

Anwendungserfahrungen und methodische Anpassungen bei der Einführung von Software-Produktlinien

Oliver Wieland, Andreas Hein, Stefan Kowalewski,
John MacGregor, Steffen Thiel

Robert Bosch GmbH, Forschung und Voraentwicklung, Frankfurt am Main
{oliver.wieland|andreas.hein1|stefan.kowalewski|
john.macgregor|steffen.thiel}@de.bosch.com

Abstract: Software-Produktlinien erhalten im Automobilbereich zunehmende Bedeutung. Mit dem *Produktlinienansatz* (PLA) des SEI steht ein methodisches Rahmenwerk zu ihrer Realisierung zur Verfügung. Dieser Beitrag berichtet von Erfahrungen bei der Einführung von Software-Produktlinien nach dem PLA unter Anwendung des Vorgehensmusters *What-to-Build*. Im Anschluss werden Aktivitäten zur Anpassung des PLA an die Erfordernisse software-intensiver Systeme in der zuliefernden Automobil-Industrie vorgestellt.

1 Einleitung

Software-intensive Systeme im Automobil sind durch eine hohe Variantenzahl gekennzeichnet. Ihre Beherrschung erfordert eine systematisch vorbereitete, an den Geschäftsanforderungen orientierte Wiederverwendung aller relevanten Analyse- und Entwurfsergebnisse. Der auf das Software Engineering Institute (SEI) zurückgehende *Produktlinienansatz* (PLA) stellt ein dazu geeignetes Rahmenwerk dar [CN01]. 29 „Practice Areas“, definiert werden. Eine Orientierung für die systematische Bearbeitung der Practice Areas bieten Vorgehensmuster, so genannte PLA-„Patterns“.

Der erste Teil dieses Beitrags berichtet über Erfahrungen, die bei der Einführung von Software-Produktlinien anhand des PLA-Patterns „What-to-build“ als Leitfaden gemacht wurden. Das Anwendungsfeld dabei war die Karosserieelektronik.

Bei der Umsetzung der methodischen Vorgaben des PLA stellt man schnell fest, dass die Methodik mit Blick auf die Besonderheiten der Anwendungsdomäne an zahlreichen Stellen angepasst werden muss. Für die Domäne der software-intensiven Systeme im Automobil tun wir dies gemeinsam mit anderen Partnern im Rahmen der beiden öffentlich geförderten Projekte ITEA-CAFÉ und IST-ConIPF. Der zweite Teil des Beitrags berichtet über Ergebnisse dieser Projekte.

2 Anwendungserfahrung mit dem *What-to-Build*-Pattern

2.1 Das *What-to-Build*-Pattern

Das *What-to-Build*-Pattern (WtBP) beschreibt einen Lösungsweg für die Überführung oder Neuordnung bestehender Produkte in eine SPL für einen gegebenen *Kontext*. Der Kontext eines Patterns beschreibt Situation und Voraussetzungen für dessen Anwendung. Im Fall des *WtBP* hat die Organisation die Einführung von Softwareproduktlinien beschlossen und kennt die möglichen Produkte der Produktlinie. Bei der Umsetzung sind folgende Practice Areas relevant:

Technology Forecasting Untersuchung von Technologietrends

Market Analysis Analyse des Marktes im Hinblick auf Konkurrenzprodukte, Kundenbedürfnisse und mögliche Strategien

Understanding Relevant Domains Untersuchung der Produkte bezüglich ihrer Features, Gemeinsamkeiten und Unterschiede

Scoping Abgrenzung der Produktlinie, d. h. welche Produkte sind Teil der Produktlinie und welche nicht

Building a Business Case Kosten-Nutzen-Analyse der Produktlinie.

Häufig gibt es neben dem Pattern zusätzlich *Varianten*, die auf besondere Gegebenheiten Rücksicht nehmen. Ihre Beschreibung beschränkt sich auf die Abweichungen gegenüber der Standardvariante. Beim WtBP gibt es zwei Varianten, "*Analysis*" und "*Forced March*" [CN01].

