Vom Projekt zum Produkt – SAP's Weg zum „Lean Software Product Development"

Olaf Mackert, Tobias Hildenbrand, Almer Podbicanin

SAP AG
Dietmar-Hopp-Allee 16
69190 Walldorf
Germany
{olaf.mackert, tobias.hildenbrand, almer.podbicanin}@sap.com

Abstract: Die Erstellung von Software, insbesondere von Unternehmenssoftware, ist ein komplexes Thema. Die Komplexität nimmt mit Anzahl der involvierten Mitarbeiter und der Auswahl an Produkten bzw. unterschiedlichen Kundensegmenten überproportional zu. Ziel dieser Abhandlung ist es, einen groben Überblick über die Geschichte eines großen Softwareanbieters zu geben und dabei insbesondere die Evolution des methodischen Ansatzes zur Softwareerstellung bzw. der Prozessmodelle darzulegen. Hierbei werden die Entwicklungsschritte hin zu aktuellen Lösungsansätzen aufgezeigt, mit welchen der Grad der Komplexität beherrschbar gemacht und eine höhere Skalierbarkeit der Entwicklungsorganisation erreicht werden soll.

1 Einleitung

Softwareerstellung erfolgt klassischerweise und vor allem in kleinen bis mittelgroßen Softwarehäusern intuitiv in enger Zusammenarbeit mit dem Kunden als Projektgeschäft. Wenn Softwarehäuser über die Zeit wachsen und neben reinen Kundenprojekten auch Standardsoftware anbieten, wird es sehr schwierig, die entsprechenden Skaleneffekte zu erzielen und gleichzeitig die Bedürfnisse des Marktes sowie der einzelnen Kunden zu treffen. Viele große Softwareanbieter führen daher Projektmanagement- und Prozessstandards ein.

Trotz bzw. gerade wegen des meist stark arbeitsteiligen Charakters dieser Standards kommt es häufig zu Problemen in der Skalierung – wie beispielsweise unzureichend durchgängige Verantwortlichkeiten, Nicht-Einhaltung gemeinsamer Prozessstandards sowie Multi-Tasking innerhalb von Teams bzw. bei den einzelnen Mitarbeitern. Zudem ist die Produktivität auf Unternehmensebene – z.B. gemessen in Umsatz pro Mitarbeiter – aufgrund dessen häufig rückläufig [OVoJ]. Eine verstärkte Ausrichtung möglichst aller Prozesse auf den Kunden mit Fokussierung auf den direkten Kundennutzen, wie beispielsweise in Lean Management-Ansätzen [Lik04], verspricht weitere Skaleneffekte durch höhere Kundenzufriedenheit – ähnlich wie im Projektgeschäft bei kleineren Unternehmen – und damit höhere Nachfrage und Umsätze.

Am Beispiel der SAP AG wird die Entwicklung vom anfangs rein kundenbezogenen Projektgeschäft hin zur heutigen Mischung aus Standardsoftwareentwicklung und speziellen Kundenprojekten (Geschäftsbereich „Custom Development“) beschriebenen. Ziel dieser Abhandlung ist es hierbei, die Evolution der Prozessstandards und deren unterschiedlichen Fokus auf Projekte, Produkte und Lösungen hin zum heutigen „Lean Software Product Development“-Ansatz aufzuzeigen. Im Lean-Ansatz tauchen Tugenden der Anfangszeit wieder auf, es wird versucht die Erfahrungen, die über Jahrzehnte gemacht wurden zu nutzen, um die Fehler der Vergangenheit nicht zu wiederholen.

Hierzu wird im folgenden Abschnitt zunächst ein Überblick über die historische Entwicklung des Entwicklungsansatzes, des Prozessmanagements bei SAP und eine Betrachtung der gewachsenen Probleme in der Standardsoftwareentwicklung gegeben. Abschnitt 3 stellt den aktuellen Ansatz „Lean Software Product Development“ vor und den Weg dorthin dar. Darüber hinaus gibt Abschnitt 4 eine Zusammenfassung der Ergebnisse. Abschnitt 5 lässt einen Ausblick auf mögliche Weiterentwicklungen und aktuelle Forschungsvorhaben zum Thema „Lean“ zu.

2 SAP’s Prozessmanagement-Evolution: Vom Projekt zur Lösung

Seit der Gründung 1972 bietet SAP ein Portfolio unterschiedlicher Unternehmenssoftware rund um das Kernprodukt, die Business Suite (früher R/2, R/3, etc.), an. Viele Bereiche dieser Software wurden ursprünglich entwickelt, um in großen Unternehmen eingesetzt zu werden. Seit einigen Jahren bietet SAP daher auch Softwarelösungen für kleine und mittelständige Betriebe an. Auch dieser Bereich wächst und wird ständig weiterentwickelt.

