

Software Produktlinien

Einführung und Überblick

Johannes Diemke

Universität Oldenburg
Department für Informatik
26111 Oldenburg

Zusammenfassung Dieser Artikel soll einen Überblick über den Software Produktlinien Ansatz verschaffen. Zu diesem Zweck werden die grundlegenden Konzepte, Vorteile und Herausforderungen von Software Produktlinien beschrieben. Dazu wird der Begriff der systematischen Wiederverwendung eingeführt. Darauf aufbauend folgt die gängige Definition von Software Produktlinien. Es wird auf die drei zentralen Prozesse Domain Engineering, Application Engineering und Management eingegangen. Des Weiteren werden Begriffe wie Produktraum, Systemartefakte und Variabilität im Zusammenhang mit Produktlinien erläutert. Schließlich wird die Produktlinienarchitektur betrachtet und der Software Produktlinien Ansatz bewertet.

1 Einführung

Die steigende Komplexität und Größe zu entwickelnder Softwaresysteme ist ein gegenwärtiges Problem der Software-Technik. Softwaresysteme sollen immer leistungsfähiger, zuverlässiger, komplexer, gleichzeitig aber immer günstiger in Produktion und Wartung sowie schneller in Entwicklung und Auslieferung werden. Die systematische Wiederverwendung von Software ermöglicht es diese scheinbar unvereinbaren Ziele zu erreichen. Insbesondere hat sich gezeigt, dass gerade die Entwicklung von Software Produktlinien einen der leistungsfähigeren Ansätze der systematischen Wiederverwendung darstellt [8].

Die Realität sieht jedoch anders aus. Der gängige Ansatz zur Erstellung von Softwareprodukten in Unternehmen beruht meist darauf, für jedes durchzuführende Projekt ein eigenes Projektteam mit eigenem Budget zusammenzustellen, welches isoliert von den anderen Teams die Entwicklung beginnt. Dadurch kommt es jedoch nicht selten zu mehrfacher und redundanter Durchführung verschiedener Arbeiten der unterschiedlichen Projektteams. Genau dieser Problematik versucht der Software Produktlinien Ansatz entgegenzuwirken [9].

Dieser Artikel beschreibt die grundlegenden Konzepte, Vorteile und Herausforderungen von Software Produktlinien. Da der Software Produktlinien Ansatz auf dem Konzept der systematischen Wiederverwendung beruht, wird in Abschnitt 2 dieses zunächst eingeführt und auf dessen Nutzen eingegangen. Anschließend folgt in Abschnitt 3 nach einer einführenden Motivation die gängige

Definition von Software Produktlinien. Der Abschnitt 3.1 behandelt die Systemartefakte, welche die Grundbausteine der Software Produktlinie darstellen. Ein kurzer Überblick über den Produktraum und dessen Beschreibung wird in Abschnitt 3.2 gegeben. Abschnitt 3.3 beschreibt die Rolle der Variabilität in Software Produktlinien, Möglichkeiten Variabilität mit Hilfe von Produkt×Feature-Tabellen und Feature-Graphen zu beschreiben und führt den Begriff des Variationspunkts ein. Daraufhin wird in Abschnitt 3.4 der grundlegende Unterschied zwischen Softwarearchitekturen von Individualsoftware und Produktlinienarchitekturen der sich zum großen Teil daraus ergibt, dass eine Produktlinienarchitektur die Referenz für alle Produkte der Produktlinie darstellt und gewisse Variabilitätsaspekte berücksichtigt werden müssen, beschrieben. Schließlich wird in Abschnitt 3.5 auf die drei zentralen Prozesse Domain Engineering, Application Engineering und Management eingegangen. Ein Fazit und eine abschließende Bewertung des Software Produktlinien Ansatzes folgt in Abschnitt 4.

2 Systematische Wiederverwendung

Die Wiederverwendung ist ein schon lange verfolgter Ansatz, dessen Nutzen relativ früh auch für die Software-Technik erkannt wurde. Bei diesem Ansatz werden, wie in den meisten anderen technischen Disziplinen, im Entwurfsprozess vorgefertigte Komponenten, die schon in anderen Systemen getestet wurden, genutzt. Vorteile der Wiederverwendung sind geringere Entwicklungskosten, eine höhere Zuverlässigkeit und eine beschleunigte Entwicklung. Um eine systematische Wiederverwendung zu erreichen, muss diese allerdings frühzeitig in den Entwurfsprozess mit einbezogen werden und richtig geplant sein [8].