Analysis bezieht zusätzlich die Anforderungen und die Softwarearchitektur der Produktlinie mit ein. Diese kann aufgrund der breiteren Datenbasis besser abgegrenzt werden, das Vorgehen ist aber aufwändiger. *Forced March* ist hingegen empfehlenswert, wenn die Produktlinie aus existierenden Produkten aufgebaut oder eine bestehende Produktlinie überarbeitet werden soll. Hier wird die Produktlinie nur bezüglich zukünftiger Anforderungen geprüft und ggf. neu abgegrenzt.

Die beiden Varianten machen deutlich, dass die für die Abgrenzung benötigte Datenbasis leicht an die Bedürfnisse der Organisation angepasst werden kann. Aufgrund der bereits vorhandenen Plattformen haben wir uns für die *Forced March*-Variante entschieden. Wegen der Bedeutung für das Management wurde zusätzlich eine Kosten-Nutzen-Analyse durchgeführt.

2.2 Understanding Relevant Domains

Für die Einführung von Produktlinien ist ein systematisches Bild der Produktlandschaft eine grundlegende Voraussetzung.

Um eine Vergleichbarkeit zu gewährleisten, wurden 28 repräsentative Produkte im Rahmen eines eintägigen Workshops analysiert. Gemeinsam mit dem Produktteam wurde das System in seine Teilfunktionen zerlegt, Sichten erarbeitet die nicht-funktionalen Anforderungen in Form eines Qualitätsbaums aufgenommen. Zusätzlich wurde gefragt, ob die jeweilige Funktion spezifisch für ein Produkt oder einen Kunden ist und wie ihr Wiederverwendungspotential bewertet wird. In der Analyse wurden vier unterschiedliche Stufen der Wiederverwendung identifiziert:

Direkt Funktion ist ohne Modifikation wiederverwendbar

Konfiguration Funktion ist durch Anpassung von Parametern oder Kennlinien wiederverwendbar

Struktur/Konzept Der Code ist nicht oder nur in geringem Maße wiederverwendbar. Jedoch gibt es für die Problemstellung eine definierte Vorgehensweise (z. B. ein "Design Pattern").

Keine Die Funktion kann nicht wiederverwendet werden, da sie z. B. speziell für einen Kunden angefertigt wurde oder von spezieller Hardware abhängig ist.

Nach der Analyse konnten durch Vergleich der jeweiligen Sichten leicht Kandidaten für wiederverwendbare Softwarefunktionen gefunden werden. Dabei wurde unterschieden, ob die Funktion nur innerhalb einer Produktplattform oder *plattform übergreifend* einsetzbar ist. Die folgende Liste benennt mögliche Kriterien für die Ermittlung von Funktionen mit hohem Wiederverwendungspotential:

- Funktion wird in vielen Produkten eingesetzt
- Funktion weist nur wenige Varianten auf
- Funktion ist als kommerzielle Software verfügbar (Eigenentwicklung in diesem Fall nicht sinnvoll)
- Funktion ist nicht kundenabhängig
- Funktion ist nicht produktabhängig¹
- Funktion hat einen nennenswerten Funktionsumfang (kein "Fünfzeiler")

Durch systematisches Prüfen der oben genannten Kriterien lassen sich leicht Kandidaten mit hohem Potential ermitteln. Ziel ist die Zusammenführung ähnlicher Funktionen zu konfigurierbaren Komponenten, die sich leicht in die zu erstellende Software-Architektur integrieren lassen.

¹ Innerhalb einer Produktplattform braucht dieses Kriterium nicht beachtet zu werden

2.3 Scoping

Unter Scoping versteht man die Abgrenzung der Produktlinie, d. h. welche Produkte sind Teil einer Produktlinie und welche nicht. Die Ableitung der Produktlinien ist in diesem Bereich nicht trivial, da sich Unterschiede und Gemeinsamkeiten nicht auf ein einziges Merkmal zurückführen lassen, sondern äußern sich auf mehreren Ebenen, z. B. Betriebssystem, Kommunikationsprotokoll oder Sicherheitsanforderungen.

