Oliver Röhrsheim
(o.roehrsheim@de.ibm.com)
ist zertifizierter Senior IT Architect, IBM Softwaregroup Cloud & Smarter Infrastructure, PMP®. Seine Aufgabenschwerpunkte liegen in den Bereichen IT-Architektur, Softwareentwicklungsprozesse, CLM und der technischen Projektleitung.

Wesentliche Rahmenbedingungen bei der Einführung agiler Entwicklungsmethoden

„Agilität" ist bei Softwareentwicklungsunternehmen bereits in aller Munde. Die viel propagierten positiven Aspekte, wie kürzere Releasezyklen, höhere Qualität, sich selbst verwaltende Teams, usw. versprechen interessantes Verbesserungspotenzial bei der Kosten-Nutzen-Darstellung der Entwicklungsprozesse. Die Einführung solcher agilen Methoden stellt für die meisten Teammitglieder jedoch eine nicht zu vernachlässigende Herausforderung dar. Gerade bei bestehenden Entwicklungsprozessen, die sich über die Jahre hinweg bewährt haben und die historisch gewachsen sind, stehen Projektleiter und leitende IT-Architekten oft vor der Herausforderung, die Notwendigkeit bzw. Vorzüge geeigneter agiler Aspekte zu bewerten und initial in den bestehenden Prozess einzugliedern. Heutige Entwicklungsunternehmen (meist Softwareentwicklung) beschäftigen sich zunehmend mehr mit der Einführung agiler Vorgehensweisen; nicht zuletzt, um einige der viel propagierten Vorzüge, die mit der Einführung agiler Methoden angepriesen werden, für ihr Projekt zu nutzen. Die Einführung agiler Methoden geschieht dabei in der Regel sehr selten „auf der grünen Wiese" – meist sind bereits entsprechende Entwicklungsprozesse etabliert. Bei der Einführung agiler Methoden gilt es nun, wichtige Rahmenbedingungen zu berücksichtigen und die entsprechend auf die Projektsituation angepassten Entscheidungen zu treffen: beispielsweise Release-Dauer, Teamgröße (insbesondere bei global verteilten Teams), Entwicklungswerkzeuge, usw. Dieser Artikel beschreibt die wesentlichen Aspekte, die es dabei zu beachten gilt, und zeigt einige Lösungsmöglichkeiten auf.

Agility – reines Modethema oder doch mehr?

Vor mehr als zehn Jahren wurde das Agile Manifest ins Leben gerufen. Die damaligen Gründungsmitglieder sahen die Notwendigkeit für drastisch reduzierte Entwicklungszyklen (damalige Produktentwicklungszyklen von 1-2 Jahren waren keine Seltenheit), für eine wesentlich stärkere Integration der Kunden während der Entwicklungsphase sowie für schlankere, flexiblere und weniger aufwendige Entwicklungsprozesse. Andererseits bestand für Softwareentwicklungsunternehmen bereits immer die Notwendigkeit, den eigenen Entwicklungsprozess fortlaufend zu optimieren.

Seit dieser Zeit beschäftigen sich mehr und mehr Unternehmen mit agilen Prozessen, um den eigenen Zielen einer bestmöglichen Kundenzufriedenheit sowie schnelleren, flexibleren und weniger bürokratischen Entwicklungsprozessen Rechnung zu tragen. Diese Notwendigkeit der fortlaufenden Optimierung und Ausrichtung

Abb. 1: Modethemen oder doch mehr?

der Prozesse an Kundenbedürfnissen zeigt, dass es sich bei Agilität um mehr als ein reines Modethema handelt (vgl. Abbildung 1).