In den ersten zwei Jahrzehnten ihrer Unternehmensgeschichte entwickelte SAP sehr eng mit ihren meist lokalen Kunden zusammen Unternehmenssoftware. Diese dichte Zusammenarbeit half, den Fokus auf das Wesentliche nicht zu verlieren, sowie Geschäfts- und Technologietrends im Auge zu behalten. Ohne diese gute Zusammenarbeit wäre die Entwicklung eines vollständigen Client-Server-Systems wie SAP R/3, das alle kritischen Geschäftsprozesse unterstützt und in jedem Großindustriezweig einsetzbar ist, nicht so schnell von statten gegangen. Die Entwickler hatten ständigen Kontakt mit Kunden und Benutzern in verschiedensten Aufgabenbereichen. Nicht nur in der Rolle als Programmierer, sondern auch als Berater, Trainer und im Support.

Der Erfolg von R/3 gestaltete den Markt für kommerzielle Softwareanwendungen neu. Für neue Kunden war R/3 nicht mehr ein System, bei dem sie mithalfen, dies zu entwickeln und die Möglichkeit hatten, es mit zu formen. Es war zu einem *Produkt* geworden. Der Kunde wollte es schnell installieren sowie konfigurieren, danach sollte es sofort einsatzbereit sein und einwandfrei funktionieren, auch wenn die Geschäftsprozesse des einzelnen Kunden die eine oder andere Eigenheiten aufwiesen. Anforderungen wandelten sich von mehr Geschäftsfunktionalitäten zu einem höheren Grad der Konfigurierbarkeit. Mit ausgewählten strategischen Kunden wurde auch in der Standardproduktentwicklung kontinuierlich zusammengearbeitet.

Bereits beginnend mit SAP R/2 und letztendlich mit SAP R/3 gelang, zusammen mit weltweit agierenden Kunden, der Aufstieg zum globalen Softwareanbieter. Innerhalb von wenigen Jahren wuchs die SAP Entwicklungsorganisation von ca. 2.000 auf über 14.000 Entwickler an. In der Zeit von R/2 wurden den Entwicklern generalistische Fähigkeiten abverlangt (vgl. oben). Innerhalb der starken Wachstumsphase der Entwicklungsorganisation wurde der Entwickler immer mehr zum Spezialisten einer bestimmten Technologie, eines spezifischen Entwicklungszweiges oder einer bestimmten Anwendungsdomäne. Dies zeigte sich u.a. darin, dass für R/3 bereits eine eigene Beratungsorganisation aufgebaut wurde. Der Anstieg der Spezialisierung korreliert unter anderem mit dem Grad der Arbeitsteilung positiv. Inzwischen war es üblich, dass eine Expertengruppe die Anforderungen erhob und in die Entwicklungsbereiche trug, an anderer Stelle wurden ausführliche Spezifikationen und Architekturentwürfe erstellt, andere Gruppen programmierten die Basisfunktionalitäten und Algorithmen, wieder andere testeten usw. [Sch10].

Im Laufe der 90er Jahre wurde klar, dass der Kommunikationsfluss zwischen den hochspezialisierten Teams, Abteilungen und Organisationen zum Problem geworden war und dies die Hauptursache für Probleme mit der Produktqualität auf verschiedenen Integrationsebenen darstellte. Um diesen Problemen Herr zu werden und die Entwicklungsaufwände über alle beteiligten Organisationseinheiten hinweg koordinieren zu können sowie aufgrund von externen Anforderungen durch bspw. ISO 9000 und der *Food and Drug Administration* (FDA), führte SAP mit Beginn der 90er Jahre ein erstes Entwicklungs-Prozess-Framework ein. Entsprechend ihren Hauptaktivitäten und Qualifikationen wurden Abteilungen Verantwortungen zugewiesen. Es existieren feste Übergabepunkte und Vorschriften zwischen den beteiligten Abteilungen, wodurch das Prozessmodell eher als wasserfallartig zu beschreiben ist. Dieses Prozessmodell wurde von den meisten Entwicklungsabteilungen ein Mal pro Jahr durchlaufen und wurde über die Jahre weiter entwickelt (siehe Abbildung 1).