Zu diesem Zweck wurden im Laufe der Zeit eine Reihe von Konzepten entwickelt, welche von der Wiederverwendung einzelner Funktionen bis hin zu ganzen Anwendungssystemen reichen. Das Konzept der Entwurfsmuster bietet beispielsweise domänenunabhängige und generische Lösungsschemata für wiederkehrende Entwurfsprobleme. Mit den Referenzarchitekturen werden domänenabhängige und generische Softwarearchitekturen geboten. Die in den späten 90er Jahren aufkommende komponentenbasierte Software-Technik konzentriert sich schließlich darauf, Softwaresysteme aus abstrakten, lose gekoppelten und wiederverwendbaren Komponenten zu entwerfen [6].

3 Software Produktlinien

Einen äußerst leistungsfähigen Ansatz zur systematischen Wiederverwendung stellen die Software Produktlinien, welche gelegentlich auch als Anwendungsfamilien bezeichnet werden, dar. Diesem Ansatz liegt zugrunde, dass sich Softwarehersteller häufig auf spezielle Anwendungsdomänen spezialisieren und für diese eine Reihe von Produktvarianten entwickeln. Produktvarianten haben aber naturgemäß eine gemeinsame Grundstruktur und eine Vielzahl ähnlicher Eigenschaften. Dies zeichnet jedoch eine Produktfamilie aus. Ziel des Software Produktlinien Ansatzes ist es nun das vorhandene Wiederverwendungspotential ei-

ner Produktfamilie auszuschöpfen, indem die dazu nötige Infrastruktur geschaffen wird [6].

Da in einer Software Produktlinie jedes Produkt auch immer eine Produktvariante darstellt, sind die Begriffe Produkt und Produktvariante im Kontext einer Produktlinie als synonym anzusehen. Im Folgenden wird eine Definition von Software Produktlinien in Anlehnung an [7] gegeben. Dabei wird der bisher allgemein gehaltene Begriff der Produktvariante weiter auf Produktvarianten von softwareintensiven Systemen eingegrenzt.

Definition 1. *Eine Software Produktlinie (SPL) ist eine Menge von softwareintensiven Systemen, die sich eine Reihe von Systemartefakten teilen, einer speziellen Anwendungsdomäne angehören und eine gemeinsame systemspezifische Architektur besitzen. Jedes dieser Systeme ist auf die eine oder andere Art spezialisiert, der gemeinsame Kern der Systeme wird jedoch jedesmal wiederverwendet.*

Der Software Produktlinien Ansatz ermöglicht so für eine spezielle Anwendungsdomäne die Erstellung einer Reihe von softwareintensiven Systemen auf Basis einer Menge gemeinsam genutzter Systemartefakte, eines gemeinsamen Produktionsprozesses und einer konsequenten Aufbau- und Ablauforganisation. Die systematische Wiederverwendung der Systemartefakte wird dabei von Anfang an in den Entwurfsprozess mit einbezogen. Die gemeinsam verwendeten Systemartefakte beschränken sich dabei nicht auf wiederverwendbare Softwarekomponenten wie Abschnitt 3.1 zeigt.

Der Software Produktlinien Ansatz kann durch eine Analogie zu traditionellen Industriebereichen wie der Autoindustrie veranschaulicht werden. So streben Autohersteller die Erstellung einer Menge von Kraftfahrzeugen an und versuchen dabei die Unterschiede der in den Fahrzeugen verwendeten Bauteile möglichst gering zu halten. Dies ermöglicht ihnen eine breite Wiederverwendung von Komponenten. In dieser Analogie ist dann die Anwendungsdomäne die Erstellung von Kraftfahrzeugen in der Autoindustrie, die zu erstellenden Produktvarianten sind dann die im Produktportfolio vorhandenen Kraftfahrzeuge und die gemeinsam genutzten Systemartefakte, die in den Fahrzeugen teilweise gemeinsam genutzten Bauteile [3].

Da sich der Software Produktlinien Ansatz jedoch nicht für jede Menge von Softwareprodukten eignet, ist vor der Entwicklung einer Produktlinie in einer Machbarkeits- und Rentabilitätsanalyse gründlich zu prüfen, ob sich diese langfristig finanziell rentiert und genügend ähnliche Eigenschaften in der Produktmenge vorhanden sind, die sich sinnvoll zu Systemartefakten abstrahieren und in die Produktlinie integrieren lassen [6].