Stellung agiler Vorgehensweisen im Softwareentwicklungs-Lebenszyklus

Agile Ansätze beschränken sich typischerweise auf einen Ausschnitt des gesamten Softwareentwicklungs-Lebenszyklus und beschreiben die Erstellung von Software primär aus der Perspektive der Entwickler. Erst neuere Betrachtungsweisen wie beispielsweise „Agility@Scale“ oder „Disciplined Agile Delivery“ übertragen den Gedanken der Agilität auf eine weitere Abstraktionsstufe, die den gesamten Produktlebenszyklus beinhaltet (siehe [AaL12]).

Die herkömmliche Sichtweise auf agile Methodik betrachtet auch die Erarbeitung der Anforderungen und die zugehörige Kommunikation zwischen Fachexperten und Entwicklern zur Sicherstellung eines gemeinsamen Verständnisses der gewünschten Lösung. Im Unterschied zu anderen Vorgehensweisen wird hier der Nutzen direkter Kommunikation und früher Erstellung von funktional unvollständigen, aber bereits ausführbaren und wertbringenden Lösungsversionen hervorgehoben.

Der Fokus agiler Vorgehensweisen endet in der Regel mit der Bereitstellung einer vom Auftraggeber für gut befundenen Softwareeinheit zur Überführung in die Produktion. Aus der umfassenden Landschaft der IT-Prozesse eines Unternehmens konzentrieren sich die agilen Ansätze auf den Bereich der direkten Transformation von fachlichen Anforderungen in auslieferbare Software. Dabei soll das Risiko der Erstellung einer am Bedarf der Benutzer vorbei entwickelten Software minimiert und der mit den verfügbaren Ressourcen erzielbare geschäftliche Nutzen maximiert werden.

Das Erarbeiten der (potenziell) auslieferbaren Softwarestände ist bei den agilen Vorgehensweisen teilweise strikt geregelt, wie man etwa bei Scrum in der Organisation der Sprints erkennt. Da hierzu auch die Verteilung einzelner Arbeitspakete an Teammitglieder sowie die Feststellung des erzielten Fortschritts gehören, stellt dies das „innere Projektmanagement“ dar. Im Gegensatz zum klassischen Projektmanagement wird dieses „innere Projektmanagement“ vom Team und nicht von einer dedizierten Projektleiterrolle betrieben.

Anforderungsmanagement: Vorab Planung vs. iterativ inkrementell

Bei der Mehrheit der nach herkömmlichen Prozessen durchgeführten Softwareprojekte wird von Beginn an festgelegt, welche Anforderungen in welchem Zeitraum mit welchem Umfang an Ressourcen realisiert werden sollen. Die Anforderungen selbst werden meist sehr feingranular verfasst und während des Projektverlaufs gar nicht oder mittels eines etablierten Change-Management-Prozesses geändert. Dies entspricht nicht gerade einem flexiblen Reagieren auf geänderte oder neue Kundenanforderungen mit möglichst geringem bürokratischem Aufwand. Es widerspricht folglich dem agilen Ansatz.

Bei einer agilen Entwicklungsmethode werden zu Beginn grobgranulare Anforderungen erfasst und priorisiert. Die Anforderungen mit der höchsten Priorität werden anschließend bis zu einem solchen Detaillierungsgrad verfeinert, dass der Aufwand für die zugehörigen Tasks und Aktivitäten gut geschätzt werden kann (entweder in Personentagen oder Story-Points).