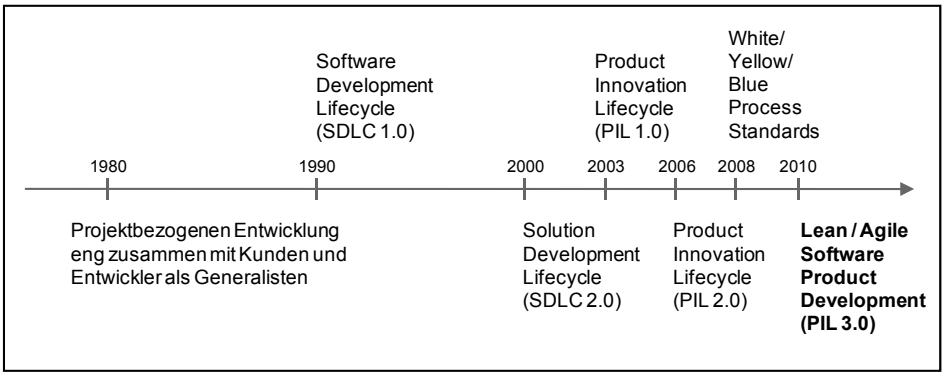

Abbildung 1: Evolution der SAP-Prozessstandards

Als zwischen 2001 und 2004 agile Methoden einen breiteren Einsatz in der Industrie fanden [Boe04; Dyb08; Din10], fing auch SAP an, diese zu pilotieren. Einige Teams begannen bspw. mit Extrem-Programming [Bec99] und Scrum [Sch01] zu experimentieren. Dieser Ansatz war für reguläre Produktentwicklung jedoch nicht einsetzbar, da Entwicklerteams in der Regel keinen direkten Zugriff mehr auf Kunden oder Benutzer hatten, sondern nur über eine Produkt- bzw. Solution- Management-Organisation. Daher wurde agiles Projektmanagement zunächst nur bei Prototypentwicklung mit starker Kundenbeteiligung verwendet und als eigener „Track“ im *Product Innovation Lifecycle* (PIL) aufgenommen (siehe Abbildung 1). Aufgrund unterschiedlicher Produkteigenschaften und daraus resultierenden Projektanforderungen wurden neben den Standard-PIL-Tracks zusätzlich die sogenannten „White/Yellow/Blue-Prozessstandards“ definiert [Sch10].

White-Prozessstandard: für innovative Lösungen, Prototyp- und Forschungsprojekte. Diese Projekte können ihre Entwicklungsmethoden, wie bspw. Scrum oder traditionelles Projektmanagement, und Tools frei wählen.

Yellow-Prozessstandard: für neue Produkte und große Projekterweiterungen. Als Projektmanagement-Methode wird Scrum eingesetzt (agiler PIL-Track, s.o. und Abbildung 1), Anwender und Kunden müssen bei der Entwicklung miteinbezogen werden. Mit den eigentlichen Entwicklungszyklen wird erst begonnen, wenn ein durchkalkulierter Business Case existiert.

Blue-Prozessstandard: kontinuierliche Weiterentwicklung von existierenden Standardprodukten, wie beispielsweise Anpassungen auf Grund gesetzlicher Bestimmungen, kleinere Anpassungen, um neue Geschäftsprozesse zu implementieren etc. Allerdings ist es in solch einem Projekt nicht erlaubt, dem Produkt neue Technologien hinzuzufügen und umfassende Änderungen an der Benutzeroberfläche vorzunehmen. Als Projektmanagement-Methode kommt hier ein eher traditioneller Ansatz zum Einsatz.

Trotz aller bisherigen Prozessstandards und deren Fokus auf Produkte und Lösungen, gelang es jedoch seit 1997 nicht mehr, Skaleneffekte bzw. Umsatzwachstum mit der wachsenden Anzahl an Mitarbeitern und Produkten zu erzielen [OVoJ]. Basierend auf dieser Erkenntnis wurde nach den möglichen Ursachen hierfür gesucht.

Als *Symptome* stellten sich zunächst folgende Punkte heraus: zu dicht aufeinander folgende Veränderungen, Ad-Hoc-Gegenmaßnahmen bei unterschiedlichsten Problemen, Einmischung und Nichtbeachtung von Managemententscheidungen, lange Durchlaufzeiten in der Entwicklung, Produkte, die nicht den Kundenanforderungen entsprechen bzw. bei denen "Over-Engineering" betrieben wird.

Als *Gründe* wurden u.a. fehlende Gesamtverantwortlichkeit für Kundenzufriedenheit, nicht gelebte Prozessstandards sowie ständiges Wechseln zwischen verschiedenen Tätigkeiten identifiziert. Die Mitarbeiter werden durch dieses Multitasking und die daraus resultierenden fortlaufenden Unterbrechungen unverschuldet weniger produktiv.

Als weitere wesentliche *Ursachen* für die Performance-Probleme in der Entwicklung wurden unklare Produktdefinition und –verantwortung, mangelnde Teamfähigkeit („Silo-Denken") und die deutlich schnellere Entwicklung des Markts für betriebliche Standardsoftware erarbeitet.

Diese Analyse stellt die Grundlage für das SAP „Road 2 Lean" Programm dar.

