*Draft published at VDI Elektronik im Kfz, Oktober 2009, Baden-Baden*


# Modellbasierte Anforderungsanalyse für die Entwicklungen variantenreicher Systeme

## Erfahrungen mit modellbasierten Spezifikationen unter Variantenmanagement für Fahrzeugfunktionen

Dr.-Ing. **C. Robinson-Mallett**, Berner&Mattner Systemtechnik, Berlin
Dipl.-Math. **J. Köhnlein**, Daimler AG, Sindelfingen
Dr. **J. Wegener**, Berner&Mattner Systemtechnik, Berlin
Dr. **M. Grochtmann**, Berner&Mattner Systemtechnik, Berlin

**Kurzfassung**

In der Automobilindustrie stellt die hohe Vielfalt an funktionalen Konfigurationsmöglichkeiten in Kombination mit immer variantenreicheren Produktpaletten eine enorme Herausforderung dar, auch weiterhin die gesteckten Termin- und Qualitätsziele trotz hohem Zeit- und Kostendruck zu erreichen. Dieser Beitrag präsentiert ein Verfahren und eine Toolumgebung zur effizienten Beherrschung einer großen Variantenvielfalt in den frühen Analyse- und Spezifikationsphasen einer Systementwicklung. Das vorgestellte Verfahren kombiniert die Vorteile der modellbasierten Anforderungsanalyse mit denen eines leistungsfähigen Variantenmanagements für Anforderungsspezifikationen. Das vorgestellte Verfahren der modellbasierten Anforderungsanalyse mit Variantenmanagement wird durch eine Toolumgebung in Telelogic DOORS unterstützt, die durch einen hohen Automatisierungsgrad effizientes Bearbeiten von Anforderungsspezifikationen ganzer Produktfamilien ermöglicht. Verfahren und Tool wurden bereits in mehreren Entwicklungsprojekten in der Automobilindustrie erfolgreich eingesetzt und validiert. Neben den Ergebnissen und Erfahrungen aus diesen Projekten wird auch die praktische Vorgehensweise am Beispiel eines einfachen Fahrassistenzsystems demonstriert und erläutert.

Durch Einsatz der modellbasierten Anforderungsanalyse mit Variantenmanagement können Zeit und Aufwand zur Erstellung von Anforderungsspezifikationen ganzer Produktfamilien messbar reduziert werden. Darüber hinaus liefert die Analysemethode redundanzarme, modellbasierte Spezifikationen mit, im Vergleich zu unsystematischeren Herangehensweisen, deutlich verbesserter Änderbarkeit und Analysierbarkeit. In allen folgenden Entwicklungs- und Qualitätssicherungsphasen macht sich zudem die Qualität der Spezifikationsdokumente positiv bemerkbar und stellt einen ausgezeichneten Anknüpfungspunkt für weitere modellbasierte Entwicklungsschritte dar.

*Draft published at VDI Elektronik im Kfz, Oktober 2009, Baden-Baden*


**Abstract/Summary (engl.), 1 page**

The trend of constantly growing numbers of product variants and features in the automotive industry makes the improvement of analysis and specification techniques a key enabler to saving costs and time. Specifically, driver assistance functions which are introduced to the market in a rapidly growing number of features and variations provide a good example: the development of a single generic driver assistance function applicable to a whole family of car variants can help to decrease costs and time to market significantly. However, the introduction of a product line approach into a large car-manufacturer's electronics development process is a challenging task, prone to human error, with the risk of spreading a single fault over a whole platform of car variants.

In this contribution we present our experiences and results using model-based and variant-management concepts for requirements analysis and specification of driver assistance functions at Daimler AG supported by a self-developed extension tool-box to DOORS.

The car-manufacturer analyses and specifies requirements and acceptance criteria and tests for a family of system variants and integrates the ready developed system. A system supplier develops and delivers the demanded system on the basis of the commited requirements specification usually separated for each single variant. In such a development process the use of a tool-supported variant management can significantly ease the control over a large number of system variations. Therefore, the presented concept introduces the following techniques and activities into a development process:

- highly modular specification structures to ease requirement transformations
- selection and transformation of requirements to variant-specific needs
- generation of variant-specific requirements books from a generic specification
- centralized management of variant-specific numerical and textual parameters
- standard-conforming tracing capabilities over generic and variant's requirements
- automatic analysis and creation of inter-requirement dependencies
- automatic indexing and management of keyword definitions
- automatic referencing and management of signal definitions
- management of test-specifications over generic and variant's requirements
- automatic referencing of test-cases
- automatic coverage analysis and reporting
- support for variant-specific test-case implementation and execution

To our knowledge the presented concept and tool-support provide a solution of unique usability and effectiveness for managing large numbers of variants during system analysis and specification. The concept and the DOORS extension tool-box have been proven in a number of industrial projects in automotive, avionics and rail-way industries. An example of a simplified driver-assistance system demonstrates our promising results.

## 1. Einleitung

Die wachsende Funktionsvielfalt und die immer variantenreicheren Produktpaletten stellen eine bedeutende Herausforderung an die effiziente und erfolgreiche Entwicklung von Systemen im Automobil dar: beispielsweise führt eine Fahrzeugentwicklung mit 100 optionalen Funktionen verteilt auf 15 Modellvarianten bereits zu einer Vielfalt von bis zu 100.000 Varianten. Verschärfend kommt hinzu, dass auch die Funktionen selbst, aufgrund unterschiedlicher physikalischer Randbedingungen und Parameter, zwischen den Produktvarianten, z.B. Fahrzeug-Abmaße oder Leistungs-Parameter, voneinander abweichen können. Dieser Beitrag stellt ein Verfahren und eine Toolumgebung vor, um die Komplexität, den Bearbeitungsaufwand und die Bearbeitungszeit in den frühen Phasen einer variantenreichen Systementwicklung zu minimieren.