Daher wurde ein eigenes statistisches Modell für die Identifizierung der Produktlinien herangezogen. Zunächst wurden die Produktmerkmale analysiert und daraus die signifikanten Produkteigenschaften herausgearbeitet. Merkmale wurden als "nicht signifikant" eingestuft, wenn sie z. B. redundant zu einem anderen Merkmal oder die Merkmale keine Unterschiede aufweisen.

Im Anschluss wurden die signifikanten Merkmale für jedes Produkt ermittelt und jeder Merkmalsausprägung ein eindeutiger Wert zugewiesen, z. B. erhielten Produkte ohne externe Kommunikation den Wert 0, bei Verwendung einer seriellen Punkt-zu-Punkt-Schnittstelle den Wert 1 und bei Comfort-CAN, d. h. einer Bus-Anbindung, den Wert 2. Im Anschluss wurde normiert, für jedes Produkt ein Merkmalsvektor gebildet und deren euklidische Abstand ermittelt.

Eine Produktgruppe erhält man, indem man einen maximalen Abstand vorgibt, zu einem "Ursprungsprodukt" alle Produkte mit einem geringeren Abstand sucht und diese zu einer Produktgruppe zusammenfasst. Durch Variation des Ursprungsprodukts und des maximalen Abstandes erhält man eine Häufigkeitsverteilung für verschiedene Produktgruppen. Die Aufteilung der Produkte auf die Gruppen ist dann optimal, wenn die Summe ihrer Häufigkeiten maximal ist und die Gruppen überschneidungsfrei sind.

Durch Anwendung des Algorithmus ergaben sich drei verschiedene Produktlinien mit 3-10 Produkten, wobei zwei Produkte aufgrund ihrer besonderen Merkmale nicht zu einer Produktlinie zugeordnet werden konnten.

2.4 Building a Business Case

Der *Business Case* beinhaltet eine Kosten-Nutzen-Analyse der Produktlinien und ist somit eine wichtige Entscheidungsgrundlage für das Management. Im vorliegenden Fall wurde auf die Methoden von [Pou97] zurückgegriffen. Das Modell von POULIN basiert u. a. auf den Faktoren *RCWR* (Relative Costs for Writing Reusable Components) und *RCR* (Relative Costs for Reusing a Component). Seine Untersuchung hat folgende Werte für die beiden Faktoren ergeben:

$$RCWR \cong 1,5 \text{ bzw. } RCR \cong 0,2. \quad (1)$$

Der Aufwand für die Entwicklung einer wiederverwendbaren Komponente ist demnach 50% höher gegenüber einer Komponente ohne Wiederverwendungsmöglichkeit und die Integration einer wiederverwendbaren Komponente ist gegenüber der Neuentwicklung um

80% günstiger.

Die Ersparnis bezieht sich natürlich nur auf die wiederverwendbaren Bestandteile, d. h. zugekaufte oder kundenspezifische Komponenten sind von dieser Betrachtung ausgeschlossen. Bei einer Produktentwicklungsdauer von 12 Monaten und einem Wiederverwendungsanteil von 60% erhält man folgende Werte für

$$Aufwand_{PL} = 12 \text{ M} \times 0,6 \times 1,5 = 10,8 \text{ M} \quad (2)$$

$$Aufwand_{Produkt} = 12 \text{ M} \times 0,6 \times 0,2 + 12 \text{ M} \times (1 - 0,6) = 6,48 \text{ M} \quad (3)$$

Dies zunächst eine Aufwand von 11 Monaten für die Entwicklung der Plattform, darauf aufbauende Produkte können dafür in knapp 7 anstatt 12 Monaten erstellt werden. Für die Gesamtrechnung wurden die einzelnen Produktlinien überlagert. Ergebnis war eine signifikante Kostenersparnis.

