Extra: 20 Seiten JAX-Programm! Alle Infos S. 49–68!

3.07

Deutschland €6,50 Niederlande €7,80 Österreich €7,50 Schweiz sFr 12,70

S&S

Javamagazin

Java magazin

Internet & Enterprise Technology

javamagazin.de

Auf CD

Alle Infos auf S.3

Testversionen & more

- Sparx Systems Enterprise Architect 6.1
- Infragistics NetAdvantage for JSF 2006 Vol. 2
- AndroMDA 3.2
- MySQL 5.0.27
- Hibernate 3.2.1
- Spring Framework 2.0

JAX Radio

Mit den W-JAX-Sessions (Audio)

- What is the "shape" of SOA? (Mark Hapner)
- Drei unterschiedliche Sichtweisen auf Hibernate (Werner Eberling)

Die Industrialisierung der Software-Produktion

→ Vom Unikat zur Produktlinien-Fertigung

Persistenz Light

Alternativen zu EJB, Hibernate und JDO

JRuby: Ruby für die Java-Welt

MDA: Modellgetriebene Entwicklung mit AndroMDA

Advanced Spring: Neue Reihe für die Spring-Profis

Enterprise ARCHITEKTUR MAGAZIN

→ WebSphere Integration Developer: Assemblierung von Geschäftslösungen

→ Gedeihen agile Teams nur in agilen Unternehmen?

Datenträger enthält nur Lehr- oder Infoprogramme

D 45867

4 194586 706503 03

Die Industrialisierung der Softwareproduktion

Liebe Leserinnen und Leser!

Die Industrialisierung der Softwareproduktion – manchem Coder alter Schule wird's da mulmig im Bauch. Wohin wollen die mit ihrer Industrialisierung, warum sprechen plötzlich alle von Factorys oder Produktlinien? Mit dem Stichwort „Industrialisierung" ist nichts anderes gemeint, als dass wir an der Schwelle zu einem neuen Zeitalter der Softwareentwicklung stehen. Starker Tobak, der uns da von der Industrie eingeschenkt wird ...

Darum ist es Zeit, ein paar Grundbegriffe in diesem Zusammenhang einer Prüfung zu unterziehen und herauszuarbeiten, was sich hinter den Schlagworten an Konkretem verbirgt. Kaum einer, der nicht schon von dem einfachen Pattern, das den Prozess der Industrialisierung beschreibt, gehört hätte: Zuerst war die Kunst, dann wurde aus ihr ein Handwerk und schließlich wurde die handwerkliche Herstellung in industrielle Produktion überführt. Dabei, so heißt es, stünden die meisten Softwareentwicklungsprojekte heute noch auf der Stufe des Handwerks und zeigten nur zaghafte Ansätze, sich den zeitgemäßen industriellen Verfahren zuzuwenden.

Ob dies zutrifft, sei dahingestellt. Roger Zacharias beschreibt jedenfalls in seinem Beitrag ab Seite 69, wie sich die wichtigsten Faktoren – Standardisierung, Wiederverwendung von Artefakten, Automatisierung, Verringerung der Fertigungstiefe – auf die Softwareproduktion auswirken (können).

Aber benötigen wir diesen Begriff der Industrialisierung überhaupt in unseren realen Projekten oder handelt es sich nicht vielmehr wieder um die nächste Sau, die von der Industrie durchs Dorf getrieben wird? Ist es nicht besser, wenn ich in guter handwerklicher Tradition meinen individuellen Kunden ihren ebenso individuellen Bedürfnissen gemäß perfekt sitzende Maßanzüge fertige, als ihnen Ware von der Stange anzubieten?

Man muss kein Industrie-Historiker sein, um sich vorstellen zu können, dass die Anzahl der traditionellen Schneidereien in den letzten einhundert Jahren stark abgenommen hat (um im Bild zu bleiben). Stattdessen gibt es heute eine Vielzahl an Marken und Produkten und es ist höchst unwahrscheinlich, dass sich im Laufe von ein paar CeBIT-Tagen zwei Männer im exakt gleichen Anzug begegnen – Diversifizierung statt individueller Maßanfertigung.

Innovation matters

Keine Industrie kann jedenfalls auf Dauer bestehen, wenn sie nicht permanente Veränderung und Innovation hervorbringt. Es sind die großen und kleinen Ideen, Visionen und Erfindungen, von denen die wichtigsten Denkanstöße ausgehen und dafür sorgen, dass sich die Räder im Getriebe weiter drehen. Diesen Impulsgebern haben wir auch in diesem Jahr den JAX Innovation Award (www.jax-award.com) gewidmet. 20.000 Euro loben wir aus, um die wichtigsten Softwareprojekte, Ideen oder Initiativen im Umfeld von Java, Eclipse und SOA zu prämieren.

Sie glauben, dass Sie einen interessanten Vorschlag haben? Dann nichts wie los – jeder kann etwas einreichen! Zeit ist noch bis zum 19. März. Und vielleicht gehören Sie oder Ihr Unternehmen ja zu den Siegern, wenn auf der JAX (23. bis 27. April) die feierliche Verleihung dieses europäischen Preises stattfindet.

In diesem Sinne – viel Spaß bei der Lektüre!

Sebastian Meyen
Chefredakteur

Produktlinien: Der nächste Schritt in Richtung Software-Industrialisierung

Industrialisierung, wir kommen!

VON ROGER ZACHARIAS

Erstellt und vermarktet ein Unternehmen mehrere Softwareprodukte, stellt sich die zugehörige Aufbau- und Ablauforganisation oft folgendermaßen dar: Für jedes zu erstellende Produkt wird ein eigenes Projektteam mit eigenem Budget aufgestellt. Dieses Team erstellt nun unter Hochdruck sein Produkt, ohne nähere Kenntnis der anderen Produkte und ohne Verwendung bereits existierender Assets. Verschiedene Arbeiten werden dabei von jedem Team redundant und aufs Neue durchgeführt. Dieser auf Unikate ausgelegte Entwicklungsansatz verursacht enorme Kosten. Doch wie bekommt man diese durch den Einsatz von Softwareproduktlinien und damit Industrialisierung der Softwareentwicklung in den Griff? Welche Vorteile entstehen daraus und vor welchen Herausforderungen steht man dabei?

Bei entsprechend hohem Bedarf durch den Konsumenten entwickelt sich jeder neue Industriezweig über verschiedene Stufen vom Künstlertum über das Handwerk bis hin zur Industrieproduktion (Abb. 1), was in der Regel als „Industrialisierung" bezeichnet wird. Das Künstlertum zeichnet sich durch einen geringen Bedarf hinsichtlich der Stückzahl der hergestellten Produkte aus, hierbei geht es also meist um Unikate mit vergleichbar hohen Stückkosten (Beispiele: Designermöbel, Kunstschmuck). Das andere Extrem ist die vollkommen automatisierte und standardisierte Industrieproduktion mit sehr hohen Stückzahlen und geringen Stückkosten (Beispiele: Kopfschmerztabletten, CD-Rohlinge). Jede Industrialisierung in diesem Sinne zeichnet sich durch Steigerung folgender fünf Kernfaktoren aus: 1. Standardisierung/Kommoditisierung, 2. Wiederverwendung, 3. Arbeitsteilung, 4. Automatisierung und 5. Verringerung der Fertigungstiefe.

Industrialisierung in der IT

Wie steht es nun um die Industrialisierung der Softwareentwicklung bzgl. dieser Kernfaktoren? Die in der Einleitung beschriebene Problematik zeigt, dass sich die Softwareentwicklung in vielen Unternehmen insbesondere im Vergleich zu Industrien wie der Automobilindustrie oder der Chemie/Pharmazie noch in einem frühen Reifestadium, also auf Ebene des Handwerks befindet.