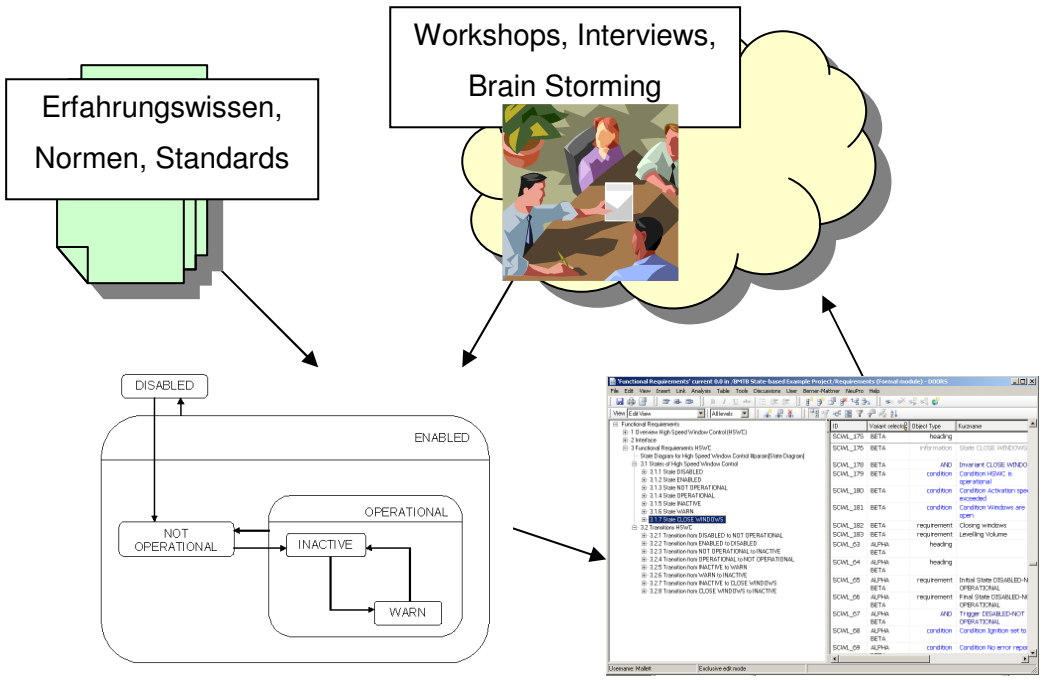


Bild 1: Erstellungsprozess einer modellbasierten Anforderungsspezifikation

In der Analysephase einer Systementwicklung werden Anforderungen an das angestrebte System erhoben und in einer oder mehreren Anforderungsspezifikationen dokumentiert [1]. Eine *Anforderungsspezifikation* kann aus diversen *Spezifikationsobjekten*, z.B. Informationen, Anforderungen oder Bedingungen, bestehen. Dieser Beitrag beschränkt sich auf die Analyse und Dokumentation funktionaler Anforderungen. Die angewendete Methode bietet allerdings auch Vorteile für die Spezifikation vollständiger Systeme, einschließlich Software-, Hardware- und nicht-funktionaler Anforderungen. Eine vollständige Erläuterung der Behandlung solcher Fälle würde allerdings den Rahmen dieses Beitrags überfordern und wird Gegenstand eines zukünftigen Beitrags sein.

Das vorgestellte Verfahren kombiniert die Nutzung von Zustandsautomaten zur Strukturierung von Anforderungsspezifikationen mit einem Verfahren zum Variantenmanagement, bestehend aus einem Selektionsverfahren, um Anforderungen einer Variante zuzuordnen, und einem Parameterersetzungsverfahren, um generische Anforderungen in variantenspezifische Anforderungen zu transformieren.

In Bild 1 wird die Vorgehensweise zur Erstellung einer *modellbasierten Anforderungsspezifikation* dargestellt. In einem initialen Analyseschritt, u.a. bestehend aus der Bestandsaufnahme, Workshops und Interviews, wird ein Zustandsautomat identifiziert, der das Verhalten der zu entwickelnden Funktion auf hohem Abstraktionsniveau beschreibt. Anschließend wird eine modellbasierte Spezifikation erstellt, welche die funktionalen Anforderungen in den Zusammenhang des Zustandsautomaten stellt. In einem nächsten Schritt werden die Anforderungen typisiert und vereinzelt. In einem iterativen Prozess können nun neue Anforderungen in die vorgegebene Struktur eingeordnet werden. Infolge der Typisierung und Strukturierung der Anforderungen besitzt die modellbasierte Anforderungsspezifikation einen funktionalen Zusammenhang der folgende automatisierte Analyse- und Änderungsschritte ermöglicht oder deutlich vereinfacht.

