Ein Ansatz zum merkmalsbasierten Konsistenzmanagement in der Produktlinienentwicklung

Norbert Wiechowski, Thomas Gerlitz, Daniel Merschen, Stefan Kowalewski

Embedded Software Laboratory
RWTH Aachen University
Aachen, Germany
{wiechowski | gerlitz | merschen | kowalewski}@embedded.rwth-aachen.de

Abstract: In dieser Arbeit werden Konzepte präsentiert, mit denen Verknüpfungen zwischen Elementen, welche während der Produktlinienentwicklung entstehen, analysiert und im Fall von identifizierten Inkonsistenzen die semiautomatische Ausführung von korrektiven Maßnahmen ermöglicht wird. Dies geschieht auf Basis entwicklungsrelevanter Meta-Informationen, speziell den Merkmalen, welche mittels Annotationen an den Artefaktelementen hinterlegt werden. Betrachtet werden Inkonsistenzen zwischen verschiedenen Artefakten und die einem Artefakt inhärenten, wobei den artefaktübergreifenden Inkonsistenzen Prioritäten zugewiesen werden. Die prototypische Realisierung orientiert sich an einer für die Industrie typischen Werkzeuglandschaft.

1 Einleitung

In der Automobilbranche sind die Entwicklung von softwarebasierten Innovationen und die hohe Nachfrage nach Individualisierungsmöglichkeiten von Kundenseite Gründe für eine steigende Komplexität, gemessen an der Anzahl der zur Realisierung eines Produkts notwendigen Elemente. Der stetig steigende Einsatz elektronischer Komponenten zieht gleichermaßen einen erhöhten Entwicklungsaufwand nach sich. Legt man den aktuellen Trend zugrunde, so werden laut einer Prognose von [Hel05] die Kosten für die Entwicklung und Wartung elektronischer Komponenten und deren Software im Jahr 2015 einen Anteil von 35% an den Gesamtentwicklungskosten eines Automobils haben. Um der steigenden Komplexität entgegenzutreten und ein erfolgreiches Variantenmanagement umzusetzen, wird zunehmend auf die Produktlinienentwicklung in Kombination mit modellgetriebener Entwicklung gesetzt. Dies erlaubt die simultane Entwicklung und Ableitung individuell konfigurierter Produkte mit gemeinsamer Merkmalsbasis anhand einer der Produktlinie inhärenten Merkmalsmenge. Der zeitlich strukturierte Entwicklungsprozess einer Produktlinie im industriellen Alltag ist optimalerweise in getrennten Phasen organisiert, orientiert an einem Vorgehensmodell. Das Entwicklungsergebnis einer solchen Phase wird als Artefakt bezeichnet. Das für diesen Zweck in der Automobilindustrie häufig verwendete V-Modell ist in Abbildung 1 exemplarisch dargestellt. Für die in den folgenden Abschnitten betrachteten Ansätze zum Konsistenzmanagement werden in dieser Arbeit 150%-Artefakte betrachtet, die aufseiten des Herstellers entstehen und sämtliche Elemente, zur Konfiguration und

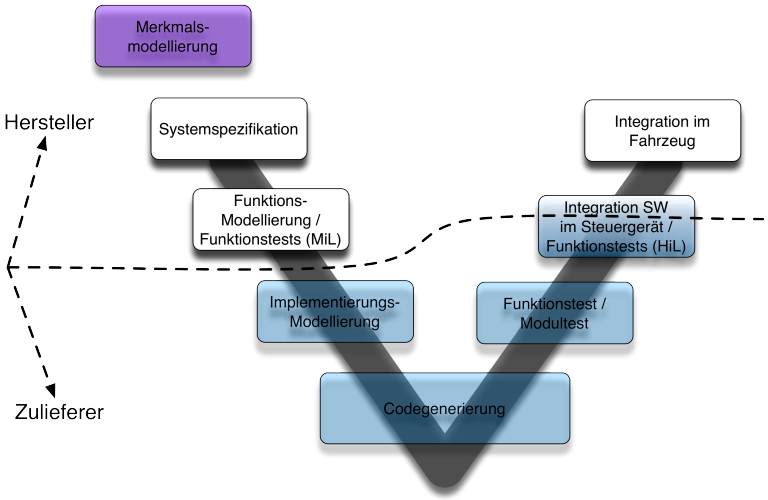

Abbildung 1: V-Modell für modellbasierte Entwicklung [MSR+10]

Ableitung von Produkten aus der Produktlinie, beinhalten. Das Artefakt der Systemspezifikation ist das Lastenheft, welches aufbauend auf dem Merkmalsmodell, als Fundament für die nachfolgenden Entwicklungsphasen dient, gefolgt von der Funktionsmodellierung. Das durch den Zulieferer in die Hardware integrierte Funkionsmodell wird anschließend auf Basis eines Test-Artefakts auf Korrektheit geprüft und bei Erfolg ins Fahrzeug integriert. Die so entstandenen Artefakte stehen durch bidirektionale Abhängigkeiten in Verbindung miteinander. Logisch zusammengehörige Artefaktelemente sollten miteinander verknüpft sein. Solche Elemente beschreiben die gleiche Funktionalität in Bezug auf das ableitbare Produkt, betrachtet aus unterschiedlichen Blickwinkeln. Das erschwert die Erzeugung, Entfernung und Wartung sowohl von Merkmalen als auch von Artefaktelementen, welche während der Evolution der Produktlinie zueinander konsistent gehalten werden sollten, um eine korrekte Funktionalität abgeleiteter Produkte sicherzustellen. Im Folgenden werden dazu zwei auf Merkmalen basierende Ansätze präsentiert. In die Thematik einführend werden in Abbildung 2 exemplarisch zusammengehörige und verknüpfte Artefaktelemente aus den Entwicklungsphasen aufgezeigt, welche jeweils mit Merkmalen verknüpft sind. In diesem Beispiel sind die Testfälle im Gegensatz zu den Funktionsmodellen und Anforderungen nicht mit Merkmal A, dafür mit Merkmal C verknüpft. Somit ist diese Verknüpfung der Artefaktelemente inkonsistent in Bezug auf deren Merkmale.