In einer ersten Iteration werden nun durch Abarbeitung dieser feingranularen

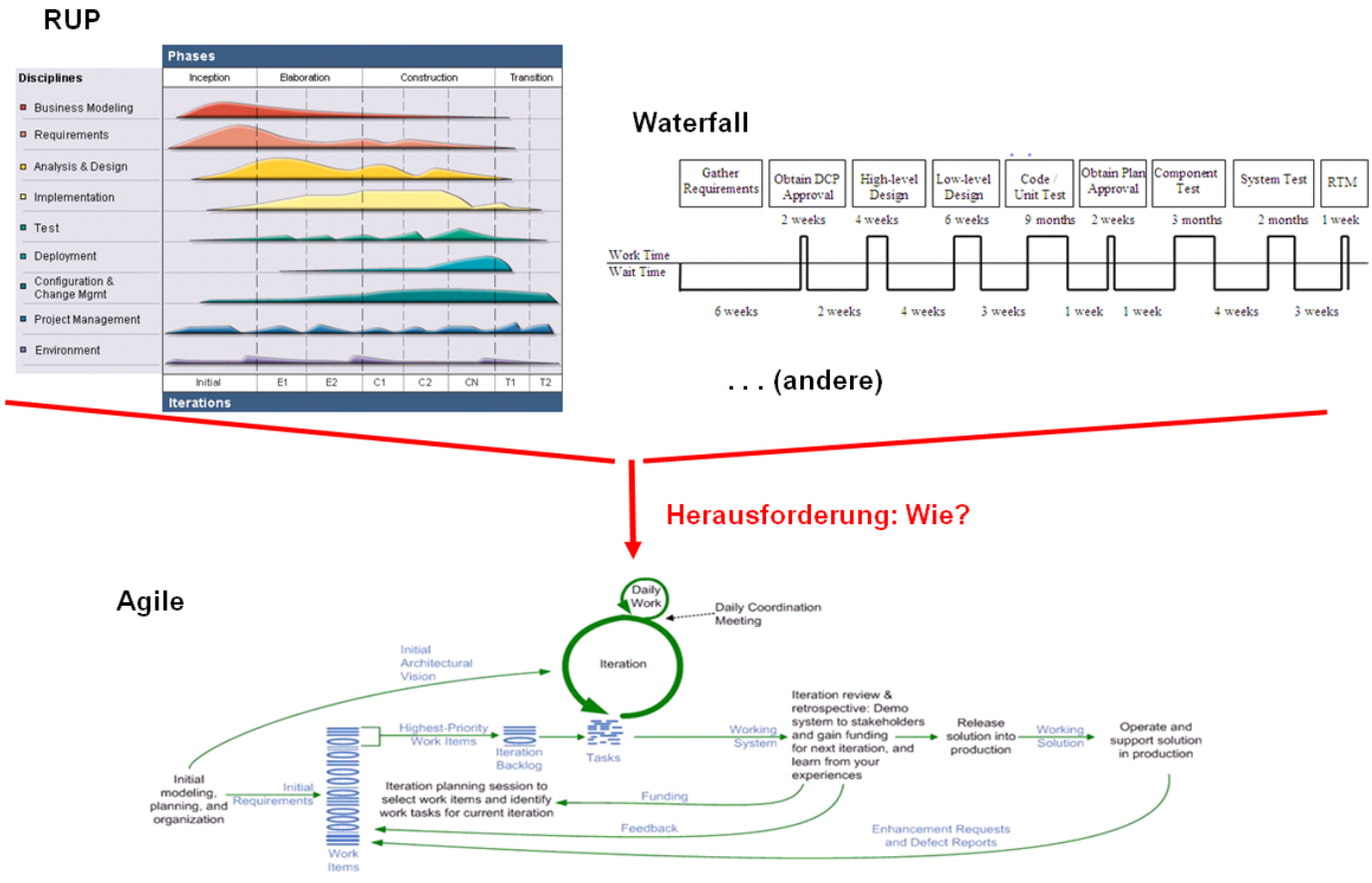

Abb. 2: Iterativ-inkrementelle Entwicklung nach Agile

Tasks die zugehörigen Anforderungen realisiert, getestet und dokumentiert. Am Ende der Iteration kann der Kunde durch Sichtung der erstellten Artefakte eine Vorabnahme durchführen oder aber Änderungswünsche anbringen. Mit den in dieser Iteration gewonnenen Erkenntnissen werden die verbleibenden Anforderungen verifiziert und bei Bedarf überarbeitet sowie neu priorisiert. Es können neue Anforderungen hinzukommen sowie gestrichen werden.

