Konfigurationsmanagement variantenreicher Simulink-Modelle

Jens Weiland, Ernst Richter

DaimlerChrysler AG Forschung und Technologie
Postfach 2360, D-89013 Ulm
{jens.weiland, ernst.richter}@daimlerchrysler.com

Abstract: Automotive embedded Software ist gekennzeichnet durch ein hohes Maß an Variabilität. Besondere Herausforderung liegt hierbei im Management der Variantenkonfiguration. Der Beitrag beschreibt für die Modell-basierte Softwareentwicklung von embedded Software Vorgehensweisen zur Verknüpfung variantenreicher Matlab/Simulink-Modelle mit dem Konfigurationsmanagement für Produktvarianten.

1 Einleitung

Embedded Software, die im Bereich Automotive eingesetzt wird, ist gekennzeichnet durch ein hohes Maß an Variabilität. Das Auftreten dieser Variabilität ist vielschichtig aufgrund einer Vielzahl an Fahrzeugkonfigurationen und Anforderungen auf Basis unterschiedlicher Hardware, Absatzmärkte und Regularien [SZ03].

Die besondere Herausforderung liegt im Konfigurationsmanagement für die Produktvarianten[1]. Gerade bei der Wartung und Weiterentwicklung der embedded Software sind für den Entwickler, der nicht mit der Erstellung der Software betraut war, Varianten und deren Abhängigkeiten nur bedingt erkennbar. Dies kann bei steigender Zahl an Softwarevarianten zu erheblichem Wartungsaufwand bzw. zu fehlerhaften Implementierungen führen.

In unserem Kontext umfaßt die Variantenkonfiguration

- Konzepte zur Modellierung der Variabilität in Architekturmodellen (beispielsweise mit Matlab/Simulink [MW04]) oder direkt im Sourcecode und
- Konzepte für das Management von Variabilität (Verwaltung und Darstellung der Variabilität mit einem Konfigurationswerkzeug) und
- Konzepte für das Auflösen der Variabilität zur Instanziierung gültiger Produktvarianten.

[1] Dies gilt für die konventionelle als auch für die modellbasierte Softwareentwicklung im gleichen Maße.

Im Rahmen dieses Beitrags wird das Vorgehen zur Verknüpfung variantenreicher Architekturmodelle – hier generische Simulink-Modelle[2] – mit dem Management der Variabilität für Produktvarianten betrachtet. Das Ziel ist die Teilautomatisierung der Variantenkonfiguration im Rahmen der modellbasierten Entwicklung von embedded Softwarevarianten.

2 Variabilitätsmanagement

Im Rahmen des Variabilitätsmanagements orientiert sich unsere Vorgehen am generativen Domänenmodell [CE00]. Dieses trennt Anwendungs-orientierte Konzepte von den Konzepten der Implementierung. Ein Werkzeug, welches das generative Domänenmodell unterstützt, ist pure::variants von pure-systems [PS04], welches mit dem Merkmalmodell und dem Familienmodell diese Trennung umsetzt.

Im Merkmalmodell werden die Merkmale der untersuchten Domäne verwaltet und hierarchisch strukturiert. Das Ergebnis ist ein umfassendes Modell der gemeinsamen und variablen Merkmale sowie deren Abhängigkeiten zwischen den Produktvarianten. Unterschieden werden verbindliche und optionale Merkmale sowie (1..n):m-Gruppenbeziehungen. Jedes Merkmal kann eine Menge von Attribut-Werte-Paaren besitzen. Abhängigkeiten zwischen Merkmalen / Attributen werden über Restriktionen definiert.

Das Familienmodell verwaltet die Artefakte, wie Dokumente, Variablen, Source-/ Objektdateien, etc., zur Instanziierung der Produktvarianten. Artefakte werden – vergleichbar den Merkmalen im Merkmalmodell – hierarchisch strukturiert abgelegt.

Das Konfigurationswissen zwischen Merkmalen, Attributen und Artefakten wird über Relationen bzw. Restriktionen definiert. Dies ermöglicht die automatische Validierung selektierter Produktvarianten. Mit Hilfe eines XSL Transformators kann der gewünschte Output auf Basis des Merkmalmodells, der Variantenkonfiguration, den Artefakten aus dem Familienmodell und anwendungsspezifischen Transformationen erzeugt werden.

3 Generisches Architekturmodell mit Matlab/Simulink

Zur Modellierung von Variabilität stehen in Simulink die beiden folgenden Modellierungselemente zur Verfügung; beide Modellierungselemente sind Simulink intrinsisch:

- Template-Blöcke ermöglichen das Ersetzen ganzer Simulinkblöcke entsprechend der auftretenden (lokalen) Variabilität. Blockvarianten werden hierbei einem Template-Block als „Member“ zugeordnet. Bei Verwendung von Template-Blöcken im Modell kann eine Auswahl einer gewünschten Variante erfolgen.