Interessant ist, dass ein erheblicher Teil der Software einzig und allein dafür herge-

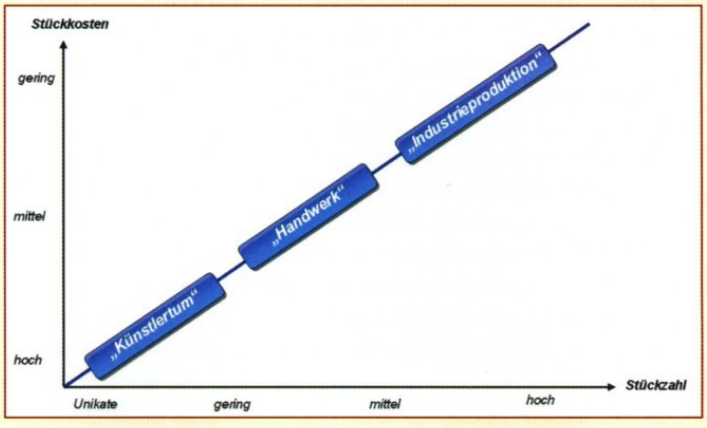

Abb. 1: Industrie-Entwicklungsstufen

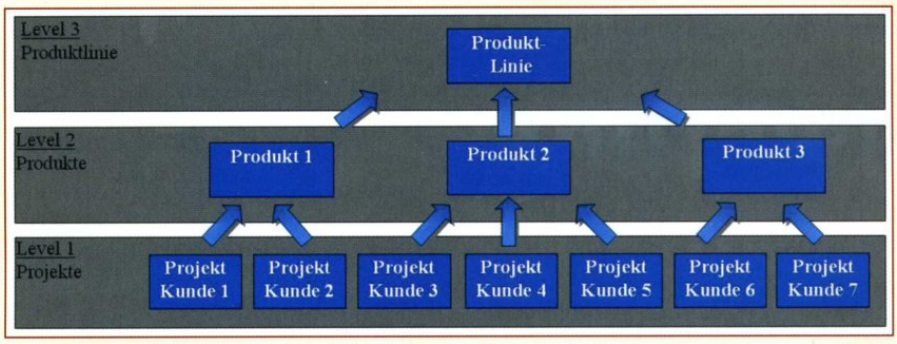

Abb. 2: Abstraktionsebenen der Entwicklung

stellt wird, um vorher manuelle Arbeitsschritte oder Prozesse zu automatisieren und damit zu optimieren. Die IT ist damit das wichtigste Mittel zur Industrialisierung der betroffenen Anwendungsdomänen (z.B. Buchführung, Produktion, Logistik). Doch gerade die Softwareentwicklung scheint sich gegen eine weit reichende Industrialisierung zu wehren. Rationalisierungen werden nicht dazu verwendet, Systeme effizienter zu erstellen, sondern immer komplexere und mächtigere Systeme zu erschaffen. Eine wirkliche Industrialisierung der IT lässt nun schon seit Jahren auf sich warten. Vielleicht ist dies die Angst der Industrie-Veränderer vor der eigenen Veränderung? Andererseits ist die Geschichte der Softwareentwicklung von einer permanenten Erhöhung der Abstraktionsebenen geprägt, die langfristig eine Industrialisierung unvermeidbar macht. Heute wird beispielsweise kein Unternehmen mehr auf den Gedanken kommen, eigene Prozessoren, Betriebssysteme, Datenbanken, Entwicklungsumgebungen oder Application Server zu erstellen – vor 15 Jahren war dies durchaus noch gängige Praxis. Für diese Systeme existieren mittlerweile entsprechende Standards und De-facto-Standards sowohl im kommerziellen als auch im nichtkommerziellen Bereich.

Mit zunehmender Standardisierung von Produkten und Gütern kommt es zu einer Kommoditisierung ebendieser. Dies bedeutet, dass ein Produkt oder ein Verfahren so weit verbreitet ist, dass es im jeweiligen Umfeld vorausgesetzt werden kann und nicht mehr individuell erstellt werden muss. Generell kann man beobachten, dass Aspekte oder Systeme, welche in vielen Projekten immer wieder neu erstellt werden mussten, mit dem jeweils nächsten Entwicklungsschritt der Basistechnologie eine Schicht tiefer vorhanden und lediglich zu nutzen sind. Diesen Trend der Kommoditisierung hinsichtlich der IT-Layer (Hardware, Netzwerk, Betriebssystem, Middleware, Anwendung) kann man mit der Maslowschen Bedürfnispyramide [1] vergleichen: Die Kommoditisierung einer Stufe beginnt erst dann, wenn die darunter liegende Stufe kommoditisert ist. Nachvollziehen lässt sich dies sehr einfach an den jeweils aktuellen Leitartikeln der Fachpresse: Diskussionen über die ideale Hardware oder das beste Netzwerkprotokoll wichen den lang andauernden Windows vs. Linux vs. UNIX-Diskussionen. Mittlerweile ist die IT auch über die Phase „Java/Java EE vs. .NET" hinweg und dringt nun mit dem Thema SOA langsam in die Ebene der Anwendung selbst vor.

Auch der zweite Kernfaktor, die Wiederverwendung, erreicht durch die Enabler Standardisierung und Kostendruck immer deutlichere Ausmaße. Je umfangreicher das neu zu erstellende Artefakt ist (Klasse > Komponente > Teilsystem > System), desto höher sind der Wunsch bzw. der Zwang zur Wiederverwendung. Auch die Wiederverwendung von Konzepten, Mustern, Prozessen und Werkzeugen wird durch eine lebendige IT-Community und realisierte (De-facto-)Standards (z.B. Java EE, RUP, XP, JUnit) stetig besser. Insbesondere KMUs können sich eine redundante oder Green-field-Entwicklung nicht leisten, hier ist die Wiederverwendung für den Geschäftserfolg essenziell, um Kosten und Risiken zu reduzieren (Steigerung der Effizienz) und die Geschwindigkeit der Entwicklung zu erhöhen (Steigerung der Effektivität). Wie aber die Einleitung zeigt, ist die Grenze hierfür noch lange nicht erreicht.

Bezüglich des dritten Industrialisierungs-Kernfaktors, der Arbeitsteilung, ist zu beobachten, dass das in konservativen Entwicklungsumfeldern noch immer vorherrschende Generalistentum, in welchem jeder Einzelne im Entwicklungsteam mehrere unterschiedliche Aufgaben (und damit Rollen) wie Anforderungsanalyse, Entwicklung und Qualitätssicherung durchführt, zunehmend einer Spezialisierung und Arbeitsteilung in klar definierte Rollen (Projektleiter, Architekt, Analyst, Entwickler etc.) mit jeweils hoher spezifischer Kompetenz weicht.

Auch der Grad an Automatisierung des Entwicklungsprozesses nimmt zu – Ansätze aus den Bereichen MDA/MDSD sowie Generierung von GUI- oder Persistenz-Code sind heute in fast jedem Projekt etabliert. Hier besteht jedoch noch großes Potenzial im Vergleich zu Automatisierungsgraden anderer Industrien.

Der fünfte Kernfaktor, die Verringerung der Fertigungstiefe, setzt sich im Bereich der Softwareentwicklung Schritt für Schritt durch: Es wird heute immer öfter auf bereits existierende, auf dem Markt erhältliche Komponenten und Produkte zurückgegriffen, statt diese selbst zu realisieren. Durch die Verwendung bzw. den Zukauf von kostengünstigeren oder qualitativ hochwertigeren Produkten spezialisierter Anbieter lassen sich die Fertigungstiefe sehr effizient verringern und die knappen Ressourcen für die Kernkompetenz nutzen. Ein weiterer Faktor ist die Outsourcing-Bewegung im Bereich der Softwareentwicklung: Handwerklich orientierte Teile wie Programmierung und Funktionstests werden an externe Dienstleister übergeben, wodurch man sich auf höherwertige Teile der Prozesskette wie Geschäftsprozessanalyse, Strategie, Architektur, Design, Testkonzeption und Projektmanagement konzentrieren kann.

Produktlinien

Der beschriebene zunehmende Grad der Faktoren Wiederverwendung, Automatisierung, Arbeitsteilung, Verringerung der Fertigungstiefe und insbesondere die zunehmende Standardisierung und Kommoditisierung von Produkten, Verfahren und Werkzeugen durch ständige Steigerung des Abstraktionsniveaus führen ein

Unternehmen letztendlich fast automatisch zur Etablierung einer Produktlinie.