Jede nachfolgende Iteration erfolgt gemäß dem Vorgehen: Detaillierung der höchst priorisierten Anforderungen, Realisierung, Sichtung, Re-Priorisierung der verbleibenden Anforderungen. Dies geschieht bis alle Anforderungen realisiert wurden. Eine solche agile Vorgehensweise erhöht die Integration mit dem Kunden und erlaubt eine kontinuierliche Ausrichtung der Entwicklung basierend auf den bisher gewonnenen Erkenntnissen sowie etwaigen Änderungen im Anforderungsportfolio. Es ist Aufgabe des Produkt- und Projektmanagements bzw. des verantwortlichen Bereichsleiters, eine entsprechende Umgestaltung des Entwicklungsprozesses vorzunehmen (**vgl. Abbildung 2**).

Berücksichtigung existierender Entwicklungsprozesse

Die Firmenkultur beziehungsweise im Unternehmen geltende Prozesse und Berichtsstrukturen beeinflussen die Etablierung von Softwareentwicklungsprozessen. Viele der beispielsweise börsennotierten Unternehmen besitzen eine quartalsgetriebene Kultur. Mittel- und langfristige Zielvereinbarungen bestimmen den Ablauf und das Reporting. Existierende Entwicklungsprozesse – insbesondere dem Wasserfallmodell ähnliche Prozesse – können eine nicht zu unterschätzende Hürde bei der Einführung agiler Methoden darstellen, da diese oft konträre Ansätze verfolgen.

Wie bereits im oberen Abschnitt *„Anforderungsmanagement: Vorab Planung vs. iterativ inkrementell“* beschrieben, muss eine Änderungen der Entwicklungsprozesse im Hinblick auf das Anforderungsmanagement und die Projektplanung erfolgen. Herkömmliche Entwicklungsprozesse weisen oft wasserfallartige Planungsmodelle und -zyklen auf. Meilensteine wie *„Anforderungen definiert“*, *„Design abgeschlossen“*, usw. müssen bei der Einführung agiler Entwicklungsmethoden durch iterativ, inkrementelle Checkpoints ersetzt werden.

Darüber hinaus ändert sich das Verantwortungsfeld einiger, an einer Entwicklung beteiligten Rollen. Das Produktmanagement beispielsweise muss die Zusammenarbeit mit Kunden intensivieren und in Teilbereichen sogar ändern. Der Kunde muss idealerweise integraler Bestandteil des Entwicklungszyklus werden und die jeweils am Ende einer Iteration erstellten Software-Artefakte sichten und bewerten (→Stakeholder-Demo).

Zu diesem Zweck ist es Aufgabe des Produktmanagements, den Kunden davon zu überzeugen, den notwendigen Aufwand seinerseits zu investieren. Es muss dem Kunden deutlich erkennbar sein, dass dieser zu erbringende Aufwand einen Mehrwert darstellt. Projektleiter und leitende IT-Architekten müssen mit einer dynamischen Menge von Anforderungen während des Entwicklungszyklus umgehen können. Sie müssen darauf achten, dass der Detaillierungsgrad der in der nächsten Iteration zu realisierenden Anforderungen feingranular genug ist, um den dahinterstehenden Aufwand zu schätzen. Darüber hinaus müssen sie die Kommunikation innerhalb des Teams mehr denn je forcieren, da ein Prozess agiler Entwicklung enge Teamintegration und häufige Kommunikation erfordert. Im Extremfall geht dies hin bis zu täglichen, jedoch kurzen, Teambesprechungen – den sogenannten Scrum-Meetings.