[2] Die Modell-basierte Entwicklung von embedded Software auf Basis von Matlab/Simulink findet in steigendem Maße Einzug in die Automotive Softwareentwicklung. Sie ermöglicht neben der Modellierung der Funktionalität auch deren Simulation und Codegenerierung der validierten Modelle.

- Einsatz von (Block-)Parametern (direkt oder – für umfangreiche Datenmengen – über .mat-Files). Anwendungsfälle für Parameter sind z.B. die Einbindung von Varianten-spezifischen Kennlinien oder um Verzweigungsverhalten in Kontrollflüssen (und somit innerhalb von Stateflow Charts) statisch festzulegen, d.h. Varianten-spezifisch zu steuern.

Template-Blöcke und Parameter repräsentieren Variationspunkte und finden ihre Entsprechung implizit oder explizit im Merkmalmodell wieder[3].

Um die Kopplung zwischen dem Management der Variabilität und der Modellierung mit Simulink durchführen zu können, aber auch um ein vereinfachtes Suchen nach derartigen Blöcken zu ermöglichen, müssen sie in Simulink besonders gekennzeichnet werden. Die von uns gewählte Möglichkeit beispielsweise Template-Blöcke als Variationspunkte zu kennzeichnen ist, diesen als Tag die Bezeichnung „VariationPoint:: Merkmalname“ anzufügen. “Merkmalname“ korrespondiert hier mit dem entsprechenden Namen des Variationspunktes im Merkmalmodell. Die Blockvarianten erhalten als Tag den Namen des korrespondierenden Merkmals in der Form „Feature::Merkmalname“.

Unter Anwendung dieser Mechanismen zur Modellierung von Variabilität entsteht ein generisches Simulink-Modell, welches auf die einzelnen Produktvarianten angewendet werden kann.

4 Konfiguration generischer Simulink-Modelle

Im Rahmen der Konfiguration generischer Simulink-Modelle werden im Folgenden zwei Vorgehensweisen beschrieben. Beide Vorgehensweisen wurden anhand eines abgeschlossenen embedded Steuerungssystems exemplarisch umgesetzt. Bei beiden Alternativen werden die gemeinsamen und variablen Merkmale mit dem Konfigurationswerkzeug pure::variants modelliert; der wesentliche Unterschied liegt in der Verwaltung des Konfigurationswissens.

Ziel ist es aus dem generischen Simulink-Modell ein konkretes Modell, d.h. ein Modell einer Produktvariante zu erzeugen. In diesem Modell ist die Variabilität aufgelöst. Die Auflösung der Variabilität geschieht über die Verwendung so genannter „Model Construction Commands“ unter Matlab zur Manipulation von Simulink-Elementen. Auf diese Weise kann automatisiert in ein Simulink-Modell eingegriffen und dieses abgeändert werden.

Eine dritte Alternative, die aktuell untersucht wird, ist die direkte Manipulation von Simulink-Modelldateien außerhalb von Matlab. Auf diese Vorgehensweise wird an dieser Stelle nicht näher eingegangen.

---

[3] Nicht alle Merkmale des Merkmalmodells stellen Variationspunkte dar. Merkmale, die als „mandatory“ klassifiziert sind, und keine variablen Submerkmale besitzen, müssen im Simulink-Modell keine explizite Entsprechung besitzen.

4.1 XML-basierte Variantenkonfiguration

Bei der XML-basierten Variantenkonfiguration werden die Model Construction Commands zur Auflösung der Variabilität im generischen Simulink-Modell in Simulink zusammengestellt.

In pure::variants werden aus dem Merkmalmodell die Merkmale selektiert, die zur Produktvariante gehören. Das Ergebnis ist eine Konfiguration, die im XML-Format abgelegt wird. Sie enthält im Wesentlichen die Hierarchie der ausgewählten Merkmale und deren Attribute.

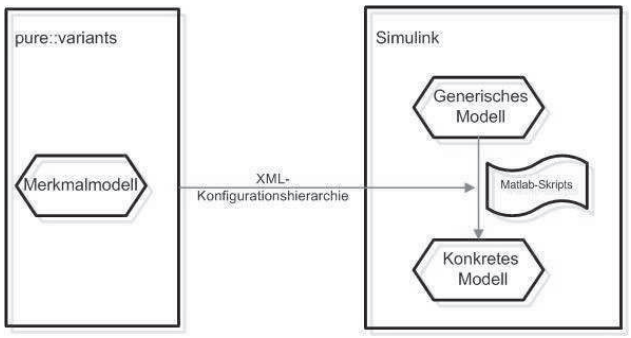

Abbildung 1: XML-basierte Variantenkonfiguration

Diese XML-Datei wird in Matlab importiert, mit vordefinierten Matlab-Skripts (die das Simulink-Modell nach Variationspunkte durchsuchen und Model Construction Commands zum Auflösen der Variabilität ausführen) ausgewertet, und somit die Modellmanipulation veranlasst. Das Ergebnis ist abschließend ein konkretes Modell einer Produktvariante.