3 Das "Road 2 Lean"-Programm

Lean Management [Lik04] und agile Entwicklungsmethoden [Agi01] bieten für viele der aktuellen Probleme in der Softwarebranche Lösungsansätze (vgl. Abschnitt 2). Wie bei personenorientierten Prozessen üblich, ist auch bei diesen Ansätzen sowohl Pragmatismus als auch Prozessdisziplin von Nöten. Fehlt einerseits der *Pragmatismus*, versinken die Beteiligten schnell im Chaos oder arbeiten sich an erhöhter Bürokratie ab; fehlt andererseits ein gewisses Maß an *Disziplin*, kann im besten Fall ein „kreatives Chaos" erreicht werden (Abbildung 2). Lean Management in Verbindung mit agilen Entwicklungsmethoden für die Softwareerstellung bietet ein gesundes Mittelmaß zwischen Pragmatismus und Disziplin [Sha07; Lar08].

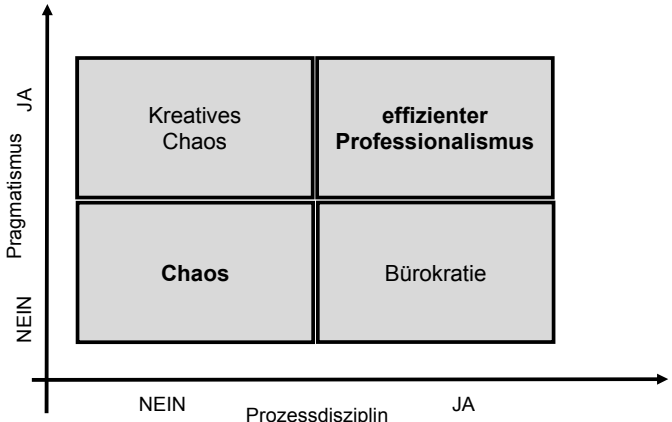

Abbildung 2: Systematik personenorientierter Prozesse

Da Lean nicht gleich agil ist und vice versa, wurden zunächst die Gemeinsamkeiten von Lean und Agilen Methoden untersucht und aufgezeigt (siehe Abbildung 3). Die drei nennenswertesten Gemeinsamkeiten sind das starke Miteinbeziehen des Kunden, der teambasierte Ansatz sowie das Ziel, möglichst hohe Qualität bereits in der ersten Iteration zu erlangen. Darüber hinaus soll die Menge an Anforderungen stets eindeutig in Form eines sog. „Backlogs" priorisiert sein [Sha07].

Aus dem Agilen Manifest [Agi01] geht zudem hervor, dass funktionierende Software als primäres Artefakt aus Kundensicht und zentrales Fortschrittsmaß wichtiger ist als eine umfassende Dokumentation. Darüber hinaus ist eine schnelle Reaktion auf Veränderungen, bspw. von Kundenanforderungen im Projektverlauf, wichtiger als einem festgeschriebenen Projektplan bzw. einer zu Anfang detaillierten Spezifikation zu folgen.

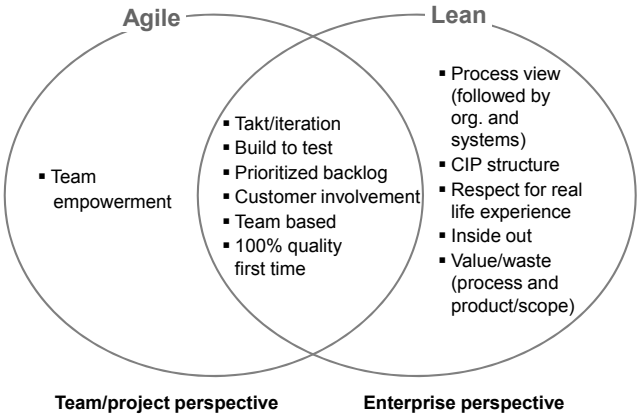

Abbildung 3: Vergleich zwischen Agilen und Lean Prinzipien [Sha07]

Im Unterschied zu traditionellen (plangetriebenen) Ansätzen ist der agile bzw. der Lean-Ansatz nutzengetrieben [Hil06]. Dies zeigt sich u.a. darin, dass sich die Verteilung der fest definierten und variablen Projektparameter des „eisernen Dreiecks“ (Auslieferungstermin, Ressourcenverbrauch und die Menge an Kundenanforderung) grundsätzlich unterscheiden. Während es in der traditionellen Softwareerstellung üblich ist, die Anforderungen, einmal aufgeschrieben, als unveränderlich zu betrachten, Ressourcen und zugesagte Auslieferungsdaten zu ändern, gilt im Lean-Umfeld: Ressourcen und zugesagte Termine sind fix, wohingegen nur die Menge der Anforderungen variabel ist. Allerdings sind alle Anforderungen zueinander eindeutig priorisiert [Sch09], sodass weniger gewichtige in Absprache mit den Kunden gestrichen oder in ein nachfolgendes Release verschoben werden können. Dies erfordert bei allen Beteiligten, sowohl Softwareanbieter als auch Kunden, ein entsprechendes Umdenken aber auch viel Disziplin (siehe Abbildung 2). So sind Kunden beispielsweise im Projektumfeld zunächst ggf. skeptisch und befürchten, nichts oder nur sehr wenig geliefert zu bekommen. Folglich ist ein gegenseitiges Vertrauensverhältnis notwendig, um ein solches Modell in der Praxis umzusetzen und erfolgreich anzuwenden.