Was aber genau sind nun Softwareproduktlinien? Der Begriff leitet sich aus dem Produktlinien-Begriff anderer Industrien wie zum Beispiel der Automobilindustrie ab, in welchen dieses Konzept schon seit vielen Jahren bekannt ist und schrittweise perfektioniert wurde und noch immer wird. Nach Definition des Software Engineering Institute (SEI), welches das Thema als Forschungs- und Beratungsschwerpunkt hat, ist eine Softwareproduktlinie eine Gruppe softwareintensiver Systeme, welche eine Menge identischer gemanagter Funktionalitäten bzw. Komponenten teilen, in einer gemeinsamen Zielmarktdomäne angesiedelt sind und auf Basis einer gemeinsamen Menge von Basiskomponenten auf identische Art und Weise entwickelt werden [2]. Anders gesagt sind Softwareproduktlinien eine Software-Engineering-Technik zur Erstellung eines Portfolios verwandter Softwaresysteme auf Basis einer Menge gemeinsam genutzter Assets, eines gemeinsamen Produktionsprozesses und konsequenter Governance.

Wie in Abbildung 2 dargestellt spielt bei rein projektgetriebenen Organisationen lediglich die Abdeckung der Anforderungen auf Projektebene („Unikate") eine Rolle. Hat das Unternehmen mehrere Kunden mit ähnlichen Anforderungen, kann man die Gemeinsamkeiten in die nächste Abstraktionsebene verschieben und nennt diese „Produkt". Die Gemeinsamkeiten (Invarianten) sind abstrahiert, die Variabilität bei unterschiedlichen Kunden wird durch projektspezifische Anpassungen und unterschiedliche Konfigurationen abgedeckt. Stellt das Unternehmen mehrere Produkte einer gemeinsamen Domäne her, kann man auch hier noch weiter abstrahieren, indem man die Invarianten der Produkte in die nächst höhere Ebene der „Produktlinie" verschiebt, sodass innerhalb der Produkte lediglich die Unterschiede bzw. Einzigartigkeiten gemanagt werden müssen ("provide commonality, manage variability"). Ein Vergleich mit der Automobilindustrie stellt sich folgendermaßen dar: In der Anfangszeit dieser Industrie wurden einzelne Unikate gefertigt (Projekte), später dann in einem Werk höhere Stückzahlen eines Automobiltyps (Produkte). Heute haben wir das typische Bild der Fertigungsstraße, auf der unterschiedliche Automobiltypen mit dem gleichen Herstellungsprozess unter gemeinsamer Verwendung identischer Bauteile gefertigt werden (Produktlinie). Zuletzt ist eine Produktlinie also lediglich eine weitere logische und konsequente Abstraktion mit dem Hintergrund, nicht mehrmals ein und dasselbe Produkt, Verfahren oder Werkzeug neu planen, konzeptionieren und realisieren zu müssen.

Produktlinien zeichnen sich durch ihre Konzentration auf die Entwicklung einer Domäne von Produkten aus, bei der ähnliche funktionale, nichtfunktionale und technische Anforderungen bestehen, wodurch man auf gemeinsame Komponenten und insbesondere einen gemeinsamen Erstellungsprozess setzen kann. Das heißt, die Produktlinie fungiert als Entwicklungsumfeld für eine spezifische Klasse von Systemen, was sie vom reinen marketinggetriebenen Produkt-Branding einer Produktfamilie unterscheidet.

Im Kern sind Produktlinien also nichts Neues, sondern eine Sammlung und Kombination sowie konsequente Umsetzung vieler guter und bekannter Ideen und Vorsätze.

Produktlinien-Prozesse

Innerhalb einer Produktlinie existieren drei zentrale, parallel und iterativ durchgeführte, Prozesse, welche in Abbildung 3 dargestellt sind:

- Produktlinienkern-Entwicklung: Dieser zentrale übergeordnete Prozess stellt die technische, fachliche und organisatorische Plattform zur Verfügung, welche die Instanziierung von einzelnen Produkten der Produktliniendomäne möglich macht. Weiterhin sorgt er für die produktübergreifende Kommunikation, Abstimmung und Governance und ist damit verantwortlich für die Einheitlichkeit aller Produkte gemäß der definierten Plattform. Hier wird auch die zentrale Asset Library gepflegt, welche eine Katalogisierung aller in der Produktlinie vorhandenen Assets zur Wiederverwendung in unterschiedlichen Produkten darstellt. Zusammenfassend spricht man bei diesem Prozess auch vom Domain Engineering.

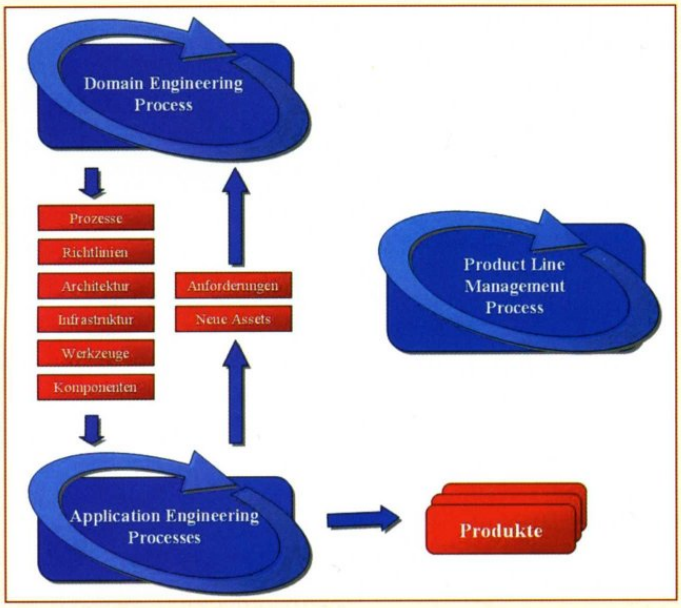

Abb. 3: Produktlinien-Prozesse

- Produktentwicklung: Hierbei geht es um die Entwicklung der vom Markt benötigten Produkte auf Basis der von der Produktlinienkern-Entwicklung zur Verfügung gestellten Plattform und Assets. Man spricht hierbei auch vom „Application Engineering"-Prozess. Für jedes neue Produkt wird ein solcher Prozess angestoßen, bei dem die Phasen Analyse, Design, Implementierung und Test durchgeführt werden. Durch die Anforderungen im Rahmen der Produktentwicklungen wird wiederum die Weiterentwicklung der Produktlinienkern-Entwicklung vorangetrieben.
- Management: Ein geeignetes Management unterstützt, koordiniert und orchestriert beide Prozesse, wobei sinnvollerweise ein technisches und ein organisatorisches Management etabliert werden. Das technische Management überwacht die Entwicklung der Assets, das organisatorische ist für die Planung, Priorisierung und Verteilung der Ressourcen zuständig.

Im Vergleich mit der industriellen Fertigung entspricht dies den Entwicklungsprozessen von Fließband und Robotern (Domain Engineering) gegenüber den darauf erstellten Erzeugnissen (Application Engineering). Die Verwendung und Wiederverwendung der von der Produktlinie bereitgestellten Assets wird im Gegensatz zur einfachen Wiederverwendung strategisch geplant und gesteuert (Management), um hier entsprechende Verbund- und Skaleneffekte zu erreichen. D.h., dass die Gesamterstellungskosten der Produkte innerhalb der Produktlinie geringer als die Summe der Einzelkosten bei deren getrennter Erstellung sind.