1.1 Motivation

Durch die steigende Anzahl von zu verwaltenden Artefaktelementen im Zuge der Produktlinienentwicklung ist eine manuelle Kontrolle der Konsistenz zwischen Entwicklungsar-

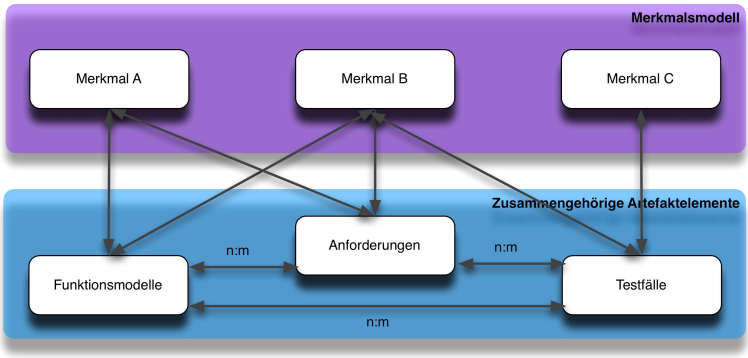

Abbildung 2: Artefaktübergreifende Inkonsistenz

tefakten immer aufwendiger und nicht wirtschaftlich. Gleichzeitig ist die Konsistenz der Artefakte innerhalb einer Produktlinie ein ausschlaggebendes Qualitätskriterium. Treten Inkonsistenzen zwischen zwei oder mehreren Artefaktelementen auf, kann dies weitreichende Folgen auf die funktionale Korrektheit und Qualität des gesamten Produktes haben. Durch Inkonsistenzen erzeugte Fehler im Entwicklungsprozess der Produktlinie können auf weitere Artefaktelemente propagiert werden [LP08]. Die Korrektur und Analyse der durch Inkonsistenzen induzierten Fehler ist für den Hersteller ein zeitlich intensives und demzufolge ein kostenträchtiges Unterfangen. Zur Sicherstellung, dass die Produktqualität erhalten bleibt und die Effizienz des Analyseprozesses verbessert wird, sind zusätzliche Techniken nötig, die eine automatisierte Analyse der Entwicklungsergebnisse in einem prozedural aufeinander abgestimmten und werkzeugtechnisch unterstützten Entwicklungsprozess erlauben. Ein erfolgreiches Konsistenzmanagement trägt dazu bei, die Effizienz des Entwicklungsprozesses zu erhöhen und unterstützt durch eine verminderte Fehlerpropagation die Wiederverwendung von Entwicklungsergebnissen. Die Herausforderung die sich dabei stellt, ist die Integration eines Konsistenzmanagements in einen in der Industrie über lange Zeit gewachsenen, bestehenden Arbeitsfluss und deren Werkzeuglandschaft ohne grundlegende Änderungen vorzunehmen. Dies beinhaltet im Speziellen auch die Berücksichtigung natürlichsprachlicher Artefakte in einem werkzeuggestützt ausführbaren Analyseprozess. Im Kontext dieser Arbeit werden daher zwei automatisiert durchführbare Analysen präsentiert, die einen Großteil der Artefaktelemente aus den Entwicklungsphasen der Automobilhersteller berücksichtigen und auf Basis der vorhandenen Merkmalsverknüpfungen Rückschlüsse auf Entwicklungsfehler einer Produktlinie erlauben. Basierend auf diesen Ergebnissen können korrektive Maßnahmen semiautomatisch durchgeführt werden.

Den Ausgangspunkt für das artefaktübergreifende Konsistenzmanagement bildet die Arbeit von Cmyrev et al. [CNHR13]. Die dort vorgestellten Inkonsistenzen zwischen Lastenheft und verknüpften Testfällen wurden in dieser Arbeit aufgegriffen und allgemeingültig für beliebige auf Merkmalen basierende Artefakte definiert. Mittels der Definitionen wird es ermöglicht, werkzeuggestützt Analysen auf solchen Artefakten auszuführen. Konkret wurde dies genutzt, um ein in Matlab/Simulink implementiertes Funktionsmodell in Ver-

bindung mit natürlichsprachlichen Anforderungen und Testfällen zu analysieren. Dabei fand eine Integration des Konsistenzmanagements in ein datenbankorientiertes Analyserahmenwerk statt [MPBK11, MPK12, MDR+12]. In diesem Rahmenwerk können Signalverläufe in Matlab/Simulink verfolgt und durch spezifische Sichten das Verständnis komplexer Simulink-Modelle erleichtert werden. In Kombination mit einem datenbankorientiertem Ansatz und Annotationen zur Verknüpfung der Artefaktelemente mit entwicklungsrelevanten Meta-Informationen wird es ermöglicht, Artefakte werkzeugübergreifend zu analysieren und identifizierte Inkonsistenzen semiautomatisch zu korrigieren. Den artefaktübergreifenden Inkonsistenzen werden dabei Prioritäten zugewiesen, die eine Einstufung sowohl der Korrekturmöglichkeiten als auch die Relevanz der Korrektur erlauben. Simultan werden die Meta-Information der Merkmalsverknüpfungen dazu genutzt, um neben der artefaktübergreifenden Analyse eine Konsistenzanalyse eines einzelnen Artefakttyps durchzuführen und artefaktinhärente Inkonsistenzen zu identifizieren. Das gesamte Konsistenzmanagement orientiert sich an industriell verwendeten Werkzeugen und wird in einen Arbeitsprozess eingebettet, der werkzeugunterstützt ausgeführt werden kann.

Der Rest der Arbeit ist folgendermaßen aufgebaut. In Abschnitt 2 werden zunächst verwandte Arbeiten und Gemeinsamkeiten bzw. Abgrenzungen von der vorliegenden Arbeit beschrieben. Anhand eines illustrierten Arbeitsprozesses wird in Abschnitt 3 das Konsistenzmanagement präsentiert. Dieser besteht aus sieben Schritten zur Analyse von artefaktübergreifenden und artefaktinhärenten Inkonsistenzen sowie deren Korrekturmöglichkeiten. Abschnitt 4 fasst zunächst den Inhalt dieser Arbeit zusammen und gibt im Anschluss an die Diskussion einen Ausblick für aufbauende Entwicklungen.