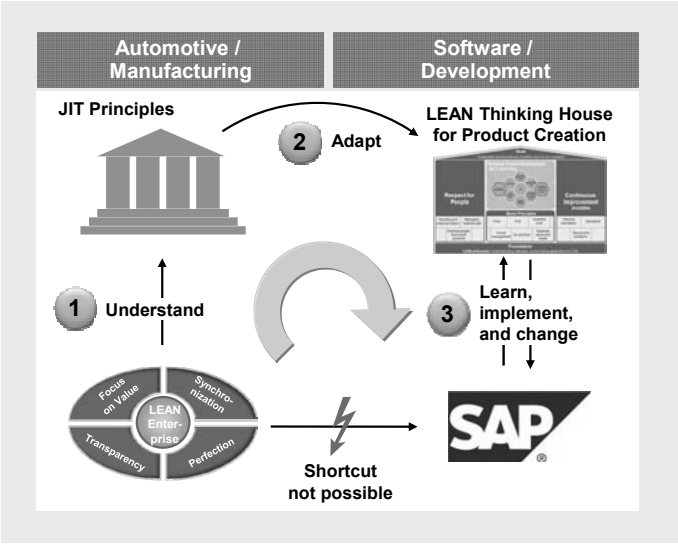

Abbildung 4: Von Lean Manufacturing zu Lean Software Development [SAP09]

Abbildung 4 illustriert den grundlegenden Ansatz, wie „Lean Production“ bzw. die Werte eines „Lean Enterprise“ [Lik04] auf die Softwareindustrie übertragen werden kann. Zunächst war es notwendig, ein einheitliches Verständnis eines „Lean Enterprise“ zu schaffen. Die Hauptbestandteile sind dabei: (a) Konzentration auf nutzenstiftende Tätigkeiten („focus on value“), (b) Synchronisation, (c) Transparenz und (d) Perfektion.

Nach der Identifikation der fundamentalen Probleme und deren Ursachen (siehe auch Abschnitt 2) wurden die etwas konkreteren *Just in Time*-Prinzipien (JIT) aus dem Toyota Produktionssystem (TPS) [Lik04] miteinbezogen. Die JIT-Prinzipien beinhalten „(One-Piece) Flow“ (durchgängige Bearbeitung eines oder weniger Artefakte), „Takt“ (Arbeiten in festen Zeitintervallen), „Pull“ (Entwicklung und Produktion werden primär durch Kundennachfrage gesteuert) und „Zero Defects“ (Fehlervermeidung in Prozess- und Produktqualität).

Aufgrund der Erkenntnis, dass eine direkte Übertragung der Werte und Prinzipien einer „Lean Production“ (wie bei Toyota und Porsche) auf die Softwareindustrie nicht möglich ist (siehe Abbildung 4), wurde als Framework für „Lean Software Product Development“ bei SAP das „Lean Thinking House for Product Creation“ in Anlehnung an Larman und Vodde [Lar08] entwickelt (Abbildung 5). Dies stellt das für SAP adaptierte Verständnis von Lean dar. Auf dieser Grundlage wurden die ersten Lean-Entwicklungsprozesse bei SAP implementiert. Diese Prozesse werden über einen *Countinous Improvement*-Prozess (CIP) ständig weiterentwickelt und optimiert [Rot09].

Das „Lean Thinking House for Product Creation“ vereint Lean- und JIT Prinzipien, erweitert diese um die von SAP definierten Kernelemente, welche zur Umsetzung der Agilen- und Lean- Prinzipien dienen (Abbildung 5). Diese basieren ebenfalls auf der Verknüpfung von Prinzipien aus Lean Management und agiler Softwareentwicklung (siehe Abbildung 3 sowie [Sha07]). Die Kernelemente beinhalten u.a. eine komplette und durchgängige Produktverantwortung (bez. Qualität, Kosten und Auslieferung) in einer Hand, interdisziplinäre Teams zur möglichst eigenständigen Entwicklung einzelner Produkte und Produkteigenschaften („Feature Teams“) [Lar08], ein stets priorisierter Backlog bspw. mit Anforderungen aus Kundensicht (z.B. „User Stories“) [Coh10; Coh04; Gei07] sowie Arbeiten in Takten [Red07] ein potenziell auslieferbares Produktinkrement pro Entwicklungszyklus (bspw. pro Sprint bei Scrum [Sch01; Sch07]).