4.2 Matlab-Skript-basierte Variantenkonfiguration

Bei dieser Alternative werden die Kommandos zur Auflösung der Variabilität des generischen Simulink-Modells im Konfigurationswerkzeug definiert.

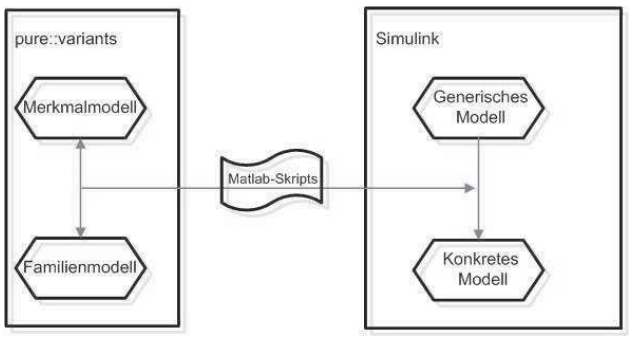

Abbildung 2: Matlab-Skript-basierte Variantenkonfiguration

Hierbei wird in pure::variants bei der Auswahl konkreter Merkmale ein entsprechendes Matlab-Skript erzeugt, welches die spezifischen Kommandos zur Auflösung der Variabilität im generischen Simulink-Modell beinhaltet. Das Skript wird in Matlab abschließend ausgeführt um ein konkretes Modell einer Produktvariante zu erzeugen.

5 Zusammenfassung

In diesem Beitrag wurden Vorgehensweisen zur Verknüpfung variantenreicher Simulink-Modelle mit dem Konfigurationsmanagement von Produktvarianten vorgestellt. Wichtigstes Kriterium für einen Vergleich der Alternativen ist dabei wie konzentriert, strukturiert und übersichtlich das Konfigurationswissen verwaltet wird.

Die Information, wie die Variabilität im generischen Simulink-Modell aufgelöst wird, ist bei der XML-basierten Variantenkonfiguration in Simulink vorhanden – für den Entwickler auf Seite des Konfigurationsmanagements jedoch nicht. Auf den ersten Blick erscheint diese Lösung ideal, da sie sehr allgemein und auf Seiten des Konfigurationsmanagements unabhängig von den zu konfigurierenden Artefakten (hier das Simulink-Modell) einsetzbar ist. Konfiguration ist aber per se ein sehr spezifischer Prozess, bei dem Eigenheiten der zu konfigurierenden Artefakte berücksichtigt werden müssen. Dieser Umstand wird bei der Matlab-Skript-basierten Variantenkonfiguration genutzt um das gesamte Konfigurationswissen dediziert in einem Werkzeug zu verwalten. In Matlab wird lediglich das ausführbare Matlab-Skript – als Ergebnis der Konfiguration einer Produktvariante – importiert. Soll nun im zweiten Fall ein weiteres, andersartiges Artefakt konfiguriert werden – z.B. ist an die Generierung von Dokumentation für eine Produktvariante gedacht – können zugehörige Regeln in einem zweiten Familienmodell gesammelt werden. Über das Konfigurationsmodell einer Variante lassen sich dann das spezifische Simulink-Modell als auch die spezifische Dokumentation erzeugen.

Durch die enge Bindung zwischen der Darstellung der Variabilität (Merkmale und Abhängigkeiten zwischen den Merkmalen) und wie die Variabilität im generischen Simulink-Modell aufgelöst wird, ist die Matlab-Skript-basierte Variantenkonfiguration für den Entwickler explizit sichtbar. Für die XML-basierte Alternative muß hier erst spezialisierte Editor-Funktionalität aufgebaut werden.

Die Matlab-Skript-basierte Variantenkonfiguration ist daher einer XML-basierten Variantenkonfiguration vorzuziehen.

Literaturverzeichnis

[CE00] K. Czarnecki, U. Eisenecker, Generative Programming – Methods, Tools, and Applications, Addison-Wesley, Boston, MA, 2000

[MW04] The MathWorks, Simulink – Simulation and Model-based Design, Natick, MA, 2004

[PS04] pure-systems, pure::variants Eclipse Plugin User Guide, 2004

[SZ03] Jörg Schäuffele, Thomas Zurawka, Automotive Software Engineering – Grundlagen, Prozesse, Methoden und Werkzeuge, Vieweg, Wiesbaden, Juli 2003