2 Verwandte Arbeiten

In [VLDL98] werden Inkonsistenzen thematisiert und klassifiziert, die während der Phase der Systemspezifikation auftreten können. Mittels Formalisierungen und Heuristiken können diese über ein Werkzeug erkannt werden. Über die Einführung neuer oder einer Transformation vorhandener Ziele besteht ebenfalls die Möglichkeit zur Korrektur erkannter Inkonsistenzen. Eine andere Art von Inkonsistenzkategorisierung im Kontext des Konsistenzmanagements von Artefakten der Produktlinienentwicklung wird in [CNHR13] beschrieben. Erwähnte Konsistenzsicherungsmaßnahmen beinhalten semantisch korrekte Definitionen des Merkmalsmodells, passende Zuordnungen von Merkmalen zu Anforderungen und Testfällen, sowie die Abdeckung jeder Anforderung durch mindestens einen Testfall. Ist das Kriterium der Anforderungsabdeckung nicht erfüllt, wird die Diskrepanz zwischen Anforderungen und Testfällen als widersprüchlich, unvollständig oder redundant klassifiziert. Auf dieser Klassifizierung aufbauend werden Korrekturmöglichkeiten beschrieben. Da sich diese Einteilung erkannter Inkonsistenzen gut dazu eignet, auch auf andere Artefakte des Entwicklungsprozesses übertragen zu werden, wurde diese Klassifizierung in der vorliegenden Arbeit aufgegriffen, um werkzeugunterstützt Konsistenzanalysen und korrektive Maßnahmen über die 150%-Artefakte aus den Entwicklungsphasen eines Herstellers auszuführen. [GS07] präsentiert ein prototypisches Framework, das spezifische Sichten für die Phasen der Anforderungsmodellierung, Analyse und Design bereitstellt. Auf

Grundlage der Merkmale der Software-Produktlinie ist eine übergeordnete Sicht erstellt worden, die anhand spezifizierter Regeln eine Konsistenzprüfung gegen ein Meta-Modell der Produktlinie zwischen den Sichten ermöglicht. Die Arbeit ist aus dem Bereich der Software-Entwicklung motiviert und nutzt aus, dass die Artefakte selbst einem formalisierten Meta-Model (UML) unterliegen. Da die Festhaltung der Artefakte speziell im industriellen Umfeld oftmals zumindest teilweise auf natürlicher Sprache basiert, beispielsweise bei der Systemspezifikation, kann die Existenz eines zur werkzeuggestützten Analyse brauchbaren Meta-Modells für alle Artefakte der Entwicklungsphasen nicht vorausgesetzt werden. In [IO05, Far11] werden Ansätze präsentiert, um natürlichsprachliche Artefakte zu formalisieren. [Far11] betrachtet natürlichsprachliche und aus der praktischen Erfahrung stammende Entwicklungsrichtlinien. Die durch Transformation erhaltenen logischen Artefakte dienen ebenfalls artefaktübergreifenden Analysen. Die dafür notwendige Formalisierung erfordert aus Sicht des Anwenders jedoch einen hohen manuellen Aufwand bzw. ein Training, um den Formalisierungsprozess fehlerfrei umzusetzen. Aktuelle Techniken zur Formalisierung natürlichsprachlicher Artefakte würden daher oftmals einen zu großen Eingriff in den in der Wirtschaft vorhandenen Entwicklungsprozess erfordern, weswegen sie in dieser Arbeit nicht verwendet werden. Die für die Konsistenzanalysen relevanten Informationen werden an den Artefakten über Annotationen mit definierter Syntax und Semantik hinterlegt, vgl. Abschnitt 3.1, die eine werkzeuggestützte Verarbeitung ermöglichen und einfach anzuwenden sind. Die Technik der Annotation wurde in [MDR+12] vorgestellt. Sie hat neben der algorithmischen Verarbeitungsmöglichkeit den Vorteil, dass entwicklungsrelevante Meta-Informationen unabhängig vom Artefakttyp auf die gleiche Art und Weise spezifiziert und analysiert werden können, sehr wenig Spielraum für Fehler erlaubt und implizit bereits potenzielle Zusammenhänge zwischen den Artefakten erkennen lässt.

3 Merkmalsbasiertes Konsistenzmanagement

Um die Konsistenzanalysen auf Grundlage der in diesem Abschnitt präsentierten formalen Definitionen für merkmalsbasierte Inkonsistenzen praktisch zu realisieren, müssen zunächst einige Voraussetzungen erfüllt werden. Diese bestehen daraus, die zu untersuchenden Artefakte an einer zentralen Stelle zusammenzuführen und anschließend um Informationen anzureichern, die zu analytischen und korrektiven Zwecken benötigt werden. Für diesen Vorgang wird der in Abbildung 3 illustrierte Arbeitsprozess empfohlen, dessen Schritte logisch aufeinander aufbauen und dem Anwender den Prozess der Informationsanreicherung erleichtern. Dies wird mittels Assistenten erreicht, welche höherwertige Informationen auf Basis der Ergebnisse zuvor durchgeführter Schritte erzeugen können, die wiederum als Grundlage für die folgende Schritte genutzt werden. Beispielsweise können potenzielle Zusammenhänge zwischen Elementen unterschiedlichen Artefakttyps aufgrund äquivalenter Merkmalsverknüpfungen identifiziert werden.

Über den Import der Artefakte in eine Datenbank werden diese zusammengeführt. Das erlaubt die Analyse von Artefakten unabhängig von den Werkzeugen zur deren Erstellung. Mittels ebenfalls in der Datenbank hinterlegten Annotationstypen [Pot12], können Artefaktelemente manuell mit entwicklungsrelevanten Meta-Informationen angereichert

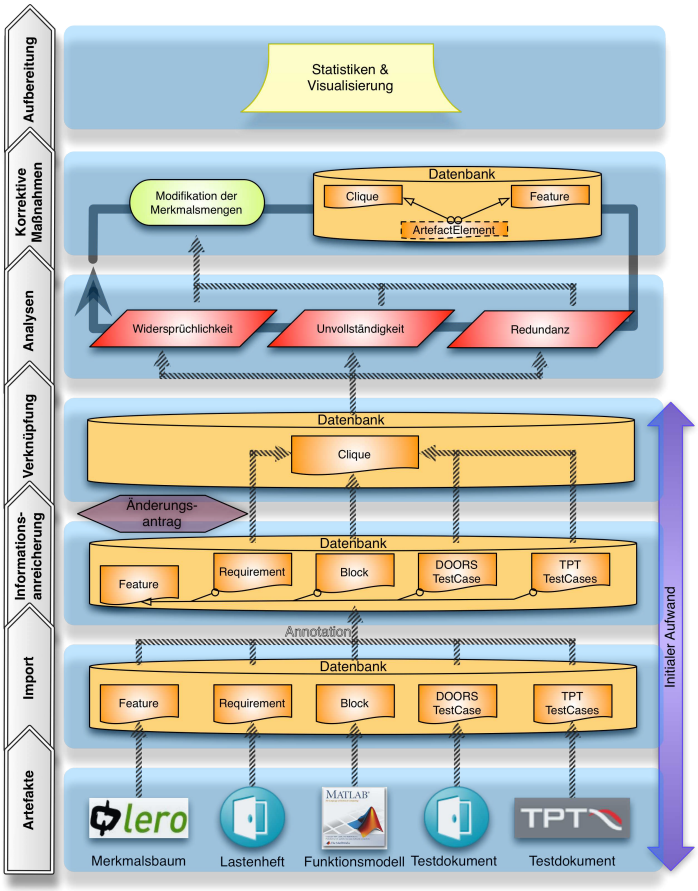

Abbildung 3: Arbeitsprozess des Konsistenzsmanagements von Artefakten der Produktlinien-Entwicklung