Produktlinien-Assets

Um welche Assets handelt es sich nun? Die bekannteste Form eines Assets ist sicher die erstellte und wieder verwendbare Softwarekomponente in unterschiedlicher Form wie technische Plattformdienste und Frameworks, Backend-Adaptoren oder Fachkomponenten. Viel wichtiger als die Wiederverwendung von Code technischer oder fachlicher Natur ist die Wiederverwendung von aufgebautem Know-how und Assets, welche in Papierform vorliegen und nach bzw. mit welchen täglich gearbeitet wird. Dazu zählen zum Beispiel Geschäftsmodelle, Anforderungsspezifikationen, Projektpläne, Risikoanalysen, Use-Cases, domänenspezifische Patterns, Geschäftsprozessmodelle sowie definierte verbindliche Prozesse und Richtlinien. Damit ist es in einer konsequenten Produktlinienentwicklung möglich, dass für mehrere Produkte identische Prozesse hinsichtlich Anforderungsanalyse, Entwicklung, Review, Freigaben, Test usw. gelebt, identische Templates und Ablagestrukturen für alle Dokumente und Modelle verwendet und die Produkte nach denselben Architektur-, Design- und Coding-Richtlinien erstellt werden und damit eine einheitliche Form aufweisen.

Weiterhin wird die mit hohem Aufwand geplante, realisierte und gewartete Infrastruktur für Entwicklung und Test von allen Produkterstellungs-Teams verwendet. Dies reicht von der integrierten Entwicklungsumgebung auf jedem Entwickler-PC über Werkzeuge für Requirements Engineering und Geschäftsprozessmodellierung bis hin zu den zentralen Werkzeugen wie Versionsverwaltungssystemen, Build-Systemen oder umfangreichen Testumgebungen.

Jedes Artefakt, welches im Rahmen der Produktlinie (sowohl im Domain- als auch im Application Engineering) erstellt wird, ist automatisch Bestandteil der Produktlinie und wird als Asset in der zentralen Asset-Library inventarisiert. Wie in den Abbildungen 4 und 5 dargestellt, dienen diese Assets in Kombination mit produktspezifischen Entscheidungen wiederum als Basis zur Erstellung neuer Produkte. Da bei einem neuen Produkt lediglich die Einzigartigkeit des Produktes, also der Unterschied zur Produktlinie, analysiert werden muss, spricht man hier auch von Delta Engineering.

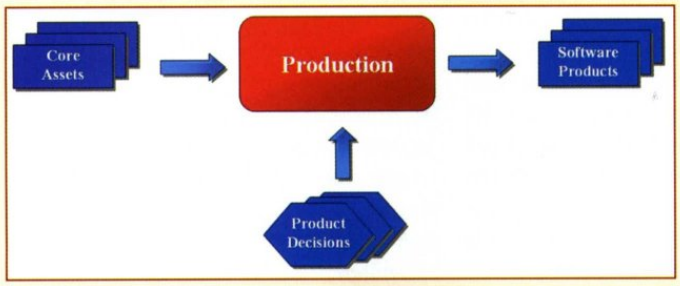

Abb. 4: Produktlinien-Produktionsprozes

Variation Points

Produkt- oder systemumgebungsspezifische Variabilität in Technik oder Fach-

lichkeit wird durch entsprechende Konfigurationspunkte, so genannte Variation Points (auch „Feature-Weichen" genannt) abgebildet. Sie spielen damit eine zentrale Rolle im Product Line Engineering und treten in unterschiedlichster Form auf. Dies reicht von der Unterstützung unterschiedlicher Betriebssysteme und Plattformen über unterschiedliche Konfiguration eines Adapters zur Anbindung verschiedener Fremdsysteme, Unterstützung unterschiedlicher Client-Typen bis hin zum komplexen Customizing der realisierten Geschäftsprozesse. Weiterhin sind verschiedene Entscheidungen hinsichtlich des Zeitpunkts des Bindens eines Variation Point möglich: Typische Ansätze sind das Binden zur Build-, zur Installations- oder zur Laufzeit. Produkte sind damit lediglich spezifische Konfigurationen von Produktlinienkomponenten. Im Konfigurationsmanagement wird die Version eines Produktes lediglich als Sammlung von Referenzen auf Versionen von Produktlinienkomponenten verwaltet (Abb. 6)

Produktlinien-Architektur

Das Kernstück einer Produktlinie ist sicherlich die Produktlinien-Architektur, an welcher alle Produktlinienprozesse, Werkzeuge und Infrastrukturkomponenten ausgerichtet sind. In der Produktlinien-Architektur wird festgelegt, wie aus den Produktanforderungen und Geschäftsprozessmodellen über Analyse-

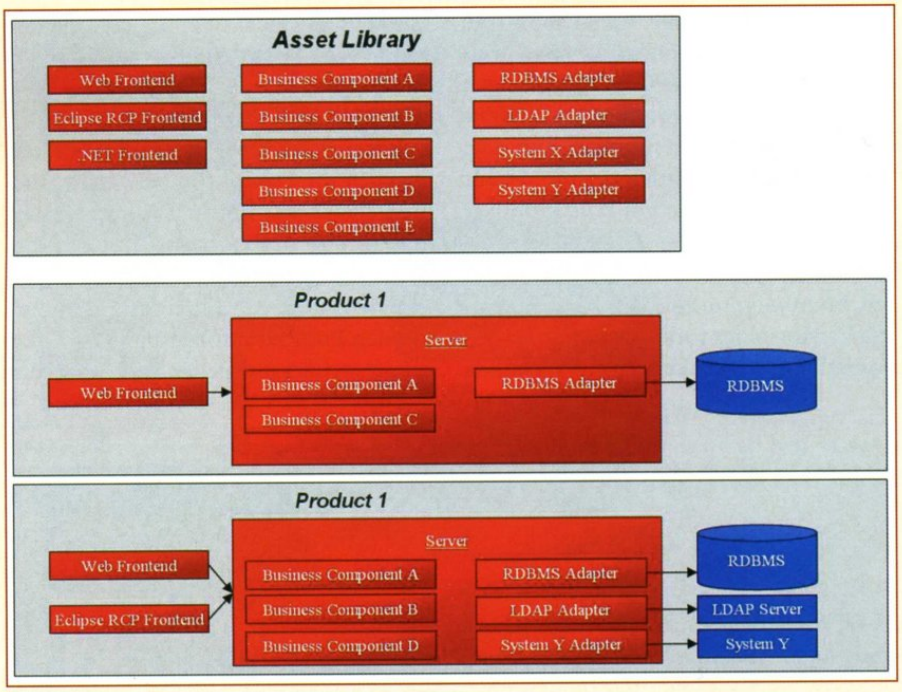

Abb. 5: Asset Library und Konfiguration

und Design-Phasen das jeweilige Produkt bzw. die jeweiligen Produktlinienkomponenten erstellt werden können. Die Produktlinien-Architektur definiert damit alle Aspekte der fachlichen und technischen Architektur des Domain Engineering wie technische- und Domänen-Nomenklatur, Aufbau der Artefakte, unterstützte Systemumgebungen, Einsatz der Variation Points u.v.a.m. Ziel dieser Referenzarchitektur muss es sein, dass alle Mitglieder des Produktlinienteams das gleiche Verständnis vom Aufbau der Komponenten und Produkte besitzen und dieselbe Nomenklatur und Notation verwenden, sodass Mehrdeutigkeiten vermieden werden. Dadurch wird ermöglicht, dass im Produktionsprozess die fachliche Architektur und die Diskussion über fachliche Schnittstellen, Prozesse und Services im Vordergrund stehen. Die technische Architektur muss nach einiger Zeit so selbstverständlich sein, dass sie intuitiv verwendet wird und nicht wie in vielen Entwicklungsvorhaben

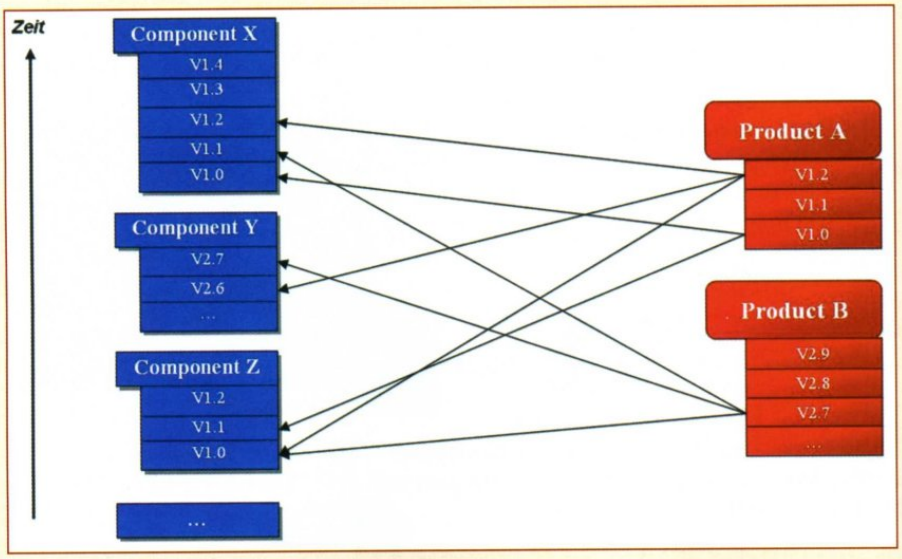

Abb. 6: Sichtenbildung auf Artefakte

Gegenstand der meisten Diskussionen („Sollen wir nicht doch Hibernate anstelle von CMP verwenden?“ etc.) ist.