3 Methodische Anpassungen

3.1 Modellierung und Handhabung von Variabilität

Ein wesentlicher Erfolgsfaktor für erfolgreiches Produktlinien-Engineering ist die systematische Behandlung der Variabilität innerhalb einer Produktlinie. Das Problem besteht darin, die Unterschiede in den Anforderungen an die einzelnen Produkte in geeigneter Weise in die Implementierung abzubilden. Es muss also methodisch die Lücke zwischen der Variabilität in den Anforderungen und der Variabilität im Code geschlossen werden. In dem von uns im ITEA-Projekt CAFÉ („From Concept to Application in Product Family Engineering“) verfolgten Ansatz geschieht dies über die Produktlinien-Architektur [TH02].

Den Ausgangspunkt bildet dabei ein Merkmal-Modell zur Erfassung der Variabilität in den Anforderungen. Das Merkmal-Modell ist ein Ergebnis der Produktlinien-Anforderungsanalyse im Rahmen des Domänen-Engineerings. Es beschreibt die funktionalen und nicht-funktionalen Merkmale der Produktlinienmitglieder sowie ihre Gemeinsamkeiten und Variabilitäten.

Der erste Schritt im Entwurf ist die Festlegung einer Produktlinienarchitektur. Die in ihr erfasste Variabilität spiegelt Entwurfsentscheidungen wider, die in der Architekturphase noch nicht getroffen werden konnten. In unserem Ansatz werden sie mit Hilfe von Variationspunkten modelliert. Neben den möglichen Alternativen ist auch die Bindungszeit kennzeichnend für einen Variationspunkt.

Die wesentliche Aufgabe bei der systematischen Behandlung von Variabilität ist nun, die Verbindung zwischen der im Merkmal-Modell erfassten Variabilität in den Anforderungen und der mit den Variationspunkten modellierten Variabilität in der Architektur herzustellen und für die Produktkonfiguration auszunutzen. Dabei ist zu beachten, dass die Abbildung von Variationspunkten auf Merkmale nicht notwendigerweise injektiv ist: Die

Realisierung eines Merkmals kann durchaus mehrere Variationspunkte beeinflussen. Sie ist aber total definiert: Jedem Variationspunkt ist mindestens ein Merkmal zugeordnet, da die Variationspunkte keine neue Variabilität einführen, sondern nur die Variabilität des Merkmal-Modells realisieren sollen.

3.2 Management von Variabilität bei der Produktableitung

Der im vorigen Abschnitt skizzierte Ansatz erfordert eine methodische Unterstützung des Produktableitungsprozesses, da die Komplexität der Konfigurationsaufgabe in industriellen Anwendungen schnell so groß wird, dass ein rein intuitives Vorgehen nicht mehr zum Ziel führt. Eine entsprechende Methodik wird im IST-Projekt ConIPF („Configuration of Industrial Product Families“) entwickelt und erprobt [HM03]. Zur Zeit wird untersucht, wie der PLA mit formalen Konfigurationstechniken so kombiniert werden kann, dass eine für industrielle Anwendungen praktikable Produktentwicklungsmethode entsteht. Diese soll sowohl die reine Produktableitung als auch Mischformen von Ableitung und Individualentwicklung unterstützen.

Die Betrachtungen werden auf wissensbasierte, strukturorientierte Konfigurationstechniken konzentriert. Eine wesentliche Herausforderung ist die Integration dieser Techniken in den Produktentwicklungsprozess. Ziel ist die schon im vorigen Abschnitt beschriebene Verfolgbarkeit von der Merkmalkonfiguration bis zur Code-Anpassung, z.B. durch Parametrierung und Kalibrierung. Entsprechende Anforderungen an die Methodik wurden durch Interviews in zwei Organisationen gewonnen. Ihre Umsetzung wird in einem Experiment in der jeweiligen Organisation erprobt. Die Experimente werden zur Zeit vorbereitet.

3.3 Werkzeugunterstützung für szenariobasierte Architekturevaluation

Architekturevaluierung bietet die Möglichkeit, frühzeitig im Entwicklungsprozess Risiken zu identifizieren, die auf eine mangelnde Eignung eines Produkts oder einer Produktlinie hinsichtlich der Qualitäts- und Geschäftsanforderungen hinweisen. Wir haben gute Erfahrungen mit szenariobasierten Techniken, vor allem mit der „Architecture Trade-off Analysis Methode“ des SEI gemacht [FHL01].