Im Folgenden werden einige für Softwareproduktlinien grundlegende Begriffe eingeführt und anschließend in Zusammenhang gebracht.

3.1 Systemartefakte

Konkrete Produktvarianten softwareintensiver Systeme werden im Produktlinien Ansatz durch das Auswählen und Anpassen von Systemartefakten erzeugt. Sys-

Systemartefakte sind daher die Grundbausteine der Produktlinie zur Erstellung der Softwareprodukte. Eine besondere Bedeutung kommt dabei den Softwarearchitekturen (siehe Abschnitt 3.4) zu, die spezielle Systemartefakte darstellen, welche erst eine Wiederverwendung und Organisation der anderen Systemartefakte im Großen ermöglichen. Weitere Systemartefakte sind die gemeinsam genutzten Features der Software Produktlinie. Dabei handelt es sich meistens um wiederverwendbare Softwarekomponenten, Subsysteme, Module, Frameworks oder Plattformdienste [9]. Hier hat sich gezeigt, dass die komponentenbasierte Software Entwicklung die Realisierung von Software Produktlinien vereinfacht [6]. Mindestens genauso wichtig wie die bisher genannten Artefakte sind jedoch Systemartefakte in Form von Dokumenten, die Erfahrungen und das angesammelte strategische Wissen der Anwendungsdomäne widerspiegeln. Dazu gehören Geschäftsmodelle, Anforderungsspezifikationen, Projektpläne, Budgets, Testfälle, domänenspezifische Muster sowie Prozesse und Richtlinien [7].

Jedes erstellte Systemartefakt ist ein Teil der Software Produktlinie und wird in einem Artefaktkatalog inventarisiert. Die Systemartefakte, mit den für eine Produktvariante produktspezifischen Entscheidungen, dienen, wie in Abbildung 1 dargestellt, im Produktions-Prozess als Basis zur Erstellung neuer Produktvarianten. Dabei sind die Systemartefakte und die produktspezifischen Entscheidungen die Eingaben des Produktions-Prozesses und die konkreten Produktvarianten die Ausgaben.



Abbildung 1. Systemartefakte als Basis zur Erstellung neuer Produktvarianten [9].

3.2 Produktraum

Der Produktraum definiert den Umfang einer Software Produktlinie. Dazu werden die zu der Software Produktlinie gehörenden Produktvarianten aufgelistet und beschrieben. Er wird, wie Abschnitt 3.5 noch zeigen wird, in der Scopingphase des Domain Engineering Prozess entwickelt.

Üblicherweise werden detailliert die Anforderungen und Unterschiede zwischen den einzelnen Produktvarianten einer Software Produktlinie sowie funktionale und nichtfunktionale Anforderungen, die diese Produkte an die Software Produktlinie stellen, dokumentiert. Produkt×Feature-Tabellen helfen dabei

Gemeinsamkeiten zwischen den Softwareprodukten der Produktlinie zu identifizieren. Featuregraphen werden dazu genutzt um Abhängigkeiten zwischen verschiedenen Features darzustellen [6]. Im folgenden Abschnitt 3.3 wird noch auf die Verwendung von Featuregraphen anhand eines Beispiels eingegangen. Dazu muss jedoch zunächst der Begriff des Features im Zusammenhang mit der Variabilität in Software Produktlinien eingeführt werden.

3.3 Variabilität

Die Produktvarianten einer Software Produktlinie zeichnen sich durch eine Vielzahl gemeinsamer, unterschiedlicher und variierender Features aus. Aus diesem Grund spielen bei der Entwicklung von Software Produktlinien insbesondere Variabilitätsaspekte eine übergeordnete Rolle. Unter Variabilität wird im Allgemeinen die Möglichkeit verstanden, Systeme oder Komponenten zu ändern und an individuelle Bedürfnisse anzupassen. Bei Produktlinien bezeichnet sie die Unterschiede der Produktvarianten und definiert den Rahmen in dem diese durch Selektion von Systemartefakten individuell angepasst werden können. Sie spielt damit eine zentrale Rolle in der Produktlinienentwicklung und tritt in unterschiedlichster Form auf. Dies kann von der Unterstützung mehrerer Betriebssysteme bis zur komplexen Anpassung von Systemartefakten reichen [6].

Im Folgenden wird auf die Beschreibung der Variabilität des Produktraums eingegangen und das Konzept des Variationspunkts eingeführt.