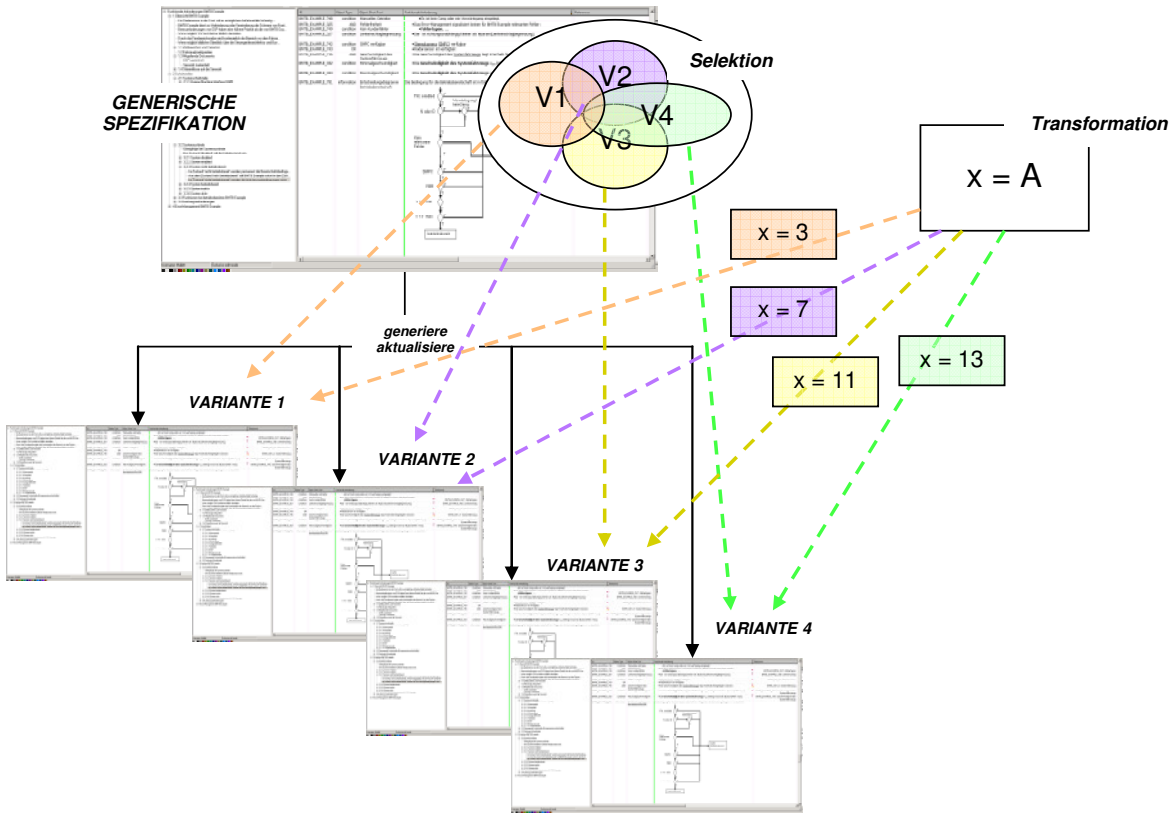


Bild 2: Beispielhafte Darstellung einer Spezifikation mit Variantenmanagement

In Bild 2 wird ein Beispiel für das Variantenmanagement einer generischen Anforderungsspezifikation dargestellt, aus der nach Bedarf vier varianten-spezifische Spezifikationen automatisch generiert oder aktualisiert werden können.
Durch *Selektion* von variantenspezifischen Anforderungen aus der generischen Spezifikation wird es ermöglicht, den Funktionsumfang und das Lastenheft in seiner Gesamtheit einer Variante anzupassen. Durch *Transformation* von Anforderungen wird es ermöglicht eine Anforderung unabhängig von varianten-spezifischen Merkmalen zu spezifizieren, und im Bedarfsfall automatisch eine variantenspezifische Anforderung zu generieren. Die verbesserte funktionale Zusammenhang der modellbasierten Spezifikation ermöglicht hierbei eine weitgehende Automatisierung der Entscheidungen, welche Anforderungen infolge der Änderung entfernt oder geändert werden müssen.

**2. Beispiel einer vereinfachten Fahrassistenzfunktion**

Die Funktion High Speed Window Control (HSWC) soll als experimentelle Variante ALPHA und für die Serie als Variante BETA entwickelt werden.

High-Level Anforderungen HSWC ALPHA

- Bei schneller Fahrt von mehr als 120 km/h sollen die offenen Fenster des PKW detektiert und der Fahrer durch eine optische Meldung im Kombi gewarnt werden.
- HSWC ist erst bei Geschwindigkeiten über 90 km/h betriebsbereit.
- Fehler sollen innerhalb der Funktion behandelt werden und im Fehlerspeicher abgelegt werden.

High-Level Anforderungen HSWC BETA

- Bei schneller Fahrt von mehr als 150 km/h sollen die Fenster des PKW automatisch geschlossen und gleichzeitig die Lautstärke der Stereoanlage auf ein festgelegtes Maß reduziert werden.
- Bei Deaktivierung der rückwärtigen Fensterheber oder bei detektierten Fondpassagieren, sollen die Fenster nicht automatisch geschlossen werden.
- HSWC ist erst bei Geschwindigkeiten über 100 km/h betriebsbereit.

Allgemeine Anforderungen

- Fehler können durch das übergeordnete Fehlermanagement behandelt werden und führen zu einer Abkopplung der Funktion HSWC von anderen Systemen.
- Die Funktion soll ein- bzw. auskodiert werden können.