werden, die essentiell zur Ausführung der präsentierten Analysen sind. Da die Konsistenzanalysen der Produktlinienartefakte auch artefaktübergreifend ausgeführt werden, müssen die Artefakte und deren Elemente miteinander verknüpft werden. Zur Formalisierung dieser Verknüpfungen wird der Begriff *Clique* eingeführt. Dafür werden Artefakte als Mengen von Artefaktelementen betrachtet. Behandelt werden das Lastenheft $R = \{r_1, ..., r_n\}$, das Funktionsmodell $I = \{b_1, ..., b_n\}$ und das Testdokument $T = \{t_1, ..., t_n\}$. Eine *Clique* $C = (R', B', T')$ verknüpft logisch zusammengehörige Artefaktelemente $R' \subseteq R, B' \subseteq I, T' \subseteq T$ miteinander. Auf diese Art und Weise lassen sich komplexe n:m-Beziehungen, die zwischen den Elementen der verschiedenen Artefakte und innerhalb eines Artefakts bestehen, formalisieren und in der Datenbank hinterlegen. Als Variante wird die Teilmenge eines Artefakts $V \subseteq A \in \{R, I, T\}$ bezeichnet, dessen Artefaktelemente $a_i^V \in V$ gemeinsam Teil einer aus der Produktlinie ableitbaren

Variante $V = \{a_1^V, ..., a_n^V\}$ sein können. Nach erfolgter Informationsanreicherung und Verknüpfung von Artefaktelementen können Konsistenzanalysen ausgeführt und identifizierte Inkonsistenzen klassifiziert werden. Unterschieden werden Inkonsistenzen durch Widersprüchlichkeit, Unvollständigkeit und Redundanz. Im Anschluss an die Analyse können korrektive Maßnahmen vorgenommen werden. Korrektive Maßnahmen können wiederum unerwartete Auswirkungen auf andere Artefaktelemente haben. Daher ist die Ausführung von Konsistenzanalysen und korrektiven Maßnahmen ein Kreislauf, der wiederholt werden sollte, bis ein stabiler Zustand erreicht wird. Verbleibende Inkonsistenzen zeigen potenzielle Fehler der Produktlinienentwicklung oder der Dokumentation dieser auf, welche die Basis der Informationsanreicherung über Annotationen ist. Der Anwender wird dabei zusätzlich durch Statistiken und Visualisierungen zur Auswertung der Analysen unterstützt. Dies beinhaltet unter anderem Sichten auf nicht verknüpfte bzw. nicht annotierte Artefaktelemente.

Das hier präsentierte Konsistenzmanagement basiert auf Merkmalen. Mittels der Merkmalsabbildung $f : A \rightarrow Pot(M),\ A \in \{R, I, T\}$ wird eine Menge von Artefaktelementen auf eine Menge von Merkmalen abgebildet $f(A') = \{f_1, ..., f_n\}$ mit $A' \subseteq A$. Dient beispielsweise Testfall t_1 zur Prüfung des Merkmals f_1 und Testfall t_2 ist mit den Merkmalen f_2 und f_3 verknüpft, so gilt $f(t_1) = \{f_1\}$ und $f(t_2) = \{f_2, f_3\}$. Sind beide Testfälle mittels einer Clique $C = (R', B', T')$, $T' = \{t_1, t_2\}$ miteinander verknüpft, so gilt für $f(T') = \{f_1, f_2, f_3\}$.

Um defizitäre Artefaktverknüpfungen anhand von Merkmale automatisiert zu erkennen, muss ein Artefakt zum Vergleichspunkt erklärt werden. Zu diesem Zweck wird der Begriff der *Merkmalsbasis* eingeführt. Als *Merkmalsbasis* F_B wird die Menge der Artefaktelemente eines Artefakts $A \in \{R, I, T\}$ bezeichnet, deren Verknüpfung mit Merkmalen als korrekt angenommen wird. Der Prozess der Systemspezifikation involviert eine Vielzahl von sowohl internen als auch externen Personen. Somit ist das Lastenheft intensiv diskutiert und oftmals durch einen zweiten Prüfprozess eines anderen Teams verifiziert worden [CNHR13], weshalb es sich als Merkmalsbasis anbietet und im folgenden als solche angenommen wird. Da jedes Artefakt des Entwicklungsprozesses aber die selbe Produktlinie aus verschiedenen Blickwinkeln beschreibt, ist es zur Unterstützung der Entwickler sinnvoll, den Vergleichspunkt sprich die Merkmalsbasis wechseln zu können.

3.1 Informationsanreicherung

Das Konzept der Verknüpfung von Artefaktelementen mit entwicklungsrelevanten Meta-Informationen über Annotationen wurde in [MDR+12] und [Pot12] präsentiert. Dieses unterstützt in Kombination mit dem datenbankbasierten Ansatz die automatisierte Analyse, die Eindeutigkeit von Meta-Informationen, die gezielte Annotation von Artefaktelementen und ermöglicht eine Kategorisierung der Meta-Informationen. Dies wird dadurch realisiert, dass jede Annotation einen festen Typ hat, dem ein Wert zugewiesen wird: *<TYPE, VALUE>*. Um die Verknüpfung von Artefaktelementen mit Merkmalen über Annotationen auszudrücken, werden zwei Annotationstypes eingeführt. *Feature Link* drückt aus, dass ein Artefaktelement mit einem Merkmal verknüpft ist und *Excluded Feature* wird verwendet,