Produktraum Variabilität Die Variabilität des Produktraums wird mit der Hilfe von Feature×Produkt-Tabellen und Feature-Graphen dokumentiert. Features beschreiben dabei Merkmale der Software Produktlinie, die Gemeinsamkeiten und Unterschiede der einzelnen Produktvarianten darstellen. Features lassen sich dabei weiter klassifizieren in externe Features, notwendige Features und optionale Features. Externe Features zeichnen sich dadurch aus, dass sie nicht Teil des Produkts selbst sind, aber von diesem benötigt werden. Notwendige Features sind grundlegende Features, ohne die das Produkt nicht funktionsfähig wäre. Optionale Features bieten zusätzliche Funktionalität, die zum Betrieb des Produkts nicht dringend notwendig ist. Feature×Produkt-Tabellen geben dabei Aufschluss über den Umfang der geplanten Produkte und deren Features. Feature-Graphen dokumentieren Abhängigkeiten zwischen Features und deren Variabilität [6].

Abbildung 2 beschreibt den Produktraum einer Software Produktlinie zur Erstellung mehrerer Varianten eines Mail-Clients mit der Hilfe eines Feature-Graphen. Dazu wird eine zu der UML sehr ähnliche Notation verwendet, welche Konstrukte zur Auszeichnung von Kompositionen von Features, Optionalen Features, OR- und XOR-Spezialisierungen von Features und externe Features enthält. In der Abbildung ist zu erkennen, dass ein Mail-Client eine Komposition aus den Features TCP Connection, Type Message, Receive Message, Send Message und Runtime Platform darstellt. Weiterhin ist ihr zu entnehmen, dass das Receive Message Feature, welches zum Empfangen von E-Mails dient, über einer OR-Spezialisierung mit den Features Pop3 und IMAP verbunden ist. Dies

bedeutet, dass das Receive Message Feature das Pop3 oder aber das IMAP Feature zum Empfangen von E-Mails verwenden kann. Die Auszeichnung „runtime“ weist darauf hin, dass die Wahl des tatsächlich zu nutzenden Features zur Laufzeit entschieden wird. Der Abbildung ist außerdem zu entnehmen, dass die Runtime Platform über eine XOR-Spezialisierung mit den Features win32 und linux in Beziehung steht. Da die Beziehung weiterhin mit „compiletime“ ausgezeichnet ist, bedeutet dies, dass ein Mail Client wahlweise Windows oder aber ein Linux-Derivat nutzen kann, dies aber zum Zeitpunkt der Übersetzung entschieden werden muß. Eine detaillierte Beschreibung zur Erstellung von Feature-Graphen wird in [5] und [4] gegeben. Orte im Feature-Graphen, an denen eine Entscheidung bezüglich des zu nutzenden Features getroffen werden muss, werden auch Variationspunkte genannt und im folgenden Abschnitt näher erläutert.

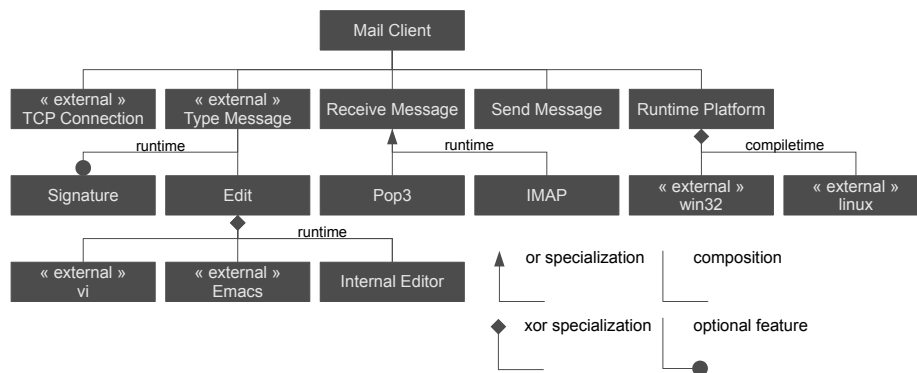


Abbildung 2. Beschreibung eines Produktraums zur Erstellung eines Mail-Clients mit Hilfe eines Feature-Graphen [4].