Im Folgenden wird die modellbasierte Anforderungsanalyse mit Variantenmanagement anhand repräsentativer Auszüge der Anforderungsspezifikation der Funktion HSWC demonstriert. Die vollständige Spezifikation kann als Projektarchiv für Telelogic DOORS von www.berner-mattner.com/produkte bezogen werden.

**3. Statecharts**

Statecharts sind eine grafische Notation für hierarchische, endliche Zustandsautomaten, die sich für die Beschreibung von zustandsbehaftetem Systemverhalten eignen. Im Rahmen des hier vorgestellten Verfahrens hat sich die Verwendung einer Teilmenge von Statecharts bewährt, die im Folgenden beschrieben wird. Eine ausführlichere Beschreibung von Statecharts kann in [2] gefunden werden.

Die Knoten eines Statecharts stellen eine endliche Menge von Systemzuständen dar, während die Kanten die Übergänge zwischen den Zuständen repräsentieren. Ein Statechart kann eine endliche Menge von Eingangssignalen verarbeiten und produziert eine endliche Menge von Ausgangssignalen. Statecharts können durch Daten beliebigen Typs erweitert werden.

Ein *Zustand* kann eine endliche Menge *Unterzustände* beinhalten, von denen einer als initial gekennzeichnet wird, d.h. bei Betreten dieses *hierarchischen Zustands* wird der initiale Unterzustand eingenommen. Ein Übergang in einen hierarchischen Zustand endet entweder in einem der Unterzustände oder im initialen Zustand. Ein *atomarer Zustand* beinhaltet keine Unterzustände. Die *Zustandsinvariante* gibt an, welche Wertebelegungen der *Datenerweiterung* erlaubt sind, solange sich das System oder die Funktion in dem jeweiligen Zustand befindet.

Ein *Zustandsübergang* kann zwischen einem Start- und einem Endzustand spezifiziert werden. Ein *gruppenweiser Zustandsübergang* kann von einem hierarchischen Zustand ausgehend spezifiziert werden und kann aus jedem der entsprechenden Unterzustände erfolgen. Jeder Zustandsübergang besitzt einen *Trigger*, der Auslösebedingungen über Eingangssignale und *Datenerweiterung* definieren kann. Weiterhin kann ein Zustandsübergang Datenoperationen und Ausgangssignale besitzen, die ausgelöst werden, sobald die Bedingung des Triggers im Ausgangszustand des Übergangs eintritt.

Die hier eingeführten Begriff sollen am folgenden Beispiel demonstriert werden.

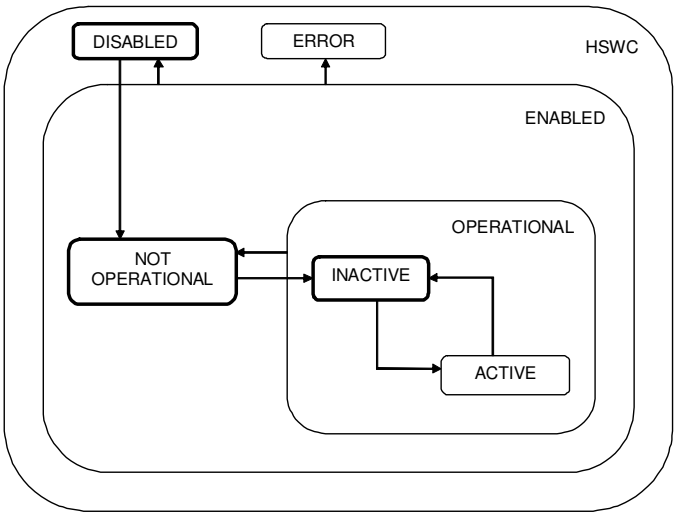


Bild 3: Statechart der Funktion HSWC

Das Beispiel eines hierarchischen Statecharts der Funktion HSWC in Bild 3 beinhaltet fünf atomare und drei hierarchische Zustände. Das System ist initial im Zustand DISABLED und kann von dort in den Zustand NOT OPERATIONAL wechseln. Von jedem Unterzustand von ENABLED kann die Funktion HSWC in die Zustände DISABLED oder ERROR wechseln. Vom Zustand NOT OPERATIONAL kann das System in die Zustände INACTIVE und ERROR wechseln. Innerhalb des hierarchischen Zustands OPERATIONAL kann die Funktion HSWC zwischen den Zuständen INACTIVE und ACTIVE wechseln und von jedem Unterzustand aus in die Zustände NOT OPERATIONAL oder ERROR übergehen. Die initialen Zustände sind durch dicke Ränder gekennzeichnet. Der Zustand ERROR steht für die interne Fehlerbehandlung zur Verfügung und kann von jedem Unterzustand von Zustand ENABLED erreicht werden. Der Zustand HSWC dient als zusammenfassender Rahmen, in dem allgemeine Anforderungen eingegliedert werden können, die in jedem Zustand von HSWC Geltung haben.

**4. Modellbasierte Anforderungsanalyse**

Im folgenden Abschnitt werden die für die Erstellung einer modellbasierten Anforderungsspezifikation benötigten Schritte und Modellierungselemente vorgestellt und anhand des Beispiels der Funktion HSWC erläutert und demonstriert.

In der modellbasierten Anforderungsanalyse dienen Statecharts zur Beschreibung des geforderten Systemverhaltens. Zu Beginn einer Systementwicklung kann ein solches Statechart durch Interviews oder Workshops erstellt werden und dient im Verlauf der modellbasierten Anforderungsanalyse zur Gliederung der Anforderungsspezifikation, um die funktionalen Anforderungen zu strukturieren und den entsprechenden Systemzuständen und Zustandsübergängen gezielt zuordnen zu können.