Grundvoraussetzung einer solchen Produktlinien-Architektur muss eine stark ausgeprägte Komponentenorientierung sein, welche voneinander unabhängige, aber kombinierbare Module mit klar definierten Schnittstellen erzwingt und damit das Assemblieren von unterschiedlichen Produkten aus diesen Komponenten ermöglicht.

Durch die für die gesamte Produktlinie gültige zentrale Architektur kann für alle Produkte die Erfüllung von nichtfunktionalen Anforderungen wie Performance, Verfügbarkeit, Skalierbarkeit, Erweiterbarkeit, Robustheit etc. leichter sichergestellt werden, da diese Aspekte zum größten Teil durch die Architektur selbst abgedeckt werden.

Da sich die Produktlinie auf eine spezifische Domäne bezieht, kann zudem das gesamte Umfeld domänenspezifisch ausgerichtet werden. Hierzu gehört der Einsatz domänenspezifischer Werkzeuge, Prozesse, Patterns, Frameworks etc. Insbesondere lohnen sich durch die gemeinsame Referenzarchitektur im Produktlinienumfeld generative Ansätze wie MDA/MDSD, welche bei einem einzelnen Produkt meist aufgrund des hohen initialen Aufwands zu teuer sind. Der Einsatz dieser Ansätze wirkt sich im Umkehrschluss zudem positiv auf die vorgeschriebene identische Struktur des Assets aus.

Für eine Produktlinie existiert keine allgemeingültige allseits bekannte publizierte Architektur. Jedes Unternehmen bzw. jede zuständige Organisationseinheit muss hier eine auf die Domäne zugeschnittene und für das Entwicklungsumfeld geeignete Produktlinien-Architektur entwickeln. Ein Beispiel bildet die vom Autor in [3] beschriebene Produktlinien-Architektur für betriebswirtschaftlich orientierte, serverbasierte Lösungen auf Basis von SOA, CBD und Java EE.

Vorteile

Welche Vorteile ergeben sich aus dem Product-Line-Engineering-Ansatz gegenüber der konventionellen Produktentwicklung? Eine konsequent umgesetzte „lebende“ Produktlinie verschafft der

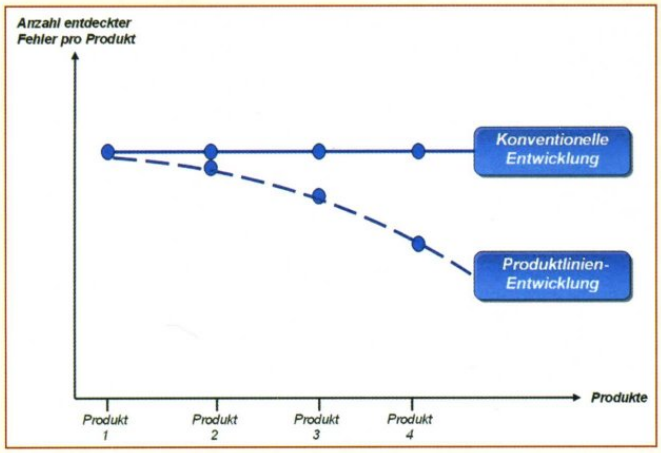

Abb. 7: Qualität in Produktlinien

jeweiligen Organisationseinheit enorme Wettbewerbsvorteile: Sind die Produktlinie und ein erster Fundus an Assets erst einmal vorhanden, kommt es bei jedem zukünftig zu erstellenden Produkt zu direkten Einsparungen hinsichtlich der im Folgenden genannten Faktoren.

Die Hauptvorteile wie eine kürzere Time to Market, Time to Revenue und drastische Reduktion der Kosten entstehen aus der strategischen Wiederverwendung von Komponenten, Infrastruktur, Architektur, Prozessen, Methoden und Werkzeugen sowie aufgebautem Wissen. Bei jedem neuen Produkt muss lediglich das Delta zum bereits bestehenden Asset-Bestand realisiert werden. Daraus erwächst auch die Fähigkeit, auf effiziente Art und Weise mehrere Varianten (je nach Markt, Umgebung, Kundenwunsch) ein und desselben Produktes mit unterschiedlichem Funktionsumfang (Stichwort: Express-, Standard-, Enterprise-Edition) zur Verfügung zu stellen.

Wie in Abbildung 7 dargestellt, ist ein weiterer Vorteil die höhere Qualität der Produkte, da bereits getestete Komponenten und Prozesse verwendet werden. Weiterhin sind die in Produkt A aufgetretenen Fehler nach deren Korrektur automatisch in Produkt B behoben (sofern es sich um gemeinsame Bestandteile handelt), was sich wiederum positiv auf den Wartungsaufwand auswirkt. In Abbildung 8 wird der Engineering-Gesamtaufwand (und damit die Kosten) für das Entwickeln, Warten und Betreuen einer Menge von Softwareprodukten in einer Domäne dargestellt – einmal für eine Produktentwicklung, einmal für eine Produktlinien-Entwicklung. Hierbei sieht man, dass, wenn beim ersten Produkt bereits allgemeingültige Entwicklungs-/Test-Prozesse, Infrastruktur und eine tragbare Produktlinien-Architektur definiert wurden, sich der Einsatz einer Produktlinie in der Regel bereits beim zweiten Produkt bemerkbar macht und die Differenz zur

