. Weitere spezialisierte Blöcke mit dem Stereotyp *BehaviorVariant* der Metaklasse *NamedElement* enthalten die Varianten mit den differenzierenden Anteilen. Dabei ist jede Zustandsmaschine einem Block zugeordnet. Für eine optimale Strukturierung von Basis und differenzierender Anteile sei auf den in [Kr14] beschriebenen Ansatz verwiesen. Eine entsprechende modellbasierte Umsetzung findet sich bei [Ba15].

Die entsprechende Zustandsmaschine für ein Kabinen-Handtelefon eines anderen Kunden auf Basis des in Bild 3 rechts gezeigten Konzeptes zur Verhaltensmodellierung ist in Bild 4 links dargestellt. Beim Verhalten gibt es viele Gemeinsamkeiten zum im Bild 2 links dargestellten System, die definitionsgemäß zur Basis zählen (grau dargestellt) und einige Abweichungen (graublau hervorgehoben), die entsprechend als differenzierende Anteile klassifiziert werden. In diesem Beispiel gehören mehrere Zustände und Transitionen zu den differenzierenden Anteilen.

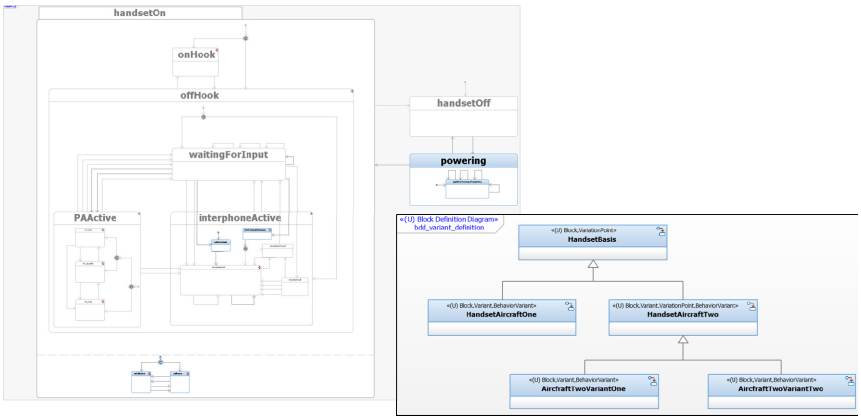

Bild 4: Struktur eines Handtelefon-Systems für zwei Flugzeuge (*aircraft one* bzw. *two*) mit zwei Varianten beim zweiten Flugzeug (rechts) und variantes Zustandsdiagramm (links) mit beim Verhalten gleichbleibenden (grau dargestellt) und differenzierenden Anteilen (graublau hervorgehoben).

Wird das Werkzeug *Cameo Systems Modeler* verwendet, so kann zwar die in Bild 4 rechts dargestellte Systemstruktur, jedoch nicht die mit dem Werkzeug *Rational Rhapsody* erzeugte Zustandsmaschine wie in Bild 4 links modelliert werden. Wenn sich hier zwei Zustandsmaschinen nur hinsichtlich eines einzigen Elementes z.B. einer Transition, unterscheiden, so ist es bei diesem Werkzeug erforderlich, dass eine neue Zustandsmaschine erstellt wird. Folglich sind dann im Modell mit den vielen Zustandsmaschinen viele invariante Elemente mehrfach vorhanden, welche aufwendig dokumentiert und gepflegt werden müssen. Andererseits lässt das Werkzeug *Cameo Systems Modeler* auch eine alternative modelltechnische Umsetzung von varianten Zustandsmaschinen zu, was jedoch vom Modellierer fordert, dass dieser zu Beginn der Modellierung alle gleichbleibenden Elemente die zur Basis gehören und alle differenzierenden Anteile bereits kennt und festlegt. Ein Umstand, welcher für Systeme mit einem Lebenszyklus praktisch nie erfüllt werden kann. Dieses Beispiel zeigt, dass Konzepte der Verhaltensmodellierung von Varianten konform zur SysML-1.4-Spezifikation unter Nutzung der beiden Werkzeuge *Rational Rhapsody* und *Cameo Systems Modeler* auf die eine oder andere Art und Weise umgesetzt werden können. In beiden Fällen folgt jedoch die Umsetzung den Möglichkeiten und Restriktionen der verwendeten Werkzeuge. Im folgenden Kapitel soll nun auf Basis der jeweiligen werkzeugspezifischen Modellierung gezeigt werden, wie zu Validierungszwecken jetzt in einem nächsten Schritt eine Simulation der erzeugten Modelle gelingt.