**Kapitel X Funktionale Anforderungen HSWC**
**Kapitel X.1 Systemzustände**
Kapitel X.1.1 Systemzustand HSWC
Kapitel X.1.2 Systemzustand DISABLED
Kapitel X.1.3 Systemzustand ENABLED
Kapitel X.1.4 Systemzustand NOT OPERATIONAL
Kapitel X.1.5 Systemzustand OPERATIONAL
Kapitel X.1.6 Systemzustand INACTIVE
Kapitel X.1.7 Systemzustand ACTIVE
Kapitel X.1.8 Systemzustand ERROR
**Kapitel X.2 Zustandsübergänge**
Kapitel X.2.1 Zustandsübergang DISABLED-NOT OPERATIONAL
Kapitel X.2.2 Zustandsübergang ENABLED-DISABLED
Kapitel X.2.3 Zustandsübergang NOT OPERATIONAL-INACTIVE
Kapitel X.2.4 Zustandsübergang OPERATIONAL-NOT OPERATIONAL
Kapitel X.2.5 Zustandsübergang INACTIVE-ACTIVE
Kapitel X.2.6 Zustandsübergang ACTIVE-INACTIVE
Kapitel X.2.7 Zustandsübergang ENABLED-ERROR

Bild 4 Gliederung der funktionalen Anforderungsspezifikation HSWC

*Beispiel: In Bild 4 ist die Gliederung der Anforderungsspezifikation der Funktion HSWC dargestellt, wie sie sich direkt aus dem Zustandsautomaten des abstrakten Verhaltens von HSWC aus Bild 3 ableitet.*

Im folgenden Verfahrensschritt werden die vorhandenen Anforderungen vereinzelt und typisiert, so dass jedes Objekt in der Anforderungsspezifikation nur einen Sachverhalt beschreibt und zu einem der folgenden Typen zugeordnet werden kann:

heading - ein Gliederungselement
information - ein beliebiger erläuternder oder kommentierender Inhalt
requirement - die Spezifikation genau einer gewünschten Produkteigenschaft

Durch Vereinzelung und Typisierung wird die Änderbarkeit von Dokumenten derart verbessert, dass ein nachträgliches Verschieben oder Löschen von Teilen der Spezifikation im Vergleich zu unsystematischeren Vorgehensweisen deutlich vereinfacht wird.

Tabelle 1: Beispiel für Anforderung und Information in der Anforderungsspezifikation HSWC

| Typ | Text |
|---|---|
| information | Auf der Rückbank sitzende Passagiere können durch sich automatisch schließende Fenster verletzt werden. |
| requirement | Bei Deaktivierung der Fensterheber der Rücksitze durch den Fahrer sollen die Fenster nicht automatisch geschlossen werden. |
| requirement | Bei detektierten Fondpassagieren sollen die Fenster nicht automatisch geschlossen werden. |

*Beispiel: In Tabelle 1 wird die Typisierung von Spezifikationsobjekten als Information und Requirement demonstriert.*

In Spezifikationen von technischen Steuerungsfunktionen werden häufig Bedingungen formuliert, die in ähnlicher Weise als Trigger und Zustandsinvarianten auch in Statecharts

Verwendung finden. Aus diesem Grund werden von *requirement* abgeleitete Typen für die Spezifikation von aussagenlogischen Bedingungen eingeführt:

| | |
|---|---|
| condition | - die Spezifikation einer Bedingung |
| XOR | - eine exklusiv verodernde Komposition von condition-objekten |
| OR | - eine verodernde Komposition von condition objekten |
| AND | - eine konjunktive Komposition von condition-objekten |

Die logischen Operatoren XOR, OR, und AND können zur Spezifikation komplexer, hierarchischer Bedingungen, beispielsweise Trigger oder Invarianten, Aktivierungs- und Hemmungsbedingungen, in der Anforderungsspezifikation genutzt werden.

Tabelle 2: Anforderungsspezifikation der Aktivierungsbedingung HSWC

| Typ | Text |
|---|---|
| AND | Die Funktion HSWC wird aktiviert, wenn alle folgenden Bedingungen gelten: |
| condition | • Die Funktion HSWC ist betriebsbereit (Zustand OPERATIONAL). |
| condition | • Die Geschwindigkeit des Systemfahrzeugs ist größer $V_{aktiv}$. |
| condition | • Auf der Rückbank werden keine Passagiere detektiert. |
| condition | • Die hinteren Fensterheber-Bedienelemente sind nicht gesperrt. |

*Beispiel: In Tabelle 2 ist beispielhaft die Spezifikation der Aktivierung der Funktion HSWC mittels einer komplexen Bedingung dargestellt.*

Ein Zustand wird durch seine optionalen Unterzustände, den optionalen initialen Zustand, eine Zustandsinvariante und optionale Anforderungen spezifiziert. In Tabelle 3 wird eine beispielhafte Spezifikation des Zustands OPERATIONAL dargestellt. Die Bedingung für den Aufenthalt im Zustand OPERATIONAL (Zustandsinvariante) wird mittels der vorgeschlagenen Bedingungstypen spezifiziert: Die Bedingung erweitert die Bedingung für den Zustand ENABLED um Bedingungen bezüglich der Fahrzeuggeschwindigkeit und der Belegung der Rückbank. In Tabelle 4 wird eine beispielhafte Spezifikation des Zustands INACTIVE dargestellt. Die Bedingung für die Betriebsbereitschaft wird um Bedingungen zur möglichen Unterdrückung einer Aktivierung der Funktion erweitert. Eine unterdrückte, betriebsbereite Funktion HSWC ist inaktiv. Das Weiterhin demonstriert das Beispiel die Schachtelung von komplexen Bedingungen; hier die Einbettung der Veroderung der Unterdrückungsbedingungen in die Verundung der Bedingung der Inaktivität.