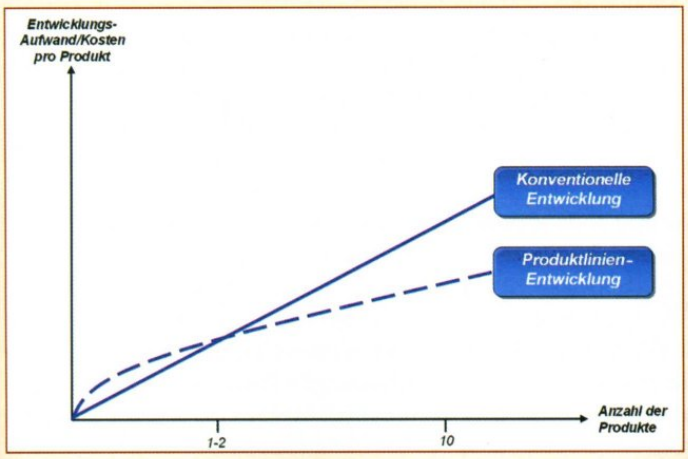

Abb. 8: Amortisierungsverlauf einer Produktlinie

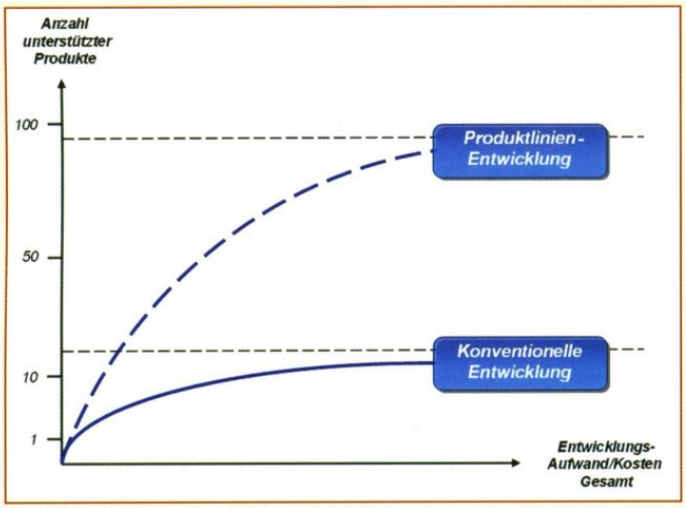

Abb. 9: Produktkapazität einer Organisationseinheit

konventionellen Softwareentwicklung bei jedem Produkt deutlich zunimmt. Insbesondere die initialen Aufwände beim Start eines neuen Produktes reduzieren sich drastisch, da keine Architektur- und Prozessdefinitionsphase anfällt und die grundlegende Infrastruktur bereits existiert. Zudem werden etablierte Prozesse mehrfach gelebt und können permanent an geänderte Rahmenbedingungen angepasst und weiter optimiert werden. Insgesamt erhöht sich auch die Produktkapazität einer Organisationseinheit, d.h., die maximale Anzahl managebarer Produkte einer Organisationseinheit: In Abbildung 9 sieht man, dass beide Entwicklungsansätze sich einem Maximum nähern, bei dem sich die Organisationseinheit hauptsächlich mit Weiterentwicklung und Wartung der bestehenden Produkte beschäftigt, aber aus Ressourcenmangel keine neuen Produkte angehen kann. Bei vergleichbarem Entwicklungsaufwand ist dieses Maximum bei der konventionellen Produktentwicklung allerdings sehr viel früher erreicht.

Ein weiterer Vorteil liegt in einer deutlich effektiveren Ressourcennutzung, indem Spezialisten für bestimmte Bereiche (z.B. Prozessdesign, Architektur, GUI, Persistenz etc.) in unterschiedlichen Produkterstellungsprozessen eingesetzt werden können und bei Engpässen und Freiräumen Mitarbeiter ohne größeren Einarbeitungsaufwand zwischen Produktteams verschoben werden können.

Daneben erhöht sich die Planungssicherheit im Rahmen der Produktlinie, da immer mehr verlässliche Werte für Aufwände zur Erstellung von Assets vorliegen, welche wegen der Gleichartigkeit zukünftiger Assets als Basis für Aufwandschätzungen dienen können.

Weiterhin werden etablierte Kommunikationspfade, Rollen, Verantwortlichkeiten und Strukturen produktübergreifend genutzt, was zum einen deren Effizienz steigert und zum anderen der Vereinfachung und Klarheit dient. Netzwerke dieser Form haben länger Bestand und sind umfangreicher als Produktnetzwerke oder gar kurzlebige Projektnetzwerke.

Zusätzlich ist die Motivation eines Produktlinienteams oft höher, da jeder im Team weiß, dass er mit „etwas Großem" arbeitet und seine Arbeitsergebnisse gegebenenfalls in mehreren Produkten und über einen längeren Zeitraum Bestand haben. Dadurch, dass das gesamte Produktlinienteam das gleiche Know-how hat, die gleichen Werkzeuge benutzt, den gleichen Prozess lebt und insgesamt damit die gleichen Vorteile und Herausforderungen hat, entstehen auch in einem größerem Team ein motivierendes „Wir-Gefühl" und eine gegenseitige Unterstützung bei Fragen und Problemen, welche sich sehr positiv auf die Motivation und Produktivität des Teams auswirkt.

Herausforderungen

Bei den beschriebenen Vorteilen stellt sich nun natürlich die Frage, warum nicht jedes Softwareunternehmen diesen Ansatz verfolgt. Vermutlich liegt dies daran, dass der Weg zu einer funktionierenden Softwareproduktlinie steinig ist und es viele Hindernisse zu überwinden und Herausforderungen zu bezwingen gilt.

Die technische Seite spielt hierbei eine untergeordnete Rolle. Lediglich die Unterstützung durch Standard-Tools für das Produktlinienumfeld stellt hier eine Schwierigkeit dar, da die Werkzeuge in der Regel speziell für die Einzelprodukt-Erstellung optimiert sind. Im Produktlinienumfeld würde zum Beispiel ein Anforderungsmanagement-Werkzeug die Zuordnung von Anforderungen zu mehren Produkten unterstützen und ein Projektmanagement-Werkzeug müsste in der Lage sein, die Projektpläne der unterschiedlichen Produkte zu kombinieren, sodass zum Beispiel die Auswirkung einer Terminverschiebung in einem Produkt auf die anderen Produkte oder produktübergreifende kritische Pfade sichtbar werden. In Entwicklungsumgebungen müsste die Möglichkeit bestehen, alle Artefakte der Produktlinie zu importieren und durch eine geschickte Sichtenbildung nur die zur jeweiligen Zeit benötigten darzustellen. Wird dann zum Beispiel an einer durch mehrere Produkte verwendeten Komponente eine Schnittstellenänderung durchgeführt, sollte die Entwicklungsumgebung direkt die betroffenen Stellen markieren. Insbesondere im Bereich der Konfigurationsmanagement-Werkzeuge wäre eine Produktlinien-Unterstützung extrem hilfreich, da zur Verwaltung der Artefakt-Variationen hinsichtlich der Zeit nun der Aspekt der Variation bezüglich der Produkte hinzukommt (Abb. 5). Derzeit muss man sich also bei all diesen Werkzeugen über eigene Notationen, Konventionen und Prozesse selbst eine geeignete Arbeitsumgebung zur Unterstützung des Produktlinienansatzes aufbauen.

Schwierig gestaltet sich der Aspekt der Planung im Produktlinienumfeld, welcher deutlich komplexer als im Produktumfeld ist. Hier müssen insbesondere bestehende Assets und Abhängigkeiten zu gemeinsam verwendeten Assets berücksichtigt werden. Beispielsweise kann es zur Gefährdung von Auslieferungsterminen kommen, wenn sich Termine für

Komponenten verschieben, von denen andere Komponenten bzw. Produkte abhängen. Dieses Problem ist in fortgeschritteneren Industrien wohlbekannt und wird durch spezielle Planungsmethodiken (z.B. Nutzung von Gozintographen) angegangen. Weiterhin ist die typische Planung oft rein featuregetrieben, wobei die dabei entstehende Reihenfolge aber nicht selten im Konflikt zur Architektur steht, da hierbei erst die zugrunde liegenden Komponenten erstellt werden müssen, bevor höherwertige Komponenten auf diese aufgesetzt werden können (zum Beispiel Stammdaten vor Bewegungsdaten vor Prozessen). Daher führt nach Meinung des Autors nur eine geschickte Kombination mit einer architektur- und assetgetriebenen Planung zu optimalen Ergebnissen.

Ein weiterer wichtiger Punkt im Rahmen einer Produktlinie ist der konsequente Top-down-Engineering-Ansatz, d.h., es müssen zunächst die fachlichen Anforderungen betrachtet werden, erst im letzten Schritt finden technische Betrachtungen Berücksichtigung. Bei jeder relevanten fachlichen Anforderung muss eine Prüfung erfolgen, ob diese durch bestehende Komponenten aus der Asset Library abgedeckt („mine"), ein COTS-Modul (commercial off the shelf) zugekauft werden kann („buy") oder eine Beauftragung eines Dienstleisters erfolgen soll („commission"). Lediglich im Fall der Konzentration auf die Kernkompetenz sollte das Modul innerhalb der Produktlinie selbst realisiert werden („make"). Alle Entwicklungsprozesse in einer Produktlinie müssen effektiv, gut organisiert und allgemein bekannt sein und gelebt werden. Ansonsten droht die Produktlinie an den Sollbruchstellen zwischen den Produkten auseinander zu brechen, da jedes Produktteam seine eigenen Prozesse etablieren würde.

