Feature Net – ein Ansatz zur Modellierung von automobilspezifischem Domänenwissen und Anforderungen

J. Hartmann, A. Fleischmann, C. Pfaller, M. Rappl, S. Rittmann, D. Wild

Lehrstuhl für Systems und Software Engineering
Institut für Informatik, Technische Universität München
Boltzmannstraße 3, D-85748 Garching b. München
{hartmanj, fleischa, pfaller, rappl, rittmann, wildd}@in.tum.de

Abstract: In diesem Beitrag wird mit dem *Feature Net*, einer Kombination aus klassischem *Feature Tree* und *Semantic Net*, ein Modell zur strukturierten Erfassung von applikations- und domänenspezifischen Anforderungen eingeführt. Das *Feature Net* unterstützt, als Ergebnis eines modellbasierten Requirements Engineering Prozesses, den Übergang zum Design, sowie, als modellbasierte Repräsentation von Domänenwissen, die gezielte Wiederverwendung von Artefakten.

Schlüsselwörter: Automobilspezifische Modellierung, Domänenwissen, Feature Tree, Feature Net, Klassifikation von Anforderungen, Semantic Net, Wiederverwendung

1 Motivation und Einleitung

Die steigende Bedeutung von Software-Systemen im Automobil sowie deren zunehmende Komplexität führen zu neuen Herausforderungen im Entwicklungsprozess [FGP04]. Die zunehmende Vernetzung einer kontinuierlich wachsenden Anzahl an Software-Funktionen erfordert eine systematische Erfassung der funktionalen Abhängigkeiten sowie Methoden zur frühzeitigen Analyse von Systemfunktionalität hinsichtlich Feature Interaction, Konsistenz und Vollständigkeit. Große Variantenvielfalt, steigender Kostendruck und die Notwendigkeit kürzerer Entwicklungszeiten begründen ein Bedürfnis nach gezielter Wiederverwendung [TH02, MG02]. Eine weitere Herausforderung bei der Entwicklung eingebetteter automobilspezifischer Applikationen stellt die frühzeitige Erfassung von Abhängigkeiten zwischen der zu entwickelnden Software und ihrer Hardwareumgebung dar.

In diesem Papier wird mit dem *Feature Net* ein Ansatz vorgestellt, der darauf abzielt, Anforderungen und deren Abhängigkeiten strukturiert zu erfassen und dadurch Inkonsistenzen zwischen Anforderungen frühzeitig aufzudecken, den Übergang zum Design zu erleichtern sowie gezielte Wiederverwendung zu unterstützen. In Abschnitt 2 wird das *Feature Net* näher definiert und mittels eines automobilspezifischen Beispiels illustriert. Abschnitt 3 skizziert den methodischen Einsatz des *Feature Net*. Abschnitt 4 gibt einen Ausblick auf aktuelle und geplante Weiterentwicklungen des Ansatzes.

2 *Feature Net* – Definition und Ziele

Ein *Feature Net* liefert eine strukturierte Darstellung aller relevanten Features eines Systems. Unter *Features* verstehen wir Bestandteile oder Eigenschaften des SW-/HW-Systems (vgl. [KCH+00]), die durch einen Abstraktionsprozess aus Anforderungen gewonnen werden. Im Wesentlichen werden damit funktionale Einheiten des Systems beschrieben, aber auch HW-Bestandteile des Systems. Auch nichtfunktionale Anforderungen, wie Qualitätsanforderungen, werden als Attribute von Features strukturiert abgelegt.

Jedes *Feature Net* setzt sich zusammen aus einem *Feature Tree* (domänenspezifischer Klassifikationsbaum) und einem *Semantic Net* (vgl. [BL85]), das die Abhängigkeiten zwischen einzelnen Features erfasst. Durch diese Aufspaltung wird eine klare Trennung unterschiedlicher Aspekte erreicht, die bei bekannten Ansätzen wie [KCH+00] oft vermischt werden: Bei unseren *Feature Trees* liegt der Fokus auf der vollständigen Erfassung und hierarchischen Strukturierung aller Features. Weitere Relationen zwischen Features spielen hier keine Rolle; diese werden beim Aufbau des *Semantic Net* untersucht.