Ein Zustandsübergang wird durch Anfangs- und Endzustand, Trigger-Bedingung und optionale Anforderungen spezifiziert.

*Beispiel: In Tabelle 5 wird die Spezifikation des Übergangs vom Zustand INACTIVE zum Zustand ACTIVE dargestellt.*

## 5. Variantenmanagement für die modellbasierte Anforderungsanalyse

Das Variantenmanagement für modellbasierte Anforderungsspezifikationen besteht aus einem Selektionsverfahren, um Anforderungen einer Variante zuzuordnen, und einem Parameterersetzungsverfahren, um generische Anforderungen in variantenspezifische Anforderungen zu transformieren. Dank der im vorangegangenen Abschnitt vorgestellten modellbasierten Strukturierung der Spezifikation können durch das Selektionsverfahren Zustände und Transitionen einer Variante zugeordnet oder entzogen werden. Die Analyse der Auswirkungen einer solchen Operation, beispielsweise welche Zustandsübergänge sind nicht mehr gültig, wenn ein Zustand aus einer Variante entfernt wird, wird durch das Modell des Zustandsautomaten erheblich erleichtert. Die Selektion kann durch Erweiterung der Spezifikationsobjekte um Selektionsattribute erfolgen. Typische Erweiterungen sehen ein binäres Attribut pro Variante oder ein mehrwertiges Attribut als Enumeration der zugeordneten Variantennamen vor. Im folgenden Beispiel wird die Zuordnung durch binäre Attribute vorgenommen.

Tabelle 3: Generische Anforderungsspezifikation für Zustand OPERATIONAL

| Typ | ALPHA | BETA | Kurzname | Text |
|---|---|---|---|---|
| heading | X | X | | X.1.5 Systemzustand OPERATIONAL |
| information | X | X | Zustand OPERATIONAL | Der Zustand OPERATIONAL repräsentiert die betriebsbereite Funktion HSWC. |
| requirement | X | X | Subzustände OPERATIONAL | Der Zustand OPERATIONAL beinhaltet den Zustand INACTIVE und den Zustand ACTIVE. |
| requirement | X | X | Initialer Zustand OPERATIONAL | Der initiale Zustand in Zustand OPERATIONAL ist der Zustand INACTIVE. |
| AND | X | X | Bedingung Betriebsbereitschaft HSWC | Die Funktion HSWC ist betriebsbereit, wenn alle folgenden Bedingungen gelten: |
| condition | X | X | Bedingung HSWC ist enabled | • Die Funktion HSWC befindet sich im Zustand ENABLED. |
| condition | X | X | Bedingung Operation Speed | • Die Fahrzeuggeschwindigkeit ist größer oder gleich $V_{Operation}$. |
| condition | | X | Bedingung Keine Fond-Passagiere | • Auf der Rückbank werden keine Passagiere detektiert. |

*Beispiel: In Tabelle 3 ist die Bedingung 'Keine Fond-Passagiere' nur für die Variante BETA selektiert, da ALPHA nur eine Warnfunktion ist, und keine Schutzfunktion für Fond-Passagiere vor der Verletzungsgefahr durch automatisch schließende Fenster notwendig ist. Ebenso ist in Tabelle 4 die Bedingung 'Keine Fond-Passagiere' nur für die Variante BETA selektiert. In Tabelle 5 ist die Aktivierungsnachricht im Kombi nur für Variante ALPHA selektiert, da Variante BETA, keine Warnung ausgibt.*

Tabelle 4: Generische Anforderungsspezifikation für Zustand INACTIVE

| Typ | ALPHA | BETA | Kurzname | Text |
|---|---|---|---|---|
| heading | X | X | | X.1.6 Systemzustand INACTIVE |
| information | X | X | Zustand INACTIVE | Der Zustand INACTIVE repräsentiert die betriebsbereite Funktion HSWC, die unmittelbar auf eine Aktivierungssituation reagieren kann. |
| AND | X | X | Bedingung Inaktivität HSWC | Die Funktion HSWC ist inaktiv, wenn alle folgenden Bedingungen gelten: |
| condition | X | X | Bedingung HSWC ist betriebsbereit | • Die Funktion HSWC ist betriebsbereit (Zustand OPERATIONAL). |
| OR | X | X | Bedingung HSWC Unterdrückung | Die Aktivierung der Funktion ist nicht möglich, wenn mindestens eine der folgenden Bedingungen gilt: |
| condition | X | X | Bedingung Fenster geschlossen | ○ Alle Fenster sind geschlossen. |
| condition | | X | Bedingung Fond-Festerheber blockiert | ○ Durch den Fahrer wurden die hinteren Bedienelement für die Fenster blockiert. |
| condition | X | X | Bedingung Activation Speed | ○ Die Fahrzeuggeschwindigkeit ist kleiner $V_{Activation}$. |
| requirement | | X | Anzeige der Funktionsunterdrückung | Ein Unterdrückung der Funktion HSWC durch Fondpassagiere oder Blockierung der hinteren Fenster soll dem Fahrer im Kombiinstrument angezeigt werden. |