Integration in unternehmensweite Prozesse

Neben dem „inneren“ Projektmanagement sehen die Controlling-Mechanismen größerer Unternehmen stets ein formales, weniger inhaltlich getriebenes „äußeres“ Projektmanagement vor. Dort werden u. a. verbrauchter und verbliebener Aufwand, der Finanzstatus und der Ampelstatus in Bezug auf Plantreue des Projekts gemeldet und vor entsprechenden Gremien vertreten. Zu diesem äußeren Projektmanagement zählen ebenso die meisten Tätigkeiten der Projektinitialisierung, wie z. B. das Einrichten von Buchungspositionen zur Erfassung von geleistetem Aufwand.

Die Einbettung des agilen Projekts in die Projektsteuerung des Unternehmens kann hier sehr einfach erfolgen: Ein Projektleiter, der nicht Mitglied des agilen Teams ist, übernimmt den äußeren Teil des Projektmanagements; er wird dabei regelmäßig mit ohnehin anfallenden Statusinformationen – z. B. die Ergebnisse des Sprint-Review-Meetings, Burndown-Charts – und/oder bei Tooleinsatz durch automatisierte Reports versorgt, sodass er die notwendigen Berichte erstellen und den Fortschritt darstellen kann. Seine Sicht auf das agile Projekt ist der Projektplan mit den geplanten Meilensteinen, synchronisiert mit den Sprints des agilen Projekts. Die Verteilung der Aufgaben und die Gestaltung der Vorgehensweise innerhalb des Projekts verbleibt bei dem Team – der agile Kern bleibt unberührt.

Eine weitere Schnittstelle zu unternehmensweiten Prozessen ergibt sich durch das übergreifende Architekturmanagement. Dort werden die für die IT geltenden Standards, z. B. Basistechnologien, Produkte, High-Level-Architekturen, definiert und deren Einhaltung in Projekten und anderen IT-Maßnahmen geprüft.

Häufig wird dieser Eingriff in die „technische Autonomie“ von Projekten gerade von agilen Entwicklungsteams sehr kritisch gesehen, fühlen sie sich doch ganz der bestmöglichen Lösung aus Sicht ihres Kunden verpflichtet. Dies muss nicht in jedem Fall mit der Technologieauswahl, welche die Unternehmensarchitektur auf der Basis berechtigter anderer Kriterien trifft, übereinstimmen.

Um hier möglicherweise auftretende Konflikte frühzeitig zu erkennen und zu lösen, ist die Festlegung auf eine Basisarchitektur zu Beginn des Projekts empfehlenswert, am besten unter direkter aktiver Beteiligung eines Unternehmensarchitekten. Beim Scrum-Vorgehen kann man eine kurze Phase mit wenigen, schnellen Explorationssprints („Architektursprints“) vornehmen, deren primäres Ergebnis weniger direkt für den Benutzer des Systems nutzbar ist, welches aber die technische Basis der Lösung festlegt.

Dazu können aus den initialen fachlichen Anforderungen technische Stories abgeleitet werden, deren Umsetzung in Form (evolutionärer) Prototypen die Auswahl und Tauglichkeit einer technischen Architektur nachweisen. Der beteiligte Unternehmensarchitekt soll dabei die Konformität zur Unternehmensarchitektur sicherstellen und ggf. für eine Erweiterung bestehender Standards und Richtlinien sorgen.

Eine gesonderte, nachgelagerte Abnahme der Basisarchitektur kann bei dieser Arbeitsweise entfallen. Gegebenenfalls erforderliche Änderungen der Basisarchitektur im Projektverlauf sollten immer direkt mit der Unternehmensarchitektur abge-

stimmt werden, dann ist die Bestätigung der Architekturkonformität des Projektergebnisses durch die Unternehmensarchitektur nur noch eine Formsache. Auf diese Weise kann die Einbeziehung der Unternehmensarchitektur in den agilen Entwicklungsprozess in organischer Weise erfolgen, ohne den agilen Charakter des Projekts zu verfremden.