Inkonsistenz durch	
Widersprüchlichkeit	Unvollständigkeit
1. $\left\lvert\left(\bigcup_{b\in B} f(b)\right) \cup \left(\bigcup_{a\in A} f(a)\right)\right\rvert \geq 1$ 2. $\left(\bigcup_{b\in B} f(b)\right) \cap \left(\bigcup_{a\in A} f(a)\right) = \emptyset$	1. $\left(\bigcup_{b\in B} f(b)\right) \supset \left(\bigcup_{a\in A} f(a)\right)$ 2. $\left(\bigcup_{b\in B} f(b)\right) \cap \left(\bigcup_{a\in A} f(a)\right) \neq \emptyset$
Redundanz	**Redundanz und Unvollständigkeit**
1. $\left(\bigcup_{b\in B} f(b)\right) \subset \left(\bigcup_{a\in A} f(a)\right)$ 2. $\left(\bigcup_{b\in B} f(b)\right) \cap \left(\bigcup_{a\in A} f(a)\right) \neq \emptyset$	1. $\left(\bigcup_{b\in B} f(b)\right) \setminus \left(\bigcup_{a\in A} f(a)\right) \neq \emptyset$ 2. $\left(\bigcup_{a\in A} f(a)\right) \setminus \left(\bigcup_{b\in B} f(b)\right) \neq \emptyset$ 3. $\left(\bigcup_{b\in B} f(b)\right) \cap \left(\bigcup_{a\in A} f(a)\right) \neq \emptyset$

Tabelle 1: Definitionen der artefaktübergreifenden Inkonsistenzen

wenn das Artefaktelement in keinem Produkt mit dem entsprechenden Merkmal konfiguriert werden kann. Die möglichen Werte der Annotation sind die durch den Merkmalsbaum vorgegebenen Merkmale. Über einen weiteren Annotationstyp *Variant* wird in der Datenbank hinterlegt, welche Artefaktelemente gemeinsam in einer Variante vorkommen können. Durch Verwendung dieser drei Meta-Informationen kann neben der artefaktübergreifenden Analyse eine aus der Produktlinie ableitbare Variante auf widersprüchliche Merkmale analysiert werden.

3.2 Artefaktübergreifende Inkonsistenz

Der über die Cliquen gegebene Zusammenhang zwischen Artefaktelementen ist der Ausgangspunkt für artefaktübergreifende Konsistenzanalysen, die *Inter-Artefakt-Analysen*. Die Inter-Artefakt-Analysen analysieren die durch eine Clique $C = (R', I', T')$ verknüpften Artefaktelemente $R' \subseteq R, I' \subseteq I, T' \subseteq T$, aller im Entwicklungsprozess vorkommenden Artefakttypen R, I, T auf zueinander konsistente Merkmalsmengen. Miteinander verglichen werden die Merkmalsmenge der Anforderungen $f(R') = \bigcup_{r\in R'} f(r)$, die Merkmalsmenge der Funktionsmodelle $f(I') = \bigcup_{b\in I'} f(b)$ und die Merkmalsmenge der Testfälle $f(T') = \bigcup_{t\in T'} f(t)$, die durch eine Clique miteinander verknüpft sind. Eine Clique $C = (R', I', T')$ ist genau dann merkmalskonsistent, wenn für alle $r \in R', b \in I', t \in T'$ gilt $\bigcup_r f(r) = \bigcup_b f(b) = \bigcup_t f(t)$.

Ist eine Clique $C = (R', I', T')$ nicht merkmalskonsistent, werden drei Arten der artefaktübergreifenden Inkonsistenz unterschieden, von denen zwei in Kombination auftreten können. Seien $A, B \in \{R', I', T'\}, A \neq B$ und $b \in F_B$. In Tabelle 1 sind die Definitionen für die verschiedenen Arten der artefaktübergreifenden Inkonsistenz gelistet. Um die algorithmische Behandlung der Kombination von Unvollständigkeit und Redundanz zu erleichtern und von der alleinigen Unvollständigkeit bzw. Redundanz abzugrenzen, ist das kombinierte Auftreten der beiden Inkonsistenzarten ebenfalls als eigenständige Definition aufgeführt.

3.3 Artefaktinhärente Inkonsistenz

Die Intra-Artefakt-Analyse analysiert die Merkmalsverknüpfungen von Artefaktelementen eines spezifischen Artefakttyps auf Konsistenz. Miteinander verglichen werden die Artefaktelemente $a_i^V \in V$, die gemeinsam in einer Variante V der Produktlinie vorkommen. Dabei werden Inkonsistenzen identifiziert, die durch widersprüchliche Merkmalsverknüpfungen in einem Artefakttyp auftreten können, da nicht alle Systemelemente, die in verschiedenen Varianten einer Produktlinie verbaut werden können, miteinander kombinierbar sind. Eine widersprüchliche Verknüpfung existiert genau dann, wenn ein Merkmal in einer Produktkonfiguration durch die Artefaktelemente gleichzeitig verwendet und ausgeschlossen wird. Beispielsweise kann ein Dieselmotor nicht mit einem Vergaser in einer Variante kombiniert werden, beide werden aber durch das Lastenheft, das Funktionsmodell und entsprechende Testfälle abgedeckt.

Eine aus der Produktlinie ableitbare Variante ist genau dann gültig, wenn für eine Variante V gilt, dass φ erfüllbar ist, wobei

$$\varphi = \bigwedge_{a_i^V \in V} \left(\left(\bigwedge_{f_{ij} \in f\left(a_i^V\right)} f_{ij} \right) \wedge \left(\bigwedge_{f_{ij}^{ex} \in f^{ex}\left(a_i^V\right)} \neg f_{ij}^{ex} \right) \right)$$

$$f\left(a_i^V\right) = \{f_{i1}, ..., f_{in}\}, f^{ex}\left(a_i^V\right) = \{f_{i1}^{ex}, ..., f_{im}^{ex}\}$$

f_{ij} und f_{ij}^{ex} bezeichnen hier aussagenlogische Variablen zur Repräsentation verknüpfter Merkmale bzw. ausgeschlossener Merkmale in der Formel. Die explizit von einer Variante ausgeschlossenen Merkmale werden mit f_{ij}^{ex} notiert.

Die Intra-Artefakt-Analyse nutzt zur Identifikation von artefaktinhärenten Inkonsistenzen einen Sat-Solver. Ist die Formel unerfüllbar, liefern die Erklärungen des Sat-Solver die widersprüchlich verknüpften Merkmale an Artefaktelementen eines Produkts.