In dem Ansatz werden ein *Domain Feature Net*, welches das gesamte Domänenwissen repräsentiert, und verschiedenen *Application Feature Nets*, die nur die Informationen umfassen, die für die Entwicklung einer Applikation benötigt werden, unterschieden. Ersteres liefert einen „Bauteilekatalog“, einen strukturierten Pool an Features, aus dem man sich bei dem Design künftiger Systeme bedienen kann. Somit wird die gezielte Wiederverwendung von Artefakten unterstützt. Ein *Application Feature Net* hingegen ist eine Projektion aus dem *Domain Feature Net* auf ein bestimmtes System. Es stellt eine geeignete Struktur für die formalisierte Erfassung von Anforderungen und deren Beziehungen bereit und erleichtert den Übergang vom Requirements Engineering zum Design.

2.1 Der *Feature Tree*

Die Grobstruktur aller *Feature Trees* ist domänenspezifisch, die weitere Strukturierung hingegen hängt von dem zu modellierenden System ab. Abb. 1 zeigt exemplarisch anhand der Fensterhebung einen Ausschnitt eines *Application Feature Trees*. Jeder *Knoten eines Feature Trees* repräsentiert ein mehr oder weniger fein granulares Feature. Jedem Knoten ist ein *Template* zugeordnet, welches alle relevanten Informationen des Features in strukturierter Form enthält. Der Aufbau des Templates unterscheidet sich für die einzelnen Teilbäume und ist unter anderem geprägt von den Design-Modellen, die zur Modellierung der entsprechenden Features verwendet werden. Die vorgegebene Struktur des Templates vereinfacht den Übergang zum Design und erlaubt eine automatische Transformation in Modellfragmente. Zudem fungieren die Templates als Checkliste und stellen sicher, dass keine relevanten Informationen vergessen werden. Jedes Template beinhaltet Metainformationen (ID, Autor,…), eine Kurzbeschreibung, die Anknüpfungspunkte des *Semantic Net* sowie weitere spezifische Informationen. Allgemein gilt, dass in das Design der Templates Erfahrung und Domänenwissen einfließt.[1]

---

[1] Der genaue Aufbau der Templates kann aus Platzgründen nicht näher erläutert/graphisch dargestellt werden.

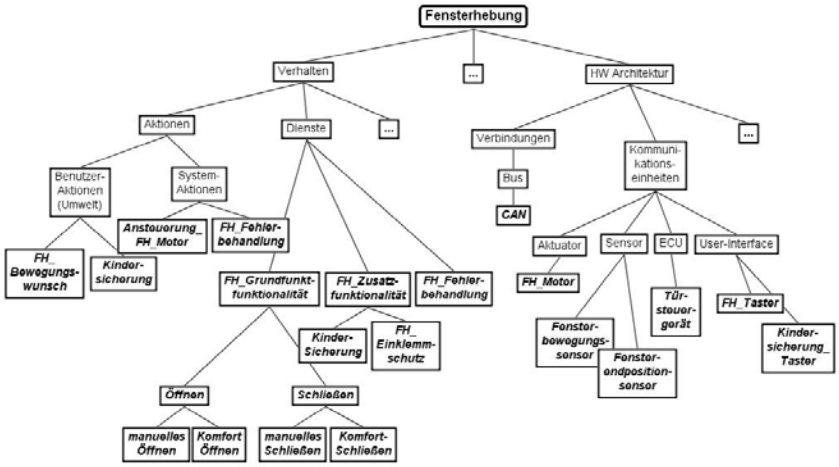

Abb. 1 Ausschnitt aus dem *Application Feature Tree* "Fensterhebung"

Die *Kanten eines Feature Trees* repräsentieren die Klassifikationsbeziehungen zwischen verschiedenen Features, d.h. sie liefern eine hierarchische Einteilung der Feature in verschiedene Klassen. Im Gegensatz zu [KCH+00] gibt der *Feature Tree* in unserem Ansatz keine Auskunft über weitere Abhängigkeiten. Bei dem Aufbau der Feature-Hierarchie wird eine Gruppierung gleichartiger Informationen angestrebt, um Feature-Familien auszuzeichnen und die Vererbung von Eigenschaften zu optimieren. Besonders auf den oberen Ebenen des Baums wird das „Separation of Concerns“-Prinzip angewendet (s. Abb. 1), um domänenspezifische Informationen möglichst vollständig zu erfassen.