Tabelle 5: Anforderungsspezifikation für den Übergang INACTIVE-ACTIVE

| Typ | ALPHA | BETA | Kurzname | Text |
|---|---|---|---|---|
| heading | X | X | | X.1.4 Zustandsübergang von INACTIVE zu ACTIVE |
| information | X | X | Zustandsübergang INACTIVE-ACTIVE | Der Übergang von Zustand INACTIVE zu Zustand ACTIVE repräsentiert die Aktivierung der Funktion HSWC.. |
| requirement | X | X | Startzustand | Der Startzustand ist Zustand INACTIVE. |
| requirement | X | X | Endzustand | Der Endzustand ist Zustand ACTIVE. |
| AND | X | X | Bedingung Aktivierung HSWC | Die Funktion HSWC wird aktiviert wenn alle folgenden Bedingungen gelten: |
| condition | X | X | Bedingung HSWC ist betriebsbereit | • Die Funktion HSWC ist betriebsbereit (Zustand OPERATIONAL). |
| condition | X | X | Bedingung Geöffnete Fenster | • Mindestens ein Fenster ist geöffnet. |
| condition | X | X | Bedingung VFzg>=VActivation | • Die Fahrzeuggeschwindigkeit ist größer oder gleich $v_{Activation}$. |
| requirement | X | | Aktivierungsnachricht im Kombi | Bei Aktivierung der Funktion HSWC soll eine Meldung an den Fahrer im Kombiinstrument ausgegeben werden. |
| requirement | X | X | Aktiv-Bit gesetzt | Bei Aktivierung der Funktion HSWC wird das Aktiv-Bit gesetzt. |

Die Nutzung von Parametern zur Erstellung von generischen Anforderungstexten erlaubt es, gleichlautende Sachverhalte über mehrere Varianten hinweg zu formulieren, und erst zu einem späteren Zeitpunkt eine variantenspezifische Ausprägung der Anforderung automatisch zu generieren.

*Beispiel: In Tabelle 3 ist die Bedingung für die Überschreitung der Bereitschafts-Geschwindigkeit generisch mit dem Parameter $v_{Operation}$, anstelle von zwei Anforderungen mit konkreten Werten für ALPHA und BETA spezifiziert. Gleichermaßen enthalten Tabelle 4 und Tabelle 5 statt zweier Anforderungen mit konkreten Geschwindigkeitsschwellwerten generische Anforderungen mit dem Parameter $v_{Activation}$.*

Tabelle 6: Beispiel einer Parameterdefinition

| **Parametername** | **Parameterwert** | | | **Paremetertyp** |
|---|---|---|---|---|
| | **default** | **ALPHA** | **BETA** | |
| $v_{Activation}$ | 150 | 120 | | km/h |
| $V_{Operation}$ | 100 | 90 | | km/h |

Wie in Tabelle 6 dargestellt, kann die Verwaltung der variantenspezifischen Parameter in einer separat geführten Liste erfolgen. Die Ersetzung der Parameter im Text und die Generierung varianten-spezifischer Anforderungen wird vollständig automatisiert. Die Verwendung von Parametern ist nicht auf numerische Daten beschränkt; von beliebigen Zeichenketten, über Bilddaten bis hin zu ganzen OLE-Objekten kann ein breites Spektrum von parametrierbaren Daten variiert werden.

*Beispiel: Ein wesentlicher Unterschied zwischen den Varianten liegt in der Fehlerbehandlung, die nur in Variante ALPHA intern erfolgt. Der Zustand ERROR und dessen eingehender Übergang, bzw. Kapitel X.1.8 und X.2.7, müssen daher aus Variante BETA deselektiert werden. Eine mögliche grafische Darstellung des Statecharts in der generischen Anforderungsspezifikation wird durch einen Parameter ersetzt, der auf ALPHA und BETA angepasste Grafiken enthält.*

## 6. Umsetzung und praktische Erfahrungen

In Teilen konnte das vorgestellte Variantenmanagement bereits vor der Kombination mit der modellbasierten Anforderungsspezifikation in diversen industriellen Projekten in Luftfahrt, Bahnindustrie und Automobilbranche [3] erfolgreich eingesetzt werden. Die Kombination mit der modellbasierten Anforderungsanalyse hat die Leistungsfähgkeit dieses Ansatz nochmals signifikant verbessert; insbesondere, da durch die modellbasierte Strukturierung der Anforderungsspezifikation die Änderungen an einzelnen Varianten mit deutlich geringerem Aufwand durchgeführt werden können. Theoretisch erlaubt das Variantenmanagment in

einem Projekt mit *n* Varianten den Bearbeitungs- und Änderungsaufwand im Idealfall um den Faktor *n* zu reduzieren. In der Praxis kann eine solche Ersparnis allerdings kaum erzielt werden, da u.a. Anforderungen nur genau einer Variante zugeordnet werden können und zusätzlicher Aufwand für die komplexere Erstellung einer generischen Anforderung entsteht.