Variationspunkte Produkt- und systemumgebungsspezifische Variabilität wird durch so genannte Variationspunkte abgebildet. So werden variable und optionale Features erst durch Variationspunkte möglich. Variationspunkte sind also Punkte im Entwicklungsablauf, an denen Entwurfsentscheidungen bezüglich der Variabilität getroffen werden müssen, um eine konkrete Variante eines Features zu erhalten. Es muss also eine aus mehreren Entwurfsalternativen gewählt werden. Wird eine aus mehreren Entwurfsalternativen gewählt so wird dies auch als das Binden von Variationspunkten bezeichnet. Das Binden von Variationspunkten kann dabei zu unterschiedlichen Zeitpunkten stattfinden. Beispielsweise kann das Binden während des Architekturentwurfs, dem Feinentwurf, der Implementierung, der Übersetzung oder zur Laufzeit stattfinden [6].

3.4 Produktlinien Architektur

Eine explizit definierte Produktlinienarchitektur ist von zentraler Bedeutung für die gesamte Produktlinieninfrastruktur. Sie stellt eine gemeinsame generische Referenzarchitektur für alle im Produktraum liegenden Softwareprodukte zur Verfügung.

Konkrete Softwareprodukte leiten ihre Architektur von der Produktlinienarchitektur ab. Durch diese für die gesamte Produktlinie gültige Architektur kann für alle Produkte die Erfüllung von nichtfunktionalen Anforderungen, wie Performance, Verfügbarkeit, Skalierbarkeit und Erweiterbarkeit leichter sichergestellt werden, da diese Aspekte zu großen Teilen durch die Architektur abgedeckt werden [9]. Eine Besonderheit bei Referenzarchitekturen ist, dass produktspezifische Anforderungen schon im Entwurf der Produktlinienarchitektur beachtet werden müssen, um eine möglichst generische Architektur, die für alle Produkte nutzbar ist, zu schaffen. Wird dies nicht beachtet, so kann die Referenzarchitektur im schlimmsten Fall produktspezifische Anforderungen unmöglich machen. Ziel ist es aber eine Produktlinienarchitektur zu entwickeln, welche den Anforderungen der Softwareprodukte der gesamten Produktlinie genügt. Des Weiteren wird durch sie auch die Kommunikation zwischen den verschiedenen Interessengruppen unterstützt [3].

Eine Produktlinienarchitektur bietet verschiedene Variationsmöglichkeiten hinsichtlich ihrer Konnektoren und Komponenten. Diese können in verschiedenen Varianten angeboten werden. In der Architektur sollte explizit definiert sein, welche Komponenten obligatorisch, optional und variabel sind und wie optionale und variable Komponenten durch konkrete Komponenten instanziiert werden können. Dabei bezeichnet eine Konfiguration die konkrete Auswahl von Systemartefakten die ein Softwareprodukt formen. Im Idealfall ist es für konkrete Softwareprodukte möglich, direkt die Referenzarchitektur der Software Produktlinie zu nutzen, indem Variationspunkte durch Selektion konkreter Komponenten in der Architektur gebunden werden. Dazu ist es nötig standardisierte Schnittstellen zu schaffen. Im konkreten Fall können Schnittstellen in der objektorientierten Programmierung durch Klassen implementiert werden. Variation kann dann durch unterschiedliche Konfigurationen und das Ableiten von Klassen erfolgen.

Bereiche der Referenzarchitektur, die nicht geändert werden, nutzen gemeinsame Komponenten. Es kann jedoch auch vorkommen, dass neben der Auswahl konkreter Komponenten an den Variationspunkten die Architektur selbst teilweise an die produktspezifischen Anforderungen angepasst werden muss [2].

3.5 Produktlinien Prozesse

Bei der Entwicklung von Software Produktlinien wird zwischen drei zentralen und iterativen Prozessen, dem Domain Engineering, dem Application Engineering und dem Management unterschieden [9]. Auf diese drei Prozesse wird im Folgenden genauer eingegangen.