3.4 Priorisierung von identifizierten artefaktübergreifenden Inkonsistenzen

Den definierten Inkonsistenzen werden Prioritäten in Abhängigkeit der betroffenen Artefaktelemente und deren Merkmalsverknüpfungen zugeordnet. Dies erlaubt eine automatisierte Einstufung der im folgenden Abschnitt eingeführten Korrekturvorschläge. Je höher die Priorität, desto eher sollten korrektive Maßnahmen erfolgen. Für den Fall, dass anstelle der Konsistenzherstellung lediglich Optimierungen möglich sind, können die Maßnahmen nach der Reduktion der Priorität ebenfalls automatisiert eingestuft werden. Die Priorität berechnet sich folgendermaßen: *Priorität = Offset + Summand*. Sowohl Offset als auch Summand variieren nach der Art der Inkonsistenz und der Anzahl der defizitären Merkmale. Der Einfluss des Offsets auf die Priorität beschreibt die Wichtigkeit der Korrektur in Abhängigkeit der Art der Inkonsistenz. Durch den Summand wirken die problematischen Merkmale mit in die Priorität ein und erlauben so eine Differenzierung bei Inkonsistenzen

	Inkonsistenz	Offset	Summand
1	Widersprüchlichkeit	20000	$\left\lvert\bigcup_{b\in B} f(b)\right\rvert + \left\lvert\bigcup_{a\in A} f(a)\right\rvert$
2	Unvollständigkeit	10000	$\left\lvert\left(\bigcup_{b\in B} f(b)\right) \setminus \left(\bigcup_{a\in A} f(a)\right)\right\rvert$
3	Redundanz	0	$\left\lvert\left(\bigcup_{a\in A} f(a)\right) \setminus \left(\bigcup_{b\in B} f(b)\right)\right\rvert$
4	Unvollständigkeit & Redundanz	10000	$\left\lvert\left(\bigcup_{b\in B} f(b)\right) \setminus \left(\bigcup_{a\in A} f(a)\right)\right\rvert$ + $\left\lvert\left(\bigcup_{a\in A} f(a)\right) \setminus \left(\bigcup_{b\in B} f(b)\right)\right\rvert$

Tabelle 2: Berechnung der Priorität von identifizierten Inkonsistenzen in einer Clique $C = (R', I', T')$ mit $B \in \{R', I', T'\}$ und $B \subseteq F_B$

der gleichen Art. In Tabelle 2 wird die exakte Berechnung der Priorität in Bezug auf identifizierte artefaktübergreifende Inkonsistenzen beschrieben. Durch die Wahl der großen Offsets ist ausreichend Spielraum vorhanden, um eine Vielzahl betroffener Artefaktelemente und Merkmale zu berücksichtigen.

3.5 Korrektive Maßnahmen

Zu beachten bei jeglichen korrektiven Maßnahme ist, dass Änderungen an einem oder mehreren Artefaktelementen einer Clique, Ursache für neu erzeugte Inkonsistenzen sein können. Dies ist dem Umstand geschuldet, dass Artefaktelemente in mehr als einer Clique enthalten sein können.

3.5.1 Korrektur artefaktübergreifender Inkonsistenzen

Inkonsistenzen werden aufgrund zueinander inkonsistenter Merkmalsmengen verknüpfter Artefaktelemente identifiziert und klassifiziert. Zur Korrektur müssen demzufolge die resultierenden Merkmalsmengen verändert werden. Dies kann dadurch geschehen, dass Artefaktelemente mit weiteren Merkmalen verknüpft werden, Verknüpfungen von Merkmalen zu Artefaktelementen gelöst werden oder Verknüpfungen von Artefaktelementen zu anderen Artefaktelementen verändert werden. Tabelle 3 listet alle Optionen, die bei einer Inkonsistenz durch Redundanz theoretisch vollzogen werden könnten. Die mit *x* markierten Operationen werden gleichzeitig ausgeführt. Die Auswirkung auf die bestehende Inkonsistenz wird in der Spalte *Res* festgehalten, die auf die Priorität in der Spalte *Prio*. Es gilt folgende Notation.

- N - Die Menge der geänderten Merkmale
- D - Die Differenzmenge der Merkmalsmengen $D = f(A) \setminus f(B)$
- C - Die Vereinigung der Merkmalsmengen $f(B) \cup f(A)$

	$N^{+B}_{\neq C}$	$N^{+B}_{\subset D}$	$N^{+B}_{=D}$	$N^{+B}_{\supset D}$	$N^{-A}_{\subset D}$	$N^{-A}_{=D}$	$N^{-A}_{\supset D}$	Res	Prio
1	x							U, R	$\nearrow$
2		x						R	$\searrow$
3			x					K	0
4				x				U	$\nearrow$
5					x			R	$\searrow$
6						x		K	0
7							x	U	$\nearrow$
8		x			x			K, R, U	$\updownarrow$
9		x				x		U	$\nearrow$
10		x					x	U	$\nearrow$
11			x		x			U	$\nearrow$
12			x			x		U	$\nearrow$
13			x				x	U	$\nearrow$
14				x	x			U	$\nearrow$
15				x		x		U	$\nearrow$
16				x			x	U	$\nearrow$

Tabelle 3: Möglichkeiten korrektiver Maßnahmen bzgl. Redundanz

- Ink. d. Unvollständigkeit U, Ink. d. Redundanz R, konsistent K
- Priorität steigend $\nearrow$, Priorität sinkend $\searrow$, Priorität nicht vorhersagbar $\updownarrow$

$+B$ gibt an, dass die Merkmalsmenge N zu $f(B)$ hinzugefügt wird, analog bedeutet $-A$, dass Merkmale aus der Merkmalsmenge $f(A)$ entfernt werden.

Im Folgenden werden die Fälle *3, 6* und *8* gesondert betrachtet, da diese in speziellen Fällen für automatisiert ausführbare Korrekturen geeignet sind. Dies ist der Fall, wenn Artefaktelemente existieren, die wenn sie zu einer inkonsistenten Clique hinzugefügt bzw. entfernt werden, aufgrund der mit ihnen verknüpften Merkmalen, in der Clique vorhandene Inkonsistenzen beheben können.

Fall 3 Es wird genau die Menge an Merkmalen ($N = D$) zu $f(B)$ hinzugefügt, die in $f(A)$ redundant ist. Durch Manipulation der Artefaktverknüpfungen und/oder Merkmalsverknüpfungen lässt sich ein konsistenter Zustand erreichen. Artefaktelemente mit redundanten

Merkmalen können mit bisher nicht in der Clique enthaltenen Artefaktelementen verknüpft werden und/oder die Merkmalsverknüpfung der Artefaktelemente aus B wird korrigiert.

Fall 6 Durch Entfernen von Artefaktelementen und/oder einer Reduktion von Merkmalsverknüpfungen kann die Inkonsistenz aufgelöst werden.