3 Simulation varianten Systemverhaltens

Für ein mit dem Werkzeug *Cameo Systems Modeler* erzeugtes Modell mit Verhaltensvarianten gemäß Kapitel 3, muss im Simulationsmodell für jede Variante ein eigenes Zustandsdiagramm mit vielen invarianten Elementen erstellt werden. Das liegt daran, dass Elemente (z.B. Zustände) einer Zustandsmaschine nicht gleichzeitig in einer anderen Zustandsmaschine erscheinen dürfen.

Anders kann mit dem Werkzeug *Rational Rhapsody* ein Modell mit Verhaltensvarianten gemäß Kapitel 3 innerhalb eines einzigen Zustandsdiagramms generiert werden, indem im vorliegenden Fall weitere Stereotypen verwendet werden: *Static* und *Varies*. *Varies* liefert für eine Simulation die notwendige Information, dass ein Block mit dem Stereotypen *VariationPoint* Varianten enthält. *Static* liefert die notwendige Information, dass bei der Erzeugung von Quellcode für ein Variantenmodell diese statischen Basiselemente zuzüglich der jeweiligen differenzierenden Anteile zur Simulation herangezogen werden müssen. Ohne die Verwendung dieser beiden präzisierenden Stereotypen sind eine Quellcodeerzeugung und Simulation nicht möglich.

Ein für ein Kabinen-Handtelefon mit variantem Systemverhalten und dem Werkzeug *Rational Rhapsody* erzeugtes Simulationsmodell ist in Bild 5 dargestellt. Soll ein hinsichtlich seines Verhaltens nicht variantes Kabinen-Handtelefon simuliert werden, so kann über die erzeugte Animation der Zustandsdiagramme eine Validierung des Systems ohne weiteres direkt vorgenommen werden. Sofern Simulationsmodelle mit Verhaltensvarianten simuliert werden sollen, so erfordert dies für die Animation eine vollständige Überschreibung der spezialisierten Zustandsdiagramme einer Variante. Diese Notwendigkeit des Überschreibens entkoppelt die Zustandsdiagramme von der Basis. Allerdings werden hierdurch alle Elemente in der Variante veränderbar. Weiterhin werden Änderungen in der Basis nach dem Überschreiben nicht mehr an die Variante vererbt. Somit ist die Konsistenz eines simulierbaren und animierten Modells nicht sichergestellt, da Änderungen der Basis nicht mehr an die Varianten vererbt werden.

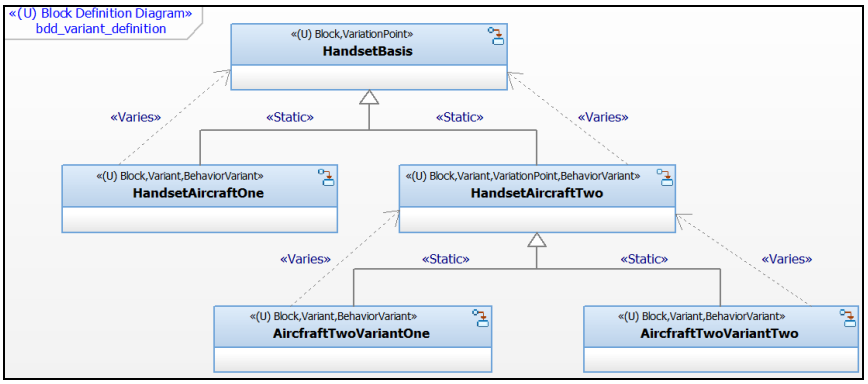

Bild 5: Blockdefinitionsdiagramm (bdd) eines simulierbaren Variantenmodells mit den beim Werkzeug *Rational Rhapsody* verwendeten beiden Stereotypen *Static* und *Varies*