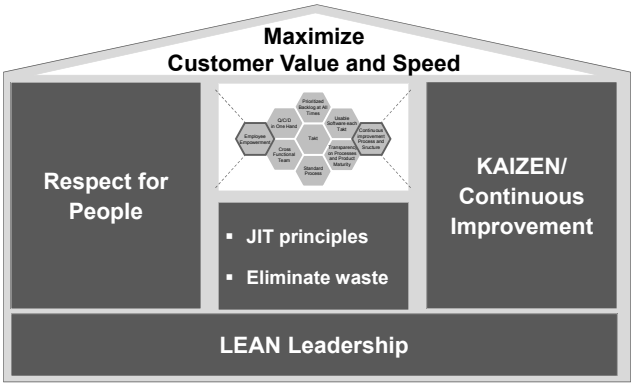

Abbildung 5: LEAN Thinking House (basierend auf [Lar08])

Basierend auf den positiven Erfahrungen mit der Implementierung von Lean und agilen Prinzipien aus mehreren parallelen Pilotprojekten in allen Hauptentwicklungsbereichen wurde Ende 2009 (vgl. Abbildung 6) die Entscheidung getroffen, diese Prinzipien sowie die neun Kernelemente in der gesamten Entwicklungsorganisation umzusetzen, auch über mehrere Vorstandsbereiche hinweg. Abbildung 6 zeigt die Entwicklung der vom „Road2Lean"-Programm betroffenen Mitarbeiter in der Entwicklungsorganisation. Bei dieser laufenden Veränderung liegt derzeit der Fokus auf dem Aspekt „Qualität", mit der Frage wie nach der Umstellung auf kurze Entwicklungszyklen (Takte, Sprints) weiterhin sehr gute Qualität geliefert werden kann. Hierzu wird aus der agilen Welt *Test-Driven Development* (TDD) eingesetzt [Bec03; Fre09].

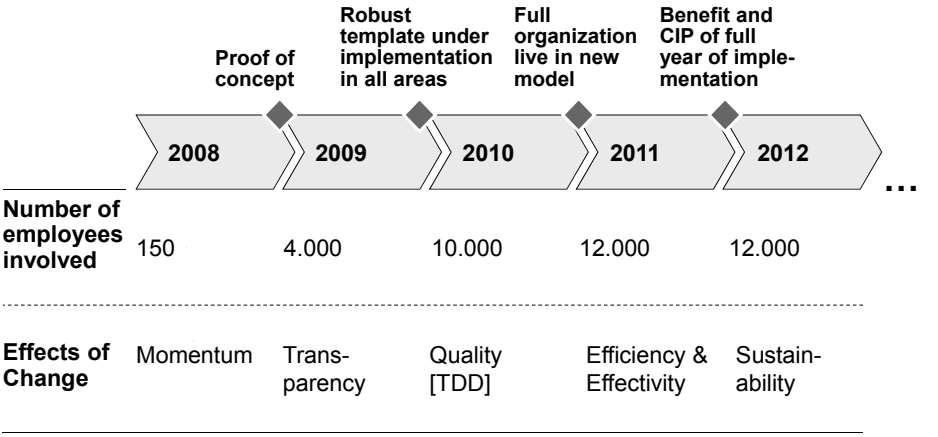

Abbildung 6: Entwicklung des Road2Lean-Pilotprogramms

Ähnlich wie in den Anfangszeiten von SAP (siehe 80er und 90er Jahre in Abbildung 1) wird neben der Standardproduktentwicklung im Bereich „Custom Development" wie bisher sehr kundennah und projektbezogen gearbeitet. Auch im Custom Development ist eine wachsende Anzahl von agilen Projekten, beispielsweise mit Scrum als Projektmanagement-Methode, zu verzeichnen. In der kundenbezogenen Entwicklung kommt es ebenso auf das gegenseitige Vertrauen der beteiligten Partner und zusätzlich auf die gesetzlich vorgeschriebenen Anforderungen des Kunden bezüglich der Vollständigkeit der Spezifikation (Lasten-/Pflichtenheft) zu Projektbeginn an.

4 Fazit

Die Erfahrungen der Vergangenheit, die Parallelen in vergleichbaren Unternehmen und das Feedback aus den heutigen Entwicklungsbereichen deuten darauf hin, dass mit Lean Management und agilen Prinzipien einige der grundlegenden Probleme, wie beispielsweise die direkte Orientierung an Kundenanforderungen, adressiert werden konnten (vgl. Ende Abschnitt 2). Der heutige Prozessmanagement-Ansatz, basierend auf den Lean-Prinzipien, beinhaltet auf der einen Seite Scrum als „Projektmanagement-Methode", andererseits wird hierdurch nur ein grober, aber fester Rahmen für die eigenständige Entwicklung von Produkten durch Team-Strukturen vorgegeben (siehe Abbildung 2). Entsprechend der agilen Prinzipien (u.a. die Betonung auf lauffähiger Software und die Möglichkeit, flexibel auf sich ändernde Kundenanforderungen zu reagieren, siehe Abschnitt 3 und Abbildung 3) liegt der Fokus klar auf dem *Produkt*, das am Kunden ausgerichtet ist. Grundsätzlich wird versucht, die Tugenden der Vergangenheit wieder zu beleben und gleichzeitig die Fehler von damals im aktuellen Kontext nicht zu replizieren.