Fall 8 Wenn $N^{+B} \cup N^{-A} = D$ und $N^{+B} \cap N^{-A} = \emptyset$ erfüllt ist, kann Konsistenz hergestellt werden. Die Priorität sinkt dann auf 0. Gilt $N^{+B} \cap N^{-A} \neq \emptyset$, resultiert daraus eine Inkonsistenz durch Unvollständigkeit. Für $D \setminus \left(N^{+B} \cup N^{-A}\right) \neq \emptyset$ bleibt die Inkonsistenz durch Redundanz, allerdings mit verbesserter Priorität.

Die korrektiven Maßnahmen bezüglich der Inkonsistenz durch Unvollständigkeit sind vergleichbar mit denen der Inkonsistenz durch Redundanz. Durch einen Wechsel der Merkmalsbasis ergibt sich aus der Inkonsistenz durch Redundanz eine Inkonsistenz durch Unvollständigkeit. Aus diesem Grund werden die ausführbaren Operationen für eine Inkonsistenz durch Unvollständigkeit hier nicht mehr im Detail ausgeführt. Besteht eine Inkonsistenz durch Widersprüchlichkeit und die Verknüpfung der Artefaktelemente ist korrekt, sollte die Verknüpfung zwischen den Artefaktelementen entfernt werden. Ist die Verknüpfung richtig und die Merkmalsverknüpfung fehlerhaft, ist die Inkonsistenz durch eine Substitution der Merkmale durch die der Merkmalsbasis eine Option zur Korrektur.

Als Beispiel wird die in Abbildung 2 illustrierte artefaktübergreifende Inkonsistenz mit den Anforderungen als Merkmalsbasis betrachtet. Die zusammengehörigen Artefaktelemente bestehend aus Testfällen, Anforderungen und Funktionsmodellen werden als Clique $C = (R', I', T')$ formalisiert. Mittels der Merkmalsabbildung kann die Art der Inkonsistenz bestimmt werden. Die Merkmalsmengen sind $f(R') = \{A, B\}$, $f(I') = \{A, B\}$ und $f(T') = \{B, C\}$. Da $f(R') = f(I')$ besteht an dieser Stelle keine Inkonsistenz. Aber da $f(R') \backslash f(T') = \{A\} \neq \emptyset$, $f(T') \backslash f(R') = \{C\} \neq \emptyset$ und $f(R') \cap f(T') = \{B\} \neq \emptyset$ ist hier eine Inkonsistenz durch Unvollständigkeit und Redundanz vorhanden. Um die Inkonsistenz zu korrigieren, werden die Testfälle näher betrachtet. Bestehen diese beispielsweise aus zwei Testfällen $T' = \{t_1, t_2\}$ (nicht in der Abbildung 2 enthalten) mit $f(t_1) = \{B\}$ und $f(t_2) = \{C\}$, so kann durch Entfernen von t_2 aus T' die Inkonsistenz durch Redundanz korrigiert werden. Findet sich des Weiteren ein Testfall t_3 mit $f(t_3) = \{A\}$, so kann durch Hinzufügen von t_3 zu T' ein konsistenter Zustand erreicht werden.

3.5.2 Tool-Support

Realisiert wurde das hier vorgestellte Konsistenzmanagement in einem Werkzeug, genannt *Artshop*. Angelehnt an den Arbeitsprozess aus Abbildung 3 steht dem Anwender bei jedem Schritt eine eigenständige Perspektive und/oder einen Wizard zur Verfügung. Unterstützt wird der Merkmalsbaum des S2T2 Configurators, das mit IBM Rational DOORS festgehaltene Lastenheft, die Funktionsmodelle mittels Matlab/Simulink und die Testfälle formuliert mit IBM Rational DOORS und PikeTec TPT. Das technische Fundament des Artshops bilden Eclipse e4 und Hibernate mit MySQL als Datenbank. Ausführlichere Informationen

zu dem Aufbau der Datenbank, dem Annotationskonzept und der praktischen Umsetzung des Artshop finden sich in [MPBK11, MPK12, Pot12, MDR+12].

3.6 Evaluation

Die Effizienz der Algorithmen zur Durchführung der Analysen ist ein entscheidendes Kriterium, ob das Konsistenzmanagement im praktischen Alltag eingesetzt werden kann. Prinzipiell besteht die algorithmische Ausführung der Inter-Artefakt-Analyse aus der Wiederholung von zwei Schritten. Diese sind die Bildung der Merkmalsmenge und deren anschließender Vergleich. Werden die Merkmalsmengen mittels Hashsets verwaltet gilt für Schritt $1 \in \mathcal{O}(n)$. Auch der Vergleich der Merkmalsmengen mittels Mengenoperationen ist in linearer Zeit durchführbar. Somit gilt für die Inter-Artefakt-Analyse die Ausführbarkeit in linearer Zeit. Zur Beurteilung der Intra-Artefakt-Analyse werden ebenfalls zwei Schritte betrachtet. Der erste Schritt ist die Bildung der Formel anhand der verknüpften und ausgeschlossenen Merkmale der Artefaktelemente, welche im zweiten Schritt mittels eines Sat-Solvers gelöst werden muss. Die Bildung der Formel erfordert eine Iteration über alle Artefaktelemente einer Variante. Somit ist Schritt 1 der Intra-Artefakt-Analyse begrenzt durch die Anzahl der Artefaktelemente der zu prüfenden Variante und damit in Linearzeit durchführbar. Der Flaschenhals ist das Lösen der Sat-Formel, welches bei Verwendung einer Implementierung des DPLL-Verfahrens [DP60] eine Worst-Case Laufzeit von $\mathcal{O}(e^n)$ hat. Aufgrund der geringen Größe der Formel, die beschränkt ist durch die Anzahl der Merkmale, ist die Intra-Artefakt-Analyse trotz exponentieller Laufzeit aber effizient durchführbar.