4 Validierung des entworfenen Konzeptes

In diesem Beitrag wurde am Beispiel eines im Flugzeug vorhandenen Kabinen-Handtelefons gezeigt, dass es unter Nutzung des Werkzeuges *Rational Rhapsody* möglich ist, ein mit der SysML-1.4-Spezifikation konformes Konzept zur Modellierung und zur Simulation varianten Systemverhaltens umzusetzen. Die generelle Anwendbarkeit dieses werkzeugspezifisch entworfenen Konzepts für andere Flugzeugsysteme wurde u.a. anhand der Modellierung der Varianten eines Passagieranzeigesystems (*passenger lighted signs*, siehe Bild 6) vorgenommen. Im Zustandsdiagramm gehören zu den Zuständen eines Passagieranzeigesystems: *power*Off und *powerOn*. Im Zustand *powerOff* ist das System ohne Stromversorgung und somit ausgeschaltet, im Zustand *powerOn* ist das System betriebsbereit. Im Zustand *powerOn* gibt es weitere Zustände, wobei zur Basis des Systems die Zustände *noSmokingSignOn* und *noSmokingSignOff* zählen. Zwei Transitionen (*Switch on by cabin*, *Switch off by cabin)* und eine interne Transition (*checkSign*) gehören bei Variante 1 nicht zur Basis (blaugrau markiert). Variante 2 und Variante 3 haben im Vergleich zu Variante 1 noch weitere Zustände (*noMobileSignOn*, *noMobileSignOff*, *noPanelAndMobilOn*) und Transitionen, die zu den differenzierenden Anteilen zählen.

Das Konzept der Verhaltensmodellierung konnte sehr gut auf dieses System übertragen werden und die eben beschriebenen Varianten beim Systemverhalten konnten ganz analog zum Beispiel des Kabinen-Handtelefons modelliert und simuliert werden.

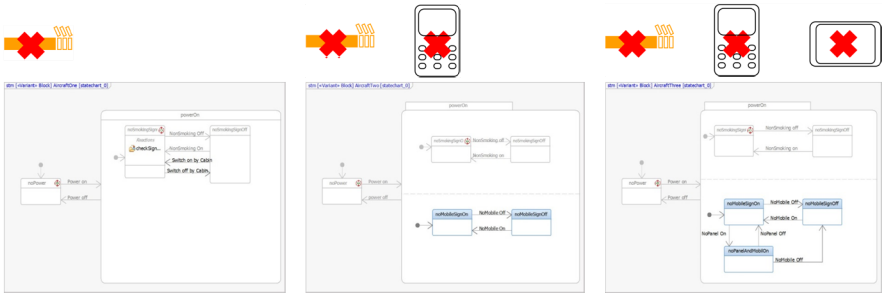

Bild 6: Varianten des Anzeigesystems für Passagiere mit den zugehörigen Zustandsdiagrammen unter Verwendung des in Bild 3 rechts dargestellten Konzeptes. Jeweils grau dargestellt sind die gleichbleibenden und graublau hervorgehoben sind die differenzierenden Anteile des Anzeigesystems.

Ziel dieser hier vorgestellten Arbeit war es, ein SysML-1.4-konformes Konzept zur Verhaltensmodellierung so zu entwerfen, dass sich dieses mit SysML-

Werkzeugen, hier *Rational Rhapsody* und *Cameo System Modeler*, umsetzen lässt. Die Möglichkeit zur anschließenden Simulation der varianten Verhaltensdiagramme war dabei die wichtige Randbedingung. Es konnte gezeigt werden, dass sich mit der VAMOS-Methode zwar SysML-1.4-konforme Konzepte zur Verhaltensmodellierung definieren lassen, dass sich diese aber nicht alle mit *Rational Rhapsody* und *Cameo Systems Modeler* umsetzen lassen. Folglich wurde für *Rational Rhapsody* ein werkzeugspezifisch adaptiertes Konzept entworfen, welches einerseits die Modellierung varianten Verhaltens ermöglicht und schließlich auch eine anschließende Simulation der Verhaltensdiagramme erlaubt. Es gelingt damit die Modellierung varianten Systemverhaltens bei heute üblichen Produktlinien bzw. Produktfamilien. Grundsätzlich ist die Anwendbarkeit des Konzeptes über Werkzeuggrenzen hinaus nicht eingeschränkt, weil das Konzept neben der SysML-Konformität auch konform zum objektorientierten Paradigma ist.