Eine weitere Schwierigkeit stellt die hohe Anfangsinvestition einer Produktlinie dar, die deutlich über der Investition eines einzelnen Produktes liegt. Zudem ist die Entwicklung einer wirklich wiederverwendbaren Komponente bis zu 50 Prozent teuerer als die Entwicklung derselben Komponente für den einmaligen Gebrauch, da hier mehr Aufwände

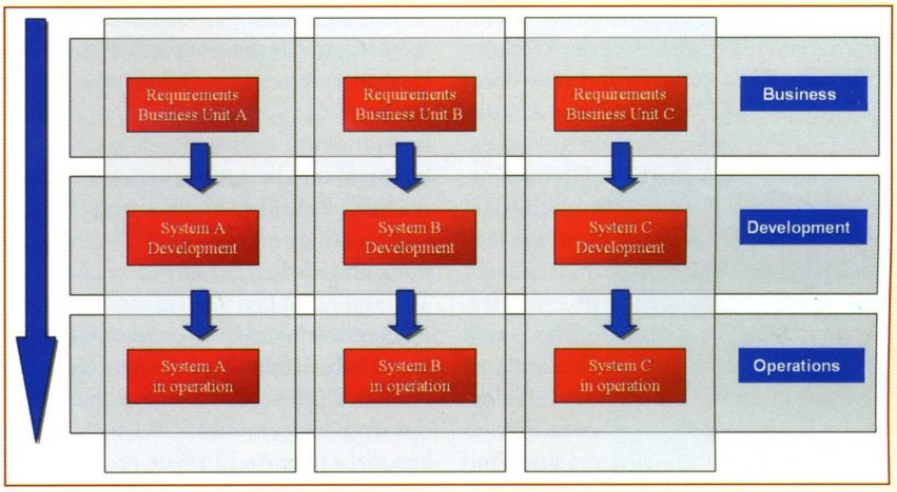

Abb. 10: Silos

im Bereich Analyse, Design und Implementierung entstehen. Meist werden die Budgets aber nur im Rahmen eines Produktes geplant, sodass nicht an andere und nachfolgende Produkte gedacht wird bzw. gedacht werden kann. Wie in Abbildung 10 dargestellt, ist dies auch der Hauptgrund für die Entstehung der Anwendungssilos: Die Wertschöpfungskette wird organisatorisch in Einzelteile aufgesplittet, die von einzelnen Organisationseinheiten verantwortet werden. Jede Organisationseinheit versucht nun ihren Anteil am Gesamtprozess durch den geschickten IT-Einsatz zu optimieren. Da dies unabhängig voneinander geschieht, entstehen grundlegend unterschiedliche IT-Systeme, welche später im Rahmen des Gesamtprozesses schwierig miteinander integriert werden müssen. Silo-Anforderungen in Kombination mit Silo-Budgetierungen und Silo-Management führen zu Silo-Entwicklungen welche zu Silo-Systemen führen!

Die Organisation in einer Produktlinie

Interessanterweise ist die größte Herausforderung, welcher eine Organisation bei der Transformation zum Produktlinien-Engineering entgegensteht, die Organisation selbst! Der Grund dafür ist folgender: Bei Etablierung einer Produktlinie darf nicht der Fehler gemacht werden, die bisherige Organisation und Prozesse beizubehalten, denn Produktlinien-Entwicklung unterscheidet sich grundsätzlich von Produktentwicklung! Mit dem Einsatz einer Produktlinie und der damit verbundenen Software-Industrialisierung muss sich die Organisation anders organisieren. Generell verursacht aber jede größere Veränderung bei den jeweils betroffenen Menschen Widerstände und Ängste, welche im Rahmen eines geeigneten Change-Managements auf Management- und Team-Ebene unbedingt berücksichtigt werden müssen, da das Vorhaben ansonsten zum Scheitern verurteilt ist.

Insbesondere am Anfang bestehen natürliche Widerstände der einzelnen Rollen gegen den Produktlinienansatz: Beispielsweise wollen Produktmanager ihre Autonomie nicht verlieren – in der Produktlinie müssen sie sich verstärkt abstimmen und bestimmte Aspekte sind durch die Architektur und Basiskomponenten strikt vorgegeben. Das obere Management ist von den mittel- und langfristigen Einsparpotenzialen begeistert, beim anfänglichen Aufbau der Produktlinie ist aber die Performance der Produktteams eingeschränkt und es fallen zusätzliche Kosten an.

Die grundlegende und wichtigste Entscheidung hinsichtlich des Organisationsaspektes einer Produktlinie ist das geschickte Ziehen der Produktliniengrenze (organisatorische Architektur): Wird jedes Produkt der Produktlinie durch ein eigenständiges abgeschottetes Team erstellt, geht der Produktlinien-Gedanke unter. Werden alle Produkte durch ein großes gemeinsames Team erstellt, zerfällt dieses mit der Zeit in Einzelteams. Natür-

lich spricht technisch gesehen oft viel für eine maximale Ausdehnung der Produktlinie, um ein Maximum an Effektivität und Effizienz zu erreichen. Diese Ansätze scheitern aber in der Regel am Menschen, der sich immer in gewissen Grenzen organisiert. Dies entsteht dadurch, dass sich eine Organisationseinheit wie jede Gruppe von Lebewesen verhält. Hierzu gehören Punkte wie Wachstumsbestreben in gewissen Grenzen, Existenz- und Autonomiebestreben gegenüber anderen Gruppen. Vorhaben, welche den effizientesten Ansatz verfolgten, aber an den natürlichen Grenzen gescheitert sind, sind zum Beispiel die Zusammenlegung von staatlichen Einheiten mit einheitlicher Führung, Kultur und Wertvorstellungen oder der Aufbau der gemeinsamen Weltsprache Esperanto.

Es existieren also sowohl technische als auch organisatorische Grenzen von Produktlinien, d.h., eine zu breit gefasste Produktliniendomäne verursacht mehr Probleme, als sie Nutzen erbringt. Die Kunst besteht nun darin, im Unternehmen geeignete Produktliniendomänen zu identifizieren und dort innerhalb des Produktlinienteams entsprechende Unterteams zu bilden (z.B. Produktteams), welche aber als Teil des Ganzen agieren müssen. Bei einem großen IT-Hersteller wie HP wären aus technischer Sicht geeignete Domänen zum Beispiel kommerzielle serverbasierte Anwendungssoftware, mobile Systeme, Druckertreiber und Hostsysteme. Aus organisatorischer Sicht ist auf Gegebenheiten wie räumliche Entfernung, Mentalität und Skills zu achten, sodass diese technischen Domänen auf die organisatorischen Gegebenheiten abgebildet werden können.

Eine weitere Herausforderung ist die Komplexität der Produktlinienentwicklung: Da sich die Produktlinienarchitektur auf einem hohen Abstraktionslevel befindet, stellt die Mitarbeit an der Produktlinie bzw. an einem Erzeugnis dieser erhöhte Anforderungen an das Abstraktionsniveau und insbesondere an das Denken in Komponenten und übergreifenden Konzepten. Der Einzelne arbeitet nicht mehr nur an einem einzelnen Produkt, dessen Scope er ohne Probleme überblicken kann, sondern an einer Komponente, welche in diversen Produkten eingesetzt werden kann. Insbesondere Entwickler im Umfeld des Core Asset Development müssen ein tiefes Verständnis der relevanten Domäne und entsprechende Weitsicht besitzen, um die gemeinsamen Bestandteile der Produkte zu abstrahieren und zu generalisieren, aber auch die Entscheidung hinsichtlich der Einzigartigkeit und Variabilität in den Produkten zu treffen. Andererseits wird die Komplexität durch die klaren, definierten Rahmenwerke der Produktlinie wieder ausgeglichen. Produktentwickler müssen sich keine Gedanken mehr über Architektur, Prozesse und Infrastruktur machen und können sich auf das konzentrieren, was an einem Produkt einzigartig ist, nicht auf das, was alle Produkte gemeinsam haben.