2.2 Das Semantic Net

Neben der Klassifikation der Features existieren zahlreiche weitere Beziehungstypen. Mit dem *Semantic Net* wird ein Modell zur Verfügung gestellt, um diese Beziehungen strukturiert zu erfassen. Grundlegende, für die Entwicklung von SW-Systemen im Automobilbereich charakteristische Beziehungstypen sind beispielsweise *hierarchische Relationen* (Dekomposition, Variantenbildung, Spezialisierung), *kausale Relationen* (Voraussetzung, wechselseitiger Ausschluss, Kompatibilität), *Interaktionsrelationen (*Aufrufbeziehung, Sender/Empfängerbeziehung) *oder Deploymentbeziehungen.*

Für die in einem *Feature Tree* klassifizierten Features lässt sich klar definieren, welche Typen von Beziehungen zwischen den Features verschiedener Teilbäume auftreten. Beispielsweise können die Beziehungen zwischen Aktionen und Diensten zu einer Relation „setzt in kausalen Zusammenhang“ abstrahiert werden, welche beschreibt, aus welchen Aktionen sich ein Dienst aufbaut. Diese Relation wiederum stellt einen Spezialfall der Dekomposition dar. Die feste Vorgabe möglicher Beziehungstypen in Verbindung mit einer formalen Fundierung dieser hilft beim Erarbeiten, Strukturieren und Analysieren des oft sehr umfangreichen Beziehungsgeflechts zwischen den Features.

3 Methodischer Einsatz eines *Feature Net*

In diesem Abschnitt wird der methodische Einsatz des *Feature Net* skizziert. Insbesondere wird aufgezeigt, an welcher Stelle im Software-Entwicklungsprozess dieses eingesetzt und wie es methodisch und systematisch erarbeitet werden kann.

3.1 *Feature Net* – Ergebnis der Requirements Engineering Phase

Ein *Feature Net* ist als Endergebnis des Formalisierungsprozesses des Requirements Engineerings anzusehen, da es alle Anforderungen der Requirements Engineering Phase formalisiert abbildet. Für die systematische Erarbeitung eines *(Application) Feature Net* sind folgende Schritte notwendig:

Identifikation und Normalisierung. Zunächst müssen die Anforderungen aus den Eingabedokumenten extrahiert, identifiziert, atomarisiert sowie attributiert werden. Außerdem muss die verwendete Terminologie sprachlich vereinheitlicht werden.

Aufbau des *Feature Trees.* Ausgehend von einer unstrukturierten Menge an normalisierten Anforderungen wird schrittweise ein *Feature Tree* aufgebaut. Hierzu wird die Menge der Anforderungen zunächst entsprechend der domänenspezifischen Grobstruktur unterteilt. Auf den unteren Ebenen werden weitere applikationsspezifische Klassen gebildet, indem Anforderungen mit gemeinsamen Aspekten in Gruppen zusammengeordnet werden. Der Aufbau des *Feature Trees* ist abgeschlossen, wenn eine weitere Unterteilung nicht mehr sinnvoll ist.

Aufbau und Befüllen der Templates. Charakteristische Informationen aus den Anforderungen, die ein spezifisches Feature betreffen, werden in Templates gespeichert und können dadurch eindeutig formuliert und verstanden werden. Die Templates müssen zu Beginn der Analysephase im Requirements Engineering einmal erarbeitet werden. Ziel ist es jedoch, bestehende Templates wiederzuverwenden und dadurch das Domänenwissen, das sich im Aufbau der Templates widerspiegelt, zu nutzen, z.B. um Lücken in den Anforderungen aufzudecken.