Domain Engineering Das Domain Engineering stellt den zentralen Teilprozess des Produktlinien Engineering dar. In ihm wird der Produktraum, die benötigte Infrastruktur und gemeinsame Funktionalität in Form wiederverwendbarer Systemartefakte für die zu erstellende Software Produktlinie einer spezifischen Domäne konzipiert und entwickelt. Das Domain Engineering lässt sich in die drei Teilprozesse Domänenanalyse & Scoping, Architekturentwurf und Implementierung unterteilen. Die Domänenanalyse umfasst die Anforderungsanalyse für die gesamte Produktlinie und dokumentiert die Gemeinsamkeiten und Unterschiede aller geplanten Produktvarianten. In der Scopingphase werden alle Informationen zu den geplanten Produktvarianten gesammelt, beschrieben und in dem Produktraum festgehalten. Aus dem Wissen der Domänenanalyse und dem Scoping wird in der Architekturentwurfsphase eine Produktlinienarchitektur entworfen, aus der die Architekturen der konkreten Produktvarianten abgeleitet werden. In der Implementierungsphase werden die Systemartefakte, die auch als Core Assets bezeichnet werden, konzipiert, entwickelt und anschließend in der Produktlinien-Infrastruktur archiviert. Zusätzlich wird in dieser Phase in einem Produktionsplan festgehalten, wie die einzelnen Produktvarianten aus den Systemartefakten zu konstruieren sind [6]. Ziel des Domain Engineering ist es, dem Application Engineering Prozess eine technische und organisatorische Plattform zur Verfügung zu stellen, welche die Erstellung von Produktvarianten der Anwendungsdomäne durch Verwendung systematisch wiederverwendbarer Systemartefakte unterstützt. Dazu werden die einzelnen Systemartefakte der Plattform genau auf die Produktlinie zugeschnitten und für eine hohe Wiederverwendbarkeit ausgelegt, so dass sie sich mit geringem Aufwand miteinander kombinieren lassen [3].

Application Engineering Das Application Engineering bezeichnet den Teilprozess des Produktlinien Engineering, in dem einzelne Instanzen der Software Produktlinie, also konkrete Produktvarianten, durch das Auswählen und Anpassen der von der Plattform zur Verfügung gestellten Systemartefakte und dem Implementieren systemspezifischer Komponenten, erzeugt werden [6]. Im Idealfall entstehen neue Produktvarianten der Plattform durch das Zusammenbauen nach einem Produktionsplan. Der Schwerpunkt wird so vom Programmieren zum Integrieren verlagert [3]. Für jede Produktvariante werden die Phasen Systemanalyse, Systementwurf, Systemimplementierung und Systemtest durchlaufen. In der Systemanalyse werden die softwaresystemspezifischen Anforderungen, die sich von denen der Produktlinie unterscheiden, ermittelt. Im Systementwurf wird die konkrete Systemarchitektur der Produktvariante durch Ableitung von der Produktlinienarchitektur entwickelt. Bei der Systemimplementierung wird schließlich durch Auswählen der Systemartefakte die konkrete Produktvariante erstellt. Die Anforderungen eines konkreten Softwaresystems können wiederum dazu führen, dass der Domain Engineering Prozess angestoßen wird um die Plattform der Software Produktlinie entsprechend anzupassen und gegebenenfalls um weitere Systemartefakte zu erweitern [7].

Management Das Management ist der zentrale Teilprozess des Produktlinien Engineering, der die beiden anderen Prozesse unterstützt und koordiniert. Hier wird zwischen technischem und organisatorischem Management unterschieden. Das technische Management überwacht die Entwicklung der Systemartefakte und der konkreten Produktvarianten. Es stellt sicher, dass alle Prozesse gemäß den Vorgaben durchgeführt werden. Dagegen ist das organisatorische Management für die Planung, Priorisierung und Verteilung der Ressourcen zuständig. Es legt weiterhin die grundlegende Unternehmensstrategie fest. Ein gut geführtes Management ist von entscheidender Bedeutung für die erfolgreiche Umsetzung einer Software Produktlinie [7, 9].

Abbildung 3 zeigt die drei zentralen Prozesse und ihre Interaktion bei der Entwicklung von Software Produktlinien.