Danksagung

Diese Untersuchungen wurden im Rahmen der Masterarbeit von David Arndt an der TUHH in Kooperation mit der Airbus Operations GmbH durchgeführt. Der modellbasierte Systementwurf von Kabinensystemen ist auch Teil des LuFo V-2 Forschungs-projektes ConCabInO *Information Centric Operation of Future Connected Cabin* [Co16] an welchem das Institut für Flugzeug-Kabinensysteme beteiligt ist. Dieses Projekt wird aufgrund eines Beschlusses des Deutschen Bundestags durch das Bundesministerium für Wirtschaft und Energie (BMWi) gefördert.

Literaturverzeichnis

[A330] Airbus: A330 Flight Crew Operating Manual (FCOM)

[Al12] Alt, O.: Modellbasierte System-Entwicklung mit SysML. Hanser Verlag München, 2012.

[Ar17] Arndt, D.: Untersuchungen zur Variantenmodellierung mit der SysML für Systemverhalten, Masterarbeit, Technische Universität Hamburg-Harburg, 2017.

[Ba15] Bahns, T.; Melzer, S.; God, R.; Krause, D.: Ein modellbasiertes Vorgehen zur variantengerechten Entwicklung modularer Produktfamilien, Tagungsband zum Tag des Systems Engineering (Hrsg.: Muggeo, C.; Schulze, S. O.), S. 141-150, 2015.

[CD05] Crane M.L., Dingel J.: UML Vs. Classical Vs. Rhapsody Statecharts: Not All Models Are Created Equal. In: Briand L., Williams C. (Eds.) Model Driven Engineering Languages and Systems. MODELS 2005. Lecture Notes in Computer Science, Vol 3713, Springer, Berlin, Heidelberg, 2005.

[Co16] ConCabInO (Information Centric Operation of Future Connected Cabin), 2016.

[Ga13] Gausemeier, J.; Dumitrescu, R.; Steffen, D.; Czaja, A.; Wiederkehr, O.; Tschirner, C.: Systems Engineering in der industriellen Praxis (Hrsg.: Universität Paderborn - Heinz Nixdorf Institut, Fraunhofer IPT, Unity Consulting & Innovation), Paderborn, 2016.

[HG12] Hintze, H.; God, R.: Ansatz für einen Systems Security Engineering Prozess zur Entwicklung eines Kabinenmanagementsystems der nächsten Generation. Deutscher Luft- und Raumfahrtkongress, Berlin, 2012.

[IB17] IBM: Rational Rhapsody Designer for Systems Engineers: http://www-03.ibm.com/software/products/de/ratirhapdesiforsystengi, 2017.

[Kr14] Krause, D.; Beckmann, G.; Eilmus, S.; Gebhardt, N.; Jonas, H.; and Rettberg, R.: Integrated Development of Modular Product Families – a Methods Toolkit. In Simpson, T.W.; Jiao, J.; Siddique, Z; Hölttä-Otto, K. (Hrsg.): Advances in product family and product platform design: Methods & applications, S. 245–269, Berlin Springer, 2014.

[LRZ06] Lindemann, U.; Reichwald, R.; Zäh, M.F.: Individualisierte Produkte – Komplexität beherrschen in Entwicklung und Produktion. Springer-Verlag Heidelberg, 2006.

[Me14] Melzer, S.; God, R.; Kiehl, T.; Möller, R.; Wessel, M.: Identifikation von Varianten durch Berechnung der semantischen Differenz von Modellen, Tagungsband zum Tag des Systems Engineering (Hrsg.: Maurer, M.; Schulze, S. O.), S. 279-288, 2014.

[Me17] Melzer, S.: unveröffentlichte Ergebnisse, 2017.

[NM14] No Magic: Cameo Systems Modeler. http://www.nomagic.com/products/cameo-systems-modeler.html, 2014.

[OM14] OMG: OMG Systems Modeling Language. http://www.omgsysml.org, 2014.

[VM14] Verband Deutscher Maschinen- und Anlagenbauer e.V.; McKinsey&Company: Zukunftsperspektive deutscher Maschinenbau. Erfolgreich in einem dynamischen Umfeld agieren, Br. VDMA, 2014.

[We16] Weilkiens, T.: Variant Modeling with SysML. MBSE4U Booklet Series, 2016.