4 Fazit und Ausblick

In dieser Arbeit wurden zwei miteinander kombinierbare Ansätze zum Konsistenzmanagement präsentiert. Der erste Ansatz betrachtet die artefaktübergreifende Konsistenz und ergänzt die von Cmyrev et al. [CNHR13] vorgestellten Konzepte zur Identifikation und Korrektur von artefaktübergreifenden Inkonsistenzen um formalen Definitionen für auf Merkmalen basierende Artefakte sowie die Merkmalsbasis, die Clique und eine Prioritätsberechnung. In Kombination mit Annotation [MDR+12] und einem datenbankorientiertem Rahmenwerk [MDR+12] können beide Ansätze zum Konsistenzmanagement über beliebige Artefakttypen werkzeuggestützt und semiautomatisch ausgeführt werden. Der zweite Ansatz betrachtet die Konsistenz eines Artefakttyps in sich. Ergänzt um die Meta-Information, welche Merkmale explizit von Artefaktelementen ausgeschlossen sind und zu welcher ableitbaren Variante ein Artefaktelement gehören kann, wurde eine Technik zur Identifikation von artefaktinhärenten Inkonsistenzen präsentiert. Die Umsetzung mittels Sat-Solver erlaubt eine Identifikation der Artefaktelemente, welche Ursache der Inkonsistenz sind. Eine bestehende Hürde der präsentierten Ansätze ist der zwingend erforderliche initiale Aufwand der Annotation der Artefaktelemente. Ein durchschnittliches Modell beinhaltet ca. 100 Merkmale. Große Modelle können aber auch mehrere tausend Merkmale beinhalten [CNHR13]. Ansätze, die die für das Konsistenzmanagement nötigen Meta-Informationen

semiautomatisch aus den Artefaktelementen extrahieren können, existieren teilweise. Um Merkmal-Informationen aus natürlichsprachlich formulierten Artefaktelementen zu extrahieren, kann beispielsweise der MIA-Algorithmus verwendet werden. Dieser führt eine lexikalische Analyse der Artefaktelemente durch und verknüpft diese durch Einsatz eines Wörterbuchs mit erkannten Merkmalen [BH11]. Anhand von zuvor definierten Mustern können aus dem 150%-Modell von Matlab/Simulink enthaltene Variabilitätspunke erkannt und Artefakte eines anderen verknüpften Artefakttyps abgeleitet werden, die nur die Artefaktelemente beinhalten, welche in einer Variante gemeinsam vorkommen. Eine Integration dieser Techniken kann genutzt werden, um automatisiert Varianten abzuleiten und deren Konsistenz mit der Intra-Artefakt-Analyse zu analysieren. Da aus dem 100%-Modell einer Variante auch ein Lastenheft abgeleitet werden kann, welches nur die Anforderungen des 100%-Modells enthält, wären die Entwickler in der Lage, das 150%-Modell auf Basis der mit den Anforderungen verknüpften Meta-Informationen zu analysieren. Interessante Ansätze dazu sind [TDH11] und [PMB+12]. Auch die Aussagekraft der mathematischen Logik kann intensiver genutzt werden, um komplexere Zusammenhänge zwischen Artefaktelementen sowohl artefaktübergreifend als auch innerhalb eines Artefakts in einer Formel auszudrücken. Durch eine Formalisierung der Annotationen mittels mathematischer Logik würden sich auch die Meta-Informationen in die Formel integrieren lassen. Dies würde beispielsweise die Formalisierung der Aussage ermöglichen, dass sobald eine Anforderung mit Merkmal X in einer Variante enthalten ist, auch eine Anforderung mit Merkmal Y enthalten sein muss.

Literatur

[BH11] E. Boutkova und F. Houdek. Semi-automatic identification of features in requirement specifications. In *Proceedings of the 2011 IEEE 19th International Requirements Engineering Conference*, RE '11, Seiten 313–318. IEEE Computer Society, 2011.

[CNHR13] A. Cmyrev, R. Noerenberg, D. Hopp und R. Reissing. Consistency Checking of Feature Mapping Between Requirements and Test Artefacts. In *Concurrent Engineering Approaches for Sustainable Product Development in a Multi-Disciplinary Environment*, Seiten 121–132. Springer London, 2013.

[DP60] M. Davis und H. Putnam. A Computing Procedure for Quantification Theory. *J. ACM*, 7(3):201–215, Juli 1960.

[Far11] T. Farkas. *Regelbasierte Konformitätsprüfung kollaborativer Artefakte*. Dissertation, 2011.

[GS07] H. Gomaa und M. Shin. Automated Software Product Line Engineering and Product Derivation. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, Seiten 285a–285a, 2007.

[Hel05] B. Hellingrath. *Future Automotive Industry Structure 2015 - die neue Arbeitsteilung in der Automobilindustrie*. VDI-Gesellschaft, 2005.

[IO05] M. Ilieva und O. Ormandjieva. Automatic Transition of Natural Language Software Requirements Specification into Formal Presentation. In *Natural Language Processing and Information Systems*, Jgg. 3513 of *Lecture Notes in Computer Science*, Seiten 392–397. Springer Berlin Heidelberg, 2005.

[LP08] K. Lauenroth und K. Pohl. Dynamic Consistency Checking of Domain Requirements in Product Line Engineering. In *International Requirements Engineering, 2008. RE 08. 16th IEEE*, Seiten 193–202, 2008.

[MDR+12] D. Merschen, Y. Duhr, T. Ringler, B. Hedenetz und S. Kowalewski. Model-Based Analysis of Design Artefacts Applying an Annotation Concept. In *Software Engineering*, Jgg. 198 of *LNI*, Seiten 169–180. GI, 2012.

[MPBK11] D. Merschen, A. Polzer, G. Botterweck und S. Kowalewski. Experiences of applying model-based analysis to support the development of automotive software product lines. In *Proceedings of the 5th Workshop on Variability Modeling of Software-Intensive Systems*, Seiten 141–150. ACM, 2011.

[MPK12] D. Merschen, J. Pott und S. Kowalewski. Integration and Analysis of Design Artefacts in Embedded Software Development. In *COMPSAC Workshops*, Seiten 503–508. IEEE Computer Society, 2012.

[MSR+10] A. Michailidis, U. Spieth, T. Ringler, B. Hedenetz und S. Kowalewski. Test front loading in early stages of automotive software development based on AUTOSAR. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, Seiten 435–440, 2010.

[PMB+12] A. Polzer, D. Merschen, G. Botterweck, A. Pleuss, J. Thomas, B. Hedenetz und S. Kowalewski. Managing complexity and variability of a model-based embedded software product line. *Innov. Syst. Softw. Eng.*, 8(1):35–49, Marz 2012.

[Pot12] J. D. Pott. Ein Annotationskonzept für die modellbasierte Analyse von Prozessartefakten. Diplomarbeit, RWTH Aachen, 2012.

[TDH11] J. Thomas, C. Dziobek und B. Hedenetz. Variability management in the AUTOSAR-based development of applications for in-vehicle systems. In *Proceedings of the 5th Workshop on Variability Modeling of Software-Intensive Systems*, VaMoS '11, Seiten 137–140. ACM, 2011.

[VLDL98] A. Van Lamsweerde, R. Darimont und E. Letier. Managing conflicts in goal-driven requirements engineering. *Software Engineering, IEEE Transactions on*, 24(11):908–926, 1998.