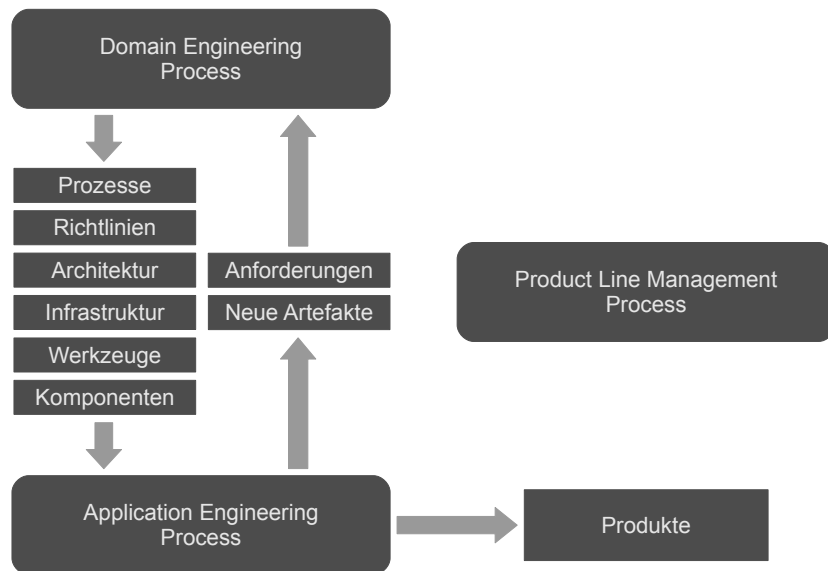


Abbildung 3. Die drei zentralen Prozesse Domain Engineering, Application Engineering und Management bei der Entwicklung von Software Produktlinien [9].

4 Fazit

Die Einführung von Software Produktlinien ist ein vielversprechender Ansatz zur Reduktion der Entwicklungskosten, Erhöhung der Produktzuverlässigkeit und Produktqualität sowie einer beschleunigten Entwicklung [6]. Die Komplexität der Einführung, des Aufbaus und des Betriebs einer Software Produktlinie in einem Unternehmen stellt aber eine große Hürde für die Ausschöpfung des vorhandenen Wiederverwendungspotentials einer Produktfamilie dar. Es ist viel Zeit und

eine hohe Vorabinvestition nötig, um eine Produktlinie erfolgreich einzuführen. Während der Aufbauphase einer Produktlinie wirft diese nämlich zunächst keine Erträge ab, sondern führt sogar zu höheren Kosten. Aus den besagten Gründen scheitern viele Unternehmen bei dem Versuch eine Produktlinie einzuführen, selbst wenn die Anwendungsdomäne und die Produktvarianten prädestiniert für die Entwicklung einer Produktlinie sind. Ein Grund ist meistens das Fehlen eines klar definierten Vorgehensmodells für die Entwicklung einer Software Produktlinie. Benötigt wird also eine ganzheitliche Methode, die ein klar definiertes Vorgehensmodell zur Entwicklung einer Produktlinie bereitstellt, denn der Software Produktlinien Ansatz ist zunächst nur ein Konzept. Eine vielversprechende ganzheitliche Methode zum erfolgreichen Aufbau, Betrieb, Einsatz und Wartung einer Software Produktlinie definiert beispielsweise das vom Fraunhofer Institut entwickelte PuLSE (Product Line Software Engineering) [1]. Ist eine Produktfamilie jedoch erst einmal etabliert, treten die genannten Nachteile immer mehr in den Hintergrund und die Vorteile, wie höhere Softwarequalität, höhere Produktivität und kürzere Entwicklungszeiten kommen zum Vorschein [2].

Literatur

- [1] M. Anastasopoulos, C. Atkinson, and D. Muthig. A Concrete Method for Developing and Applying Product Line Architectures.
- [2] K. Böllert. Objektorientierte Entwicklung von Software-Produktlinien zur Serienfertigung von Software-Systemen, 2002. URL <http://www.db-thueringen.de/servlets/DocumentServlet?id=1158>.
- [3] P. Clements and L. Northrop. *Software Product Lines: Practices and Patterns (The SEI Series in Software Engineering)*. Addison-Wesley Professional, 2001. ISBN 0201703327.
- [4] J. V. Gurp, J. Bosch, and M. Svahnberg. On the notion of variability in software product lines. In *WICSA '01: Proceedings of the Working IEEE/I-FIP Conference on Software Architecture (WICSA '01)*, page 45, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7695-1360-3. doi: <http://dx.doi.org/10.1109/WICSA.2001.948406>.
- [5] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical report, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, 1990.
- [6] R. Reussner and W. Hasselbring. *Handbuch der Software-Architektur*. Dpunkt Verlag, 2006. ISBN 3898643727.
- [7] Software Engineering Institute. Product Lines, 2007. URL <http://www.sei.cmu.edu/productlines/>. Stand: 31.12.2007.
- [8] I. Sommerville. *Software Engineering*. Pearson Studium, 2007. ISBN 3827372577.
- [9] R. Zacharias. Produktlinien: Der nächste Schritt in Richtung Software-Industrialisierung. *Javamagazin*, (3.07):69–79, 2007.