Bei der Durchführung von Architekturevaluierungen in Form von Workshops werden zahlreiche Daten und Erkenntnisse aufgenommen, auf die bei der Analyse, der Ergebnispräsentation und in den entsprechenden Berichten Bezug genommen wird. In den bisherigen Durchführungen wurde diese Information unstrukturiert in Prosa aufgenommen. Als Folge war der Zugriff ineffizient: Die Suche nach einzelnen Daten und das Erstellen des Berichts mit entsprechenden Statistiken nahm viel Zeit in Anspruch, Daten wurden bei der Analyse nicht beachtet, und der aktuelle Bearbeitungsstand war nur mit Mühe nachvollziehbar. Dies ist bei dem verwendeten Workshop-Format, in dem schon am Ende der Veranstaltung die wesentlichen Ergebnisse präsentiert werden müssen, unbefriedigend.

Aus diesem Grund wurde ein Werkzeug entwickelt, das die aufgenommene Information strukturiert ablegt, Zugriffe erleichtert und Auswertungen automatisiert (z.B. durch Generierung von Berichten). Das Ziel war einerseits die Unterstützung des Evaluierungsteams während der Durchführung der Evaluierung. Andererseits sollte auch eine Möglichkeit geschaffen werden, Erfahrungen für weitere Architekturevaluierungen zu sichern. Struktur und Anwendung des Werkzeuges werden in [THE03] beschrieben.

4 Zusammenfassung und Ausblick

Der Beitrag berichtet über die Anwendung des Vorgehensmusters WtBP bei der Einführung von Software-Produktlinien und über grundlegende methodische Anpassungen des PLA.

Das WtBP liefert eine gute Orientierung für die Aktivitäten bei der Einführung von Softwareproduktlinien und ist leicht anpassbar. Durch die Analyse der Produkte wurde ein systematisches Bild der Produkte gewonnen und eine Grundlage für die Produktlinie geschaffen. Im Rahmen des Scopings wurden die Produkte zu Produktlinien zusammengefasst und deren Wirtschaftlichkeit nachgewiesen. Auf Basis dieser Ergebnisse hat die Entwicklung wiederverwendbarer Softwarekomponenten und gemeinsamer Softwarearchitekturen begonnen.

In den im zweiten Teil skizzierten Projekten konnten Anpassungen der PLA-Methodik erfolgreich umgesetzt werden. Die Notwendigkeit weiterer methodischer Ergänzungen ist zu erwarten und wird sich mit dem zunehmenden Einsatz des PLA ergeben.

Literatur

- [CN01] Paul Clements and Linda Northrop. *Software Product Lines – Practices and Patterns*. SEI Series in Software Engineering. Addison–Wesley, 2001.
- [FHL01] Stefan Ferber, Peter Heidl, and Peter Lutz. Reviewing Product Line Architectures: Experience Report of ATAM in an Automotive Context. In *Proc. 4th Int. Workshop on Product Family Engineering (PFE-4), Bilbao, Spain*, volume 2290 of *Lecture Notes in Computer Science*, pages 364–382. Springer, October 2001.
- [HM03] Andreas Hein and John MacGregor. Managing Variability with Configuration Techniques. In *Proc. Int. Workshop on Software Variability Management at ICSE'03, Portland, Oregon*, pages 19–23, May 2003.
- [Pou97] J. S. Poulin. *Measuring Software Reuse - Principles, Practices, and Economic Models*. Addison-Wesley, Reading, Ma., 1997.
- [TH02] Steffen Thiel and Andreas Hein. Modeling and Using Product Line Variability in Automotive Systems. *IEEE Software*, pages 66–72, July/August 2002.
- [THE03] Steffen Thiel, Andreas Hein, and Heiner Engelhardt. Tool Support for Scenario-based Architecture Evaluation. In *Proc. 2nd Int. Workshop on Software Requirements to Architectures (STRAW'03) at ICSE'03, Portland, Oregon*, pages 122–129, May 2003.