Aufbau des *Semantic Net.* Hand in Hand mit der Erstellung des Feature Trees geht die Spezifikation des *Semantic Net* einher, um die Abhängigkeiten zwischen den Features zu erfassen und damit auch Inkonsistenzen aufzudecken. Domänenspezifische Vorgaben, die festlegen, welche Beziehungen zwischen Features existieren, helfen bei der korrekten und vollständigen Erfassung von existierenden Abhängigkeiten. Die über die Zeit erarbeiteten spezifischen Beziehungstypen spiegeln somit Teile des Domänenwissens wider.

3.2 *Feature Net* – Ausgangspunkt des Designs

Die in einem *Feature Net* systematisch gespeicherten Informationen bilden den Ausgangspunkt für das Design. Die eingeführte Strukturierung unterstützt eine Entwicklung über verschiedene Abstraktionsebenen hinweg, da die Anforderungen bereits nach derar-

tigen Gesichtspunkten klassifiziert sind. Von diesen Teilmengen ausgehend können aus den Templates teilweise automatisch Modellfragmente generiert werden. Die formale Erfassung der Beziehungen im *Semantic Net* unterstützt die Integration der einzelnen Modellfragmente und hilft im Umgang mit Problemen wie ungewollter Feature Interaction.

3.3 *Feature Net* – Ausgangsbasis für die gezielte Wiederverwendung

Zum Abschluss wird kurz das Potential des *Feature Nets* in Hinblick auf gezielte Wiederverwendung von Entwicklungsartefakten aufgezeigt. Der *Domain Feature Tree* fungiert hierbei als "Bauteilekatalog". Bei der Entwicklung eines neuen Systems, können im *Domain Feature Tree* passende, existierende Systemteile aufgefunden werden. Die im *Semantic Net* festgehaltenen Beziehungen unterstützen die Navigation im *Feature Tree* und damit das Auffinden wiederverwendbarer Teile. Außerdem zeigen sie Abhängigkeiten auf, die sonst evtl. übersehen worden wären. Werden Systemteile nicht komplett wiederverwendet, sondern leicht modifiziert, helfen die erfassten Beziehungen bei der Identifikation von Auswirkungen der Modifikation und bei der sukzessiven Anpassung der abhängigen Komponenten. Grundlage ist allerdings, dass das *Domain Feature Net* zu jeder Zeit korrekt und vollständig gehalten wird.

4 Zusammenfassung und Ausblick

In diesem Beitrag wurde der modellbasierte Ansatz des *Feature Net* eingeführt und Einsatzszenarien im Entwicklungsprozess wurden aufgezeigt. Erste Fallbeispiele aus dem Automobil-Umfeld zeigen, dass das *Feature Net* ein viel versprechender Ansatz ist, um die aktuellen Herausforderungen in der Automobilindustrie, wie zunehmende Komplexität, Variantenvielfalt und Bedarf nach gezielter Wiederverwendung, zu adressieren und zu lösen. Aktuelle Forschungstätigkeiten konzentrieren sich auf eine formale Fundierung der Theorie, auf eine detaillierte Erarbeitung automobilspezifischer Beziehungstypen innerhalb des *Semantic Net* sowie auf eine umfassende methodische Einbettung des Ansatzes in einen praxisnahen Entwicklungsprozess.

Literaturverzeichnis

[BL85] R.J. Brachman and H.J. Levesque, eds., *Readings in Knowledge Representation*, Morgan Kaufmann, Los Altos, Calif., 1985.

[FGP04] A. Fleischmann, E. Geisberger und M. Pister, *Herausforderungen für das Requirements Engineering eingebetteter Systeme*, Technischer Bericht, TUM-I0414, München, 2004.

[KCH+00] K. C. Kang, S.G. Cohen, J.A. Hess, W. E. Novak, A. S. Peterson, *Feature-Oriented Domain Analysis (FODA) Feasibility Study*, Technical report CMU/SEI-90-TR-021, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1990.

[MG02] J. MacGregor: *Requirements Engineering in Industrial Product Lines*, In: Proceedings of REPL'02 Intern. Workshop Requirements Engineering for Product Lines, pp. 5-11, Essen, 2002.

[TH02] S. Thiel, A. Hein: *Modeling and Using Product Line Variability in Automotive Systems*, IEEE Software 19(4): 66-72, 2002.