Besonderer Wert bei der Entwicklung des Verfahrens zur modellbasierten Anforderungsanalyse wurde auf die uneingeschränkte Möglichkeit zur Verwendung natürlichsprachlicher Spezifikationsobjekte gelegt: Verfahren und Tool sollen vorrangig in den frühen Phasen der Systementwicklung ein Hilfsmittel darstellen, um die Analyse und Erstellung natürlichsprachlicher Spezifikationen zu erleichtern; im Unterschied beispielsweise zu dem in [4] präsentierten Ansatz, Anforderungen in ein Designmodell zu transformieren.

Zur Unterstützung der modellbasierten Anforderungsanalyseschritte und zur Automatisierung des Variantenmanagements wurde bei Berner&Mattner eine Erweiterung für Telelogic DOORS (*B&M Toolbox für DOORS*) entwickelt, die für die modellbasierte Spezifikation mit Variantenmanagement neuer und bestehender Projekte verwendet werden kann. Die Bedienung ist in vollem Umfang durch Bedienoberflächen gesteuert und erlaubt dem Benutzer auf einfache Weise neue Varianten zu erzeugen oder bestehende zu aktualisieren. Die *B&M Toolbox für DOORS* unterstützt darüberhinaus auch die automatische Indexierung und Verwaltung von Stichwortverzeichnissen, die automatisierte Verwaltung von Signallisten und Testfällen, die automatische Generierung von Anforderungsverlinkungen, Unterstützung bei der Spezifikation, Test-Vollständigkeitsanalysen [5] und Generierung von (ausführbaren) Testfällen sowie die Generierung diverser Exportformate, z.B. XML und RiF. Ein weiteres Feature ist die automatische Transformation von Anforderungsspezifikationen in Telelogic DOORS zu Mindmaps, was sich insbesondere während Workshops und Interviews zur Anforderungserhebung bewährt hat.

Die modellbasierte Anforderungsanalyse für variantenreiche Systementwicklungen und die B&M Tool-box für DOORS wurden bereits in mehreren Projekten zur Entwicklung von Fahrassistenzsystemen bei der Daimler AG erfolgreich eingesetzt. Im Vergleich zu vorherigen Entwicklungsprojekten konnte der Umfang der Anforderungsspezifikation um bis zu 30% reduziert werden, was auf die Bereinigung von nicht relevanten Informationen und Redundanzen zurückzuführen ist. Die verbesserte Verständlichkeit und Änderbarkeit der Dokumentenstruktur führten zu einer Ersparnis von bis zu 25 % des Erstellungs- und Bearbeitungsaufwandes der Anforderungsspezifikationen.

**7. Zusammenfassung**

Das in diesem Beitrag vorgestellte Konzept zur modellbasierten Anforderungsanalyse hat sich in mehreren industriellen Projekten als geeignetes Mittel zur Reduzierung von Bearbeitungsaufwand und -zeit zur Erstellung, Aktualisierung und Analyse von (funktionalen) Anforderungsspezifikationen erwiesen. Weiterhin hat sich die Kombination von modellbasierten Ansätzen mit dem Variantenmanagement als äußerst leistungsfähig erwiesen, den Aufwand und die Zeit zur Erstellung von Spezifikationen für ganze Produktfamilien zu minimieren.

Erste Ergebnisse aus der Anwendung eines adaptierten Konzepts zur Analyse und Spezifikation von Testfällen zeigten bereits, dass das vorgestellte Variantenmanagement auch auf andere Probleme mit der Variantenvielfalt industrieller Funktionsentwicklungen anwendbar ist.

Ausblickend ist eine Verbesserung der Identifikation und Absicherung eines geeigneten Zustandsmodells durch die Anwendung von modellbasierten Analysemethoden geplant, z.B. mittels Sequence Enumeration [6] oder Model Checking [7]. Weiterhin können die modellbasierten Anforderungsspezifikationen genutzt werden, um automatisierte Verfahren zur Entwicklung von Testfällen [8] und zur Auswertung der Testergebnisse mit Bezug auf die Anforderungen [5] anzuwenden. Ebenfalls vielversprechende Ansätze betreffen die Einbindung modellbasierter Entwicklungstools zur Editierung der Statecharts sowie die Entwicklung von entsprechenden Toolumgebungen für weitere kommerzielle RE-Systeme.

[1] M. Grochtmann, L. Schmuhl: *Systemverhaltensmodelle zur Spezifikation bei der modellbasierten Entwicklung von eingebetteter Software im Automobil*, 2005, Dagstuhl Seminar *Modellbasierte Entwicklung Eingebetteter Systeme*

[2] Object Management Group (OMG): *Unified Modeling Language (UML)*, Version 2.2, 2009, verfügbar auf www.omg.org

[3] M. Grochtmann: *Zahlen und Züge - Bessere Anforderungen in der Bahntechnik*, elektronik industrie 11/2008

[4] E. Geisberger, J. Grünbauer, B. Schätz: *A Model-Based Approach To Requirements Analysis*, 2007, Dagstuhl Seminar Methods for Modelling Software Systems

[5] P. Liggesmeyer: Software-Qualität, 2002, Spektrum Akademischer Verlag

[6] S. Prowell, J. H. Poore: *Foundations of Sequence-Based Software Specification*, 2003 IEEE Transaction of Software Engineering, Vol. 29, No. 5

[7] E. M. Clarke, O. Grumberg, D. A. Peled: *Model Checking*, 2000 MIT Press

[8] C. Robinson-Mallett: *Modellbasierte Modulprüfung für die Entwicklung technischer, software-intensiver Systeme*, 2005, verfügbar auf www.systematic-testing.org