Die Herausforderung dabei besteht darin, agile Ansätze wie Scrum, über die Teamebene hinaus auf verschiedene Entwicklungsbereiche oder gar auf ein komplettes Softwareunternehmen zu skalieren [Lef07; Lar08; Sch10]. Hierbei helfen die Erkenntnisse aus dem Lean Management-Umfeld sowie neuere Arbeiten zum Thema „Lean Product Development" [Rei09].

5 Ausblick

Parallel zur aktuellen Implementierung der neun Kernelemente (siehe Abschnitt 3) wurde ein Forschungsprojekt im Rahmen der BMBF[1]-Initiative „Softwarecluster bzw. Spitzencluster für Prozessinnovation in der Softwareentwicklung" mit einem Zeitrahmen von insgesamt drei Jahren bis 2013 ins Leben gerufen. Dieser Forschungsverbund setzt sich aus Unternehmen der Softwareindustrie sowie einschlägigen Forschungseinrichtungen und Universitäten zusammen.

Ziel dieses Forschungsvorhabens ist parallel zur operativen Einführung von Lean Management und agilen Methoden in der Softwareindustrie die kausalen Wirkungszusammenhänge innerhalb und zwischen Teams [Lee10] sowie den Prozessfluss in einem „*Lean (Software) Product Development*"-System [Rei09] besser zu verstehen, zu erklären und letztendlich auch Erkenntnisse zu dessen Verbesserung zu erlangen, von der wiederum die Implementierung in der Entwicklung profitieren kann.

[1] BMBF = Bundesministerium" für Bildung und Forschung

Ein Unter-Cluster mit Beteiligung unterschiedlicher SAP-Entwicklungsbereiche, SAP Research und der Universität Mannheim hat basierend auf aktuellen Forschungsfragen drei Blöcke von Fokusthemen identifiziert: (1) Erklärung der Wertschöpfung von Implementierung- und Produktteams, (2) Analyse und Optimierung des Informationsflusses zwischen den Teams, d.h. im Gesamtentwicklungssystem entlang der Wertschöpfungskette sowie (3) Erklärung und Verbesserung des Lern- und Innovationsverhaltens von Teams und der Organisation als Ganzes.

Fokusthema 1 (Team-Ebene) soll Einflussfaktoren wie bspw. die Autonomie und Diversität von Teams auf Output-Größen wie Lieferung von Funktionalität innerhalb von Zeit- und Budgetrestriktionen erheben und Rückschlüsse auf Erfolgsfaktoren ermöglichen [Lee10]. Komplementär dazu will Fokusthema 2 (teamübergreifende Ebene) den Informationsfluss von Portfolio-Entscheidungen hin zu Backlog Items unterschiedlicher Granularität analysieren, Abhängigkeiten [Hil08] und Warteschlangenverhalten [Lar08] in diesem Prozess aufdecken und damit Aussagen zur Optimierung der Durchlaufzeit einzelner Produkt-Features herleiten [Rei09]. Methodisch analog zu Thema 1 will Fokusthema 3 (Lernverhalten) das Lern- und Innovationsverhalten von Teams und der Entwicklungsorganisation erklären [Ham08; Edm07; Sav09].

Neben diesen Fokusthemen werden im Kontext des BMBF-Forschungsprojekts weitere Forschungsfragen wie beispielsweise die sozialwissenschaftlichen Hintergründe agiler Methoden sowie die Auswirkungen der Veränderungen durch Lean und agiler Entwicklung auf die Mitarbeiterzufriedenheit und die wahrgenommene Arbeitsbelastung betrachtet.

Außerhalb des Clusters, jedoch auch im Lean-Kontext und von SAP unterstützt, entsteht in Zusammenarbeit mit der Universität Stuttgart eine Forschungsarbeit, die sich mit der Fragestellung beschäftigt, wie ein flexibler und schlanker Prozess für die Entwicklung von Softwareprodukten, beispielsweise die Skalierung von Scrum, bei großen Softwareherstellern aussehen sollte [Sch10].

Danksagungen

Wir danken den Kollegen Anton Dillinger, Martin Fassunge, Rainer Schmidtke und Joachim Schnitter für ihre Kommentare und Anmerkungen.