Durch den Aspekt der Wiederverwendung mit in unterschiedlichen Produkten gemeinsam genutzter Infrastruktur und Komponenten spielt die Disziplin aller Beteiligten bezüglich Informationsfluss und Abstimmung innerhalb der Produktlinie eine entscheidende Rolle. Bei Änderungen und Erweiterungen an gemeinsam verwendeten Assets müssen alle jeweiligen Verantwortlichen in den Entscheidungs- und Informationsprozess eingebunden sein, da sonst insbesondere in großen Teams die notwendige Einheitlichkeit zerstört und damit die gemeinsame Strategie gefährdet ist. Da dieses Verständnis in der Regel nicht von alleine entsteht, müssen hier zwingend geeignete Governance-Prozesse (Regelwerke, Genehmigungsverfahren, Kontroll- und Eskalationsverfahren etc.) etabliert werden, um die Disziplin zu gewährleisten. Mitarbeiter, welche sich nicht an die definierten Regeln der Produktlinie halten, müssen zum Beispiel durch Versetzung in weniger relevante Projekte ausgetauscht werden, bei den Produkten muss die Konformität an das Budget geknüpft werden usw. Im Zusammenhang mit der notwendigen Governance ist weiterhin ein Produktlinien-Rollenmodell grundlegend, in welchem für jede Rolle klar definierte Aufgaben, Verantwortlichkeiten und Kompetenzen festgeschrieben sind. Aus Sicht des Autors sind innerhalb einer Produktlinie folgende Rollen (im Bereich der Entwicklung) zwingend notwendig: Für jedes Produkt der Produktlinie muss ein Produktverantwortlicher bzw. Projektleiter existieren, parallel dazu sollte es je nach Komplexität des Produktes für jedes Produkt einen Architekten geben. Zur Gewährleistung der Stabilität der Produktlinie muss es zwingend zwei übergeordnete und möglichst produktneutrale Rollen geben, eine für technische (Produktlinien-Architekt) und eine für organisatorische Entscheidungen (Produktlinien-Manager), welche bei Dissensen die letztendliche Entscheidung treffen. Sowohl Governance- als auch Rollen-Modell müssen sich an Organisation, Kritikalität und insbesondere Team-/Budget-Umfang ausrichten, damit keine unnötige Bürokratie (mit wiederum höheren Kosten) entsteht.

Während sich manche in einem solchen Regelwerk und Grenzen geborgen fühlen, führt dies bei anderen dazu, dass sie sich in ein Rahmenwerk gezwängt fühlen, das mit einem scheinbaren Verlust der Kreativität des Einzelnen dahergeht (im Sinne der Fließbandfertigung). Dies kann durch unterschiedliche Ansätze vermieden werden: Reicht der Skill des Betroffenen aus, kann er in die Gestaltung des Rahmenwerks oder in die Forschung bzgl. der Weiterentwicklung der Produktlinie (Bereich: Domain Engineering) eingebunden werden. Andere Möglichkeiten sind Job-Rotation in den unterschiedlichen Produkten oder die Spezialisierung des Einzelnen (z.B. für den Bereich Datenbanken, Testinfrastruktur, Architektur, Prozesse, GUI-Entwicklung), der dann sein Spezialwissen allen Produkten der Produktlinie zur Verfügung stellt und dadurch auch noch umfangreicher aufbauen kann.

Bei einer Trennung des Produktlinienteams in ein Core-Asset-Developer-Team und mehrere Product-Developer-Teams kann es zu Konflikten zwischen diesen Teams kommen, da die Aufmerksamkeit des Managements in der Regel eher auf den kundennäheren Produktentwicklungsteams liegt und lediglich auf die Kernentwicklung schwenkt, wenn etwas nicht funktioniert. Meist enthält die Kernmannschaft aber die eigentlichen Kompetenzträger, welche für eine solide Basis und einen reibungslosen Prozessablauf sorgen, deren Arbeit aber nie beendet ist, und wel-

che im Gegensatz zu den Produktteams keine Releases oder erfolgreiche Kundeninstallationen feiern können. Deshalb und aus Sicht des gegenseitigen Verständnisses und frühzeitigen Erkennens und Verstehens gegenseitiger Probleme ist aus Sicht des Autors keine strikte Trennung zwischen den Teams (Domain Engineering und Application Engineering) anzustreben. Alle Beteiligten der Produktlinie müssen als Einheit agieren, was durch Teambildungsmaßnahmen wie gemeinsame Veranstaltungen oder Vorgabe gemeinsamer Ziele optimiert werden kann.

Bei der Entwicklung von Produkten auf Basis einer Produktlinie besteht generell Konfliktpotenzial. Da die Produkte durch die Produktlinie miteinander gekoppelt und voneinander abhängig sind, kann es im Gegensatz zu einer parallelen, autarken Entwicklung zu Dissensen bzgl. der gemeinsamen Strategie, Prioritäten der einzelnen Produkte sowie geeignetem Einsatz der zur Verfügung stehenden Ressourcen kommen. Zum Beispiel sind die Leistungsträger innerhalb des Gesamt-Produktlinienteams ein typischer Grund für Auseinandersetzungen, da jeder Projektleiter gern seine produktspezifischen Arbeitspakete an diese delegiert. Natürlich ist es hierbei sinnvoll, Leistungsträger nicht nur einem Produkt zuzuordnen, sondern die Kapazitäten auf die Produkte zu verteilen, wobei aber die Gefahr von „Flaschenhälsen" entsteht. Wichtig und schwierig ist weiterhin, dass auch unter Termindruck und hohen Prioritäten eines einzelnen Produktes die Entscheidungen immer in Hinblick auf die gemeinsamen Produktlinienziele getroffen werden. Insbesondere hier ist ein entsprechendes Management-Commitment bzgl. des Produktlinienansatzes gegenüber dem einzelnen Produkt absolut notwendig und fundamental.

Fazit

Die Industrialisierung, welche sich durch den zunehmenden Grad der Faktoren Standardisierung/Kommoditisierung, Wiederverwendung, Arbeitsteilung, Automatisierung und Verringerung der Fertigungstiefe auszeichnet, macht auch vor der Softwareentwicklung nicht Halt. Eine Konsequenz dieser Entwicklung ist die fortschreitende Abstraktion der Softwareentwicklung von der Projekt- über die Produkt- bis hin zur Produktlinienentwicklung, welche bei anderen Industrien schon stattgefunden hat.

Die Hauptvorteile des Product Line Engineering liegen in einem deutlich verbesserten Time to Market, einer Reduktion der Produkterstellungskosten, einer höheren Produktqualität und einer verbesserten Portfolio-Skalierbarkeit. Daraus ergibt sich ein Wettbewerbsvorteil vergleichbar der Umstellung der ersten Unternehmen auf Massenproduktion zu Beginn der Industrialisierung.

Die größte Herausforderung, welche einer Organisation bei der Einführung einer Produktlinie entgegensteht, ist die Organisation selbst, da die Einführung (wie auch bei der Industrialisierung anderer Industrien) weit reichende Veränderungen der Prozesse, Denkweisen, Werkzeugen etc. mit sich bringt. Zudem muss der Einzelne hierbei oft Teile seines Einflussgebietes zugunsten eines gemeinsamen, höheren Zieles aufgeben. Die dabei entstehenden Widerstände und Ängste vor der Veränderung müssen durch ein geeignetes Change Management auf Management- und Teamebene abgebaut werden, um nach Einschwingen des Systems die angesprochenen Vorteile vollständig ausnutzen zu können.

Mit einem konsequenten Produktlinienansatz gelingt es, die typischen Marketing-Folien der Hersteller, in welchen eine Reihe an Produkten als gemeinsame aufeinander abgestimmte und voll integrierte Produktfamilie verkauft werden, Wirklichkeit werden zu lassen.

Dipl.-Inform. (FH) **Roger Zacharias** ist Sun Certified Enterprise Architect und als Software Engineer bei der Wincor Nixdorf AG beschäftigt. Dort nimmt er zurzeit die Rollen des Produktlinien-Architekten und -Projektleiters ein. Als JCP-Mitglied beteiligt er sich zudem aktiv an der Weiterentwicklung der Java/Java EE-Plattform.

Links & Literatur

[1] Abraham H. Maslow: Motivation und Persönlichkeit, rowohlt, 1954

[2] Software Engineering Institute: Product Lines (9/2006): www.sei.cmu.edu/productlines/

[3] Roger Zacharias: Serviceorientierung – Der OO-König ist tot, es lebe der SOA-König!, in OBJEKTspektrum 2/2005