Organisationsstrukturen

Ein agiler Entwicklungsprozess erfordert eine enge Teamintegration und offene, häufige Kommunikation, gemeinsame Ziele sowie ein klares, gemeinsames Verständnis des Mehrwerts einer jeden Anforderung aus Sicht des Kunden. Je mehr Barrieren zwischen Projektleitern, Entwicklern, Kunden, Managern und anderen Beteiligten bestehen, desto schwerer ist es, diese notwendige Basis für agile Softwareentwicklung zu schaffen.

Eine dieser Barrieren kann in der Organisationsstruktur des Entwicklungsbereiches liegen. Das Management ist hier gefordert, maximale Transparenz und Integration zwischen den Teams zu ermöglichen. Besteht ein Team aus Mitgliedern unterschiedlicher Organisationen, so muss der Leiter des Teams für die Dauer des Projektes bzw. der Iteration klar und deutlich die Weisungsbefugnis für alle Teammitglieder erhalten. Diese übertragene Autorität gestattet es dem Teamleiter teaminterne Konflikte ohne Eskalation über das Management und damit wesentlich effizienter zu lösen. Das Team muss als Einheit ohne unnötige, externe Abhängigkeiten agieren können.

Eine weitere wichtige Rolle bei der Zusammenstellung von agil arbeitenden Teams bildet der ausgewogene Mix an notwendigen Skills innerhalb des Teams. Agil arbeitende Teams sollen in der Lage sein, spezifische Anforderungen autark im Team zu realisieren. Dies umfasst Aktivitäten wie Architektur, Design, Implementierung, Test und Dokumentation. Damit das Team dazu in der Lage ist, sollte es idealerweise aus Mitgliedern mit jeweils unterschiedlichen Skill-Schwerpunkten zusammengesetzt sein.

Dies ist erneut konträr zu den meisten bisher existierenden Teamansätzen, da Teams bisher eher auf Basis von Skill-Schwerpunkten definiert wurden. Zur Realisierung eines agilen Ansatzes ist demzufolge eine Team-Restrukturierung erforderlich. Auch hier ist das jeweilige Management gefordert, die neuen Teambildungsansätze durchzusetzen oder zumindest zu ermöglichen.

Ein weiterer wichtiger Punkt bei der Team-Zusammenstellung kann die Zuweisung eines sogenannten „Agile Moderators“ sein. Gerade bei Entwicklungsorganisationen, die bisher keine Erfahrung im Bereich agiler Entwicklungsprozesse gesammelt haben, ist eine solche unterstützende Funktion als Bestandteil eines jeden Teams sinnvoll.

Ein solcher „Agile Moderator“ sollte über ausreichend praktische Erfahrung im Bereich agiler Entwicklungsprozesse verfügen. Seine Hauptaufgabe ist es, das Team während der verschiedenen Iterationen zu coachen und die Beachtung der Grundprinzipien des Agilen Manifests zu einem gewissen Grad sicherzustellen. Der Projektleiter sowie der leitende IT-Architekt sind hier gefordert, das Management davon zu überzeugen, den notwendigen Mehraufwand für den Einsatz eines solchen Moderators je Team zu gewähren. ■

Literatur

[AgM] The Agile Manifest, http://agilemanifesto.org/

[Chi04] Gary Chin, „Agile Project Management – How to succeed in the face of Changing Project Requirements”, Mcgraw-Hill Professional, 2004

[Coh08] Mike Cohn, „Agile Estimating and Planning“, Prentice Hall, 2008

[AaL12] Scott Ambler, Mark Lines, „Disciplined Agile Delivery: A Practitioner‘s Guide to Agile Software Delivery in the Enterprise”, IBM Press, 2012

[Pic08] Roman Pichler, „Scrum – Agiles Projektmanagement erfolgreich einsetzen“, dpunkt.verlag, 2008

[Pre01] Roger S. Pressman, „Software Engineering – A Practitioner’s Approach“, 5th Edition, 2001, http://catalogs.mhhe.com/mhhe/home.do

[Sch04] Ken Schwaber, “Agile Project Management with Scrum”, Microsoft Press, 2004