Literaturverzeichnis

[Agi01] Agile Alliance, 2001: Agile Manifesto. http://agilemanifesto.org/

[Bec99] Beck, K. 1999. Embracing change with extreme programming. Computer, Jg. 32, H. 10, S. 70–77. http://dx.doi.org/10.1109/2.796139

[Bec03] Beck, K. 2003. Test-Driven Development: By Example. Addison-Wesley

[Boe04] Boehm, B. W., Turner, R. 2003. Balancing agility and discipline: a guide for the perplexed, Addison-Wesley

[Coh04] Cohn, M. 2004. User Stories Applied: For Agile Software Development. Addison-Wesley Professional

[Coh10] Cohn, M. 2010. User stories: für die agile Software-Entwicklung mit Scrum, XP u.a. mitp

[Din10] Dingsoyr, T., Dybå, T., Brede M., N. 2010. Agile Software Development: Current Research and Future Directions. Springer

[Dyb08] Dyba, T., Dingsoyr, T. 2008. Empirical studies of agile software development: A systematic review. Information and Software Technology, Volume 50, Issues 9-10, August, pp. 833-859

[Edm07] Edmondson, A. C., Dillon, J. R., Roloff, K. 2007. Three perspectives on team learning: Outcome improvement, task mastery, and group process. Walsh, J. P., Brief, A.P. (Eds.), The Academy of Management annals, Hillsdale, NJ: Psychology Press, pp. 269-314

[Fre09] Freeman, S., Pryce, Nat, 2009. Growing Object-Oriented Software, Guided by Tests. Addison-Wesley Professional

[Gei07] Geisser, M., Heinzl, A., Hildenbrand, T., Rothlauf, F. 2007. Verteiltes, internetbasiertes Requirements-Engineering. WIRTSCHAFTISINFORMATIK, Volume 49 (3), pp 199-207

[Ham08] Hamilton, P. 2008. Wege aus der Softwarekrise: Verbesserungen bei der Softwareentwicklung. Springer

[Hil06] Hildenbrand, T., Geisser, M., Nospers, M. 2006. Die Übertragbarkeit der Open Source-Entwicklungsmethodik in die Unternehmenspraxis. Softwaretechnik-Trends, Volume 26 (1), pp 37-42

[Hil08] Hildenbrand, T. 2008. Improving Traceability in Distributed Collaborative Software Development – A Design Science Approach. Peter Lang Publishing Group

[Lar08] Larman, C., Vodde, B. 2009. Scaling lean & agile development: thinking and organizational tools for large-scale Scrum. Addison-Wesley

[Lee10] Lee, G. and Xia, W. 2010. Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data. MIS Quarterly, (34: 1), pp.87-114.

[Lef07] Leffingwell, D. 2007. Scaling software agility: best practices for large enterprises. Addison-Wesley

[Lik04] Liker, J. K. 2004. The Toyota Way. McGraw-Hill Professional

[Red07] Redmiles, D., van der Hoek, A., Al-Ani, B., Hildenbrand, T., Quirk, S., Sarma, A., Silveira, S., Filho, R., de Souza, C., Trainer, E. 2007). Continuous Coordination: A New Paradigm to Support Globally Distributed Software Development Projects. WIRTSCHAFTSINFORMATIK, Volume 49, pp. S28-S38

[Rei09] Reinertsen, D. G. 2009. The Principles of Product Development Flow: Second Generation Lean Product Development. Celeritas Publishing

[Rot09] Rother, M. 2009. Toyota Kata: Managing People for Improvement, Adaptiveness and Superior Results. McGraw Hill Professional

[Sav09] Savelsbergh, C. M. J. H., van der Heijden, B. I. J. M., Poell, R. F. 2009. The Development and Empirical Validation of a Multidimensional Measurement Instrument for Team Learning Behaviors. Small Group Research: 40, pp. 578-607

[Sch10] Schnitter, J., Mackert, O. 2010. Introducing Agile Software Development at SAP AG – Change Procedures and Observations in a Global Software Company. Proc. ENASE 2010, Athen, pp.132-138
[Sch01] Schwaber, K., Beedle, M. 2001. Agile Software Development with Scrum. Prentice Hall
[Sch07] Schwaber, K. 2007. The enterprise and Scrum. Microsoft Press
[Sch09] Scheibmayr, S., Hildenbrand, T., Geisser, M. 2009. An Experimental, Tool-Based Evaluation of Requirements Prioritization Techniques in Distributed Settings. Proceedings of the Fifteenth Americas Conference on Information Systems (AMCIS), San Francisco, California
[Sha07] Shalloway, A. 2007. Jack be Agile, Jack be Lean. BETTER SOFTWARE June 2007. P.32
[SAP09] SAP AG, 2009. interne Schulungsunterlagen