Dogmatisches "Entweder agil oder klassisch" im Projektmanagement hat ausgedient – die richtige Mischung macht's

Dr. Michael Kirchhof
Dr. Michael Kirchhof ist Unternehmensberater bei der mm1 Consulting & Management PartG. Studium und Promotion absolvierte er an der RWTH-Aachen mit Vertiefung Software-Engineering und durchlief verschiedene berufliche Positionen und unternehmerische Tätigkeiten. Er berät und unterstützt insbesondere Konzern-Unternehmen in Deutschland und Österreich bei der langfristigen Ausrichtung und Umsetzung innovativer Technologie-Vorhaben und der Organisationsentwicklung. Er ist als Associate Partner Mitglied der mm1-Geschäftsleitung und verantwortet den Unternehmensbereich Schlankes IT-Management und Technologie-Transformation.

Prof. Dr. Bodo Kraft
Nach Studium und Promotion an der RWTH-Aachen mit Vertiefung Software-Engineering arbeitete Dr. Bodo Kraft bei der Generali Deutschland Informatik Services als IT-Manager in verschiedenen Positionen. Er war dort u.a. verantwortlich für die Projektmanagement-Methodik im Rahmen der Systementwicklung und den Aufbau der Konzern-Projektleiter-Community. Dr. Bodo Kraft ist an der FH Aachen für das Lehrgebiet Wirtschaftsinformatik berufen. Ein Schwerpunkt seiner Lehr- und Forschungsaktivitäten liegt im Bereich Prozessmodellierung und Projektmanagement.

1 Einleitung

Agile Entwicklungs- und Projektmanagementmethoden gewinnen an Bedeutung. Viele Unternehmen denken, nach aufwändigen Einführungen von klassischen Entwicklungsmodellen wie z. B. RUP oder PRINCE, verstärkt über den Einsatz agiler Methoden nach. Die Gründe hierfür sind vielfältig. Sie liegen einerseits in dem Wunsch nach schnelleren und flexibleren Steuerungsmechanismen begründet. Andererseits schrecken der Aufwand der zu erstellenden Dokumentation und die transparente Planung aller benötigten Aufgaben und Rollen viele Projektbeteiligte auf unterschiedlichen Ebenen ab.

In der Tat bieten agile Methoden gute Möglichkeiten, Projekte effizienter zu steuern und flexibler auf Anforderungen der Stakeholder zu reagieren. Eine Produktneuentwicklung mit vielen Unklarheiten auf Auftraggeberseite erfordert beispielsweise geradezu ein agiles Vorgehen. Aber auch klassische Methoden haben weiterhin ihre Berechtigung und ermöglichen – ein sorgfältiges Tailoring vorausgesetzt – effiziente Projekte. So bieten sich zum Beispiel für kritische Migrationsprojekte, bei denen der Kundenauftrag in Bezug auf Termin und Umfang genau und endgültig abgegrenzt ist und zudem ein hohes Maß an formaler Dokumentation erforderlich wird, klassische Steuerungsmechanismen geradezu an.

Die Herausforderung beim Zuschnitt neuer Projekte und bei der Festlegung ihrer Entwicklungsmethodik liegt demnach darin, für die jeweiligen (Teil-)Projekte das passende Modell zu identifizieren. Es ist hierbei vielversprechend, die Methodik innerhalb der Projektstruktur zu variieren, das heißt einzelne Teilprojekte und Arbeitspakete unterschiedlich entsprechend ihrer Bedürfnisse zu steuern. In der Folge müssen während der Projektinitialisierung die Schnittstellen, insbes. die Kommunikations- und Steuerungsmechanismen, festgelegt werden. Gerade bei großen Projekten mit mehreren Teilprojekten und prominenten Lenkungsausschüssen ist eine durchgängige Planung mit Fortschrittskontrolle sowie ein einheitliches Reporting und Stakeholdermanagement für den Projekterfolg elementar. Nicht zuletzt erfordern auch Compliances, z. B. CMMI, eine integrierte Sichtweise auf das Gesamtprojekt.

Dieser Artikel gibt einen differenzierten Blick auf den Einsatz agiler und klassischer Vorgehensmodelle. Statt der teilweise fast naiven Begeisterung für agile Methoden zu folgen, werden Vor- und Nachteile sowie Einsatzgebiete sorgfältig diskutiert. Es wird eine Kombination von klassischen und agilen Vorgehensweisen dargestellt, die die Vorteile beider Methoden vereint.

Zur Einführung werden zunächst typische Anwendungsgebiete und Projekttypen für klassische und agile Methoden dargestellt, um die Stärken der jeweiligen Modelle herauszustellen. Auf der Basis dieser Grundlagen wird argumentiert, dass der Zuschnitt der Entwicklungsmethode auf ein Projekt nur selten so eindeutig erfolgen kann. Es wird dargestellt, dass insbesondere bei großen Projekten eine differenzierte Auswahl der Vorgehensweisen für den Projekterfolg elementar ist und dass für diese Auswahl die Kombination von klassischen und agilen Entwicklungsmethoden gute Ergebnisse verspricht.

Im Hauptteil des Artikels wird anhand eines Praxisbeispiels vorgestellt, wie die Kombination von agilen und klassischen Methoden innerhalb eines Projekts funktionieren kann. Es wird gezeigt, wie auf Teilprojektebene die eher volatilen und die stabilen Anforderungen identifiziert und durch entsprechende Methoden umgesetzt werden können. Ebenso wird gezeigt, wie die Gesamtprojektsteuerung einen konsolidierten Blick auf das Projekt erhält und wie einheitliche Dokumentationsanforderungen erreicht werden können.

2 Grundlagen traditioneller und agiler Vorgehensmodelle

Im Bereich der Softwaretechnik werden seit den 70er Jahren Methoden zur Strukturierung des Entwicklungsprozesses entwickelt. Ausgehend von einem chaotischen, unstrukturierten

Entwicklungsprozess und der daraus resultierenden Softwarekrise (1) definieren diese Methoden mehr oder weniger detailliert die Rollen und die Aufgaben im Entwicklungsprozess. Als erwartete Ergebnisse werden Artefakte und Lieferobjekte beschrieben, die zur Steuerung des Projekts erforderlich sind bzw. das Projektergebnis darstellen sollen. In der Praxis sind häufig sehr detaillierte Rahmenwerke für die Projektarbeit in den Unternehmen vorgeschrieben.

Die meisten dieser Vorgehensmodelle verwenden das Konzept einer Phase, um den Gesamtprozess auf oberster Ebene zu partitionieren. Für jede Phase wird zwischen Auftraggeber und Projektleiter die Umsetzung eines bestimmten Teils des gesamten Projektumfangs als Zielvorgabe festgelegt. Der Auftraggeber erhält dadurch eine frühe Möglichkeit, den Projektfortschritt zu bewerten und ggf. Korrekturmaßnahmen einzuleiten. Durch inkrementelle und iterative Vorgehensweisen, wie z. B. im Rational Unified Prozess RUP (2) beschrieben, werden die bekannten Nachteile streng wasserfall-orientierter Modelle abgeschwächt.

Für große Projekte mit Teilprojekten muss das gewählte Vorgehensmodell um Konzepte zur Strukturierung der Arbeitsteilung und zur Synchronisation zwischen den Teilprojekten erweitert werden. Die meisten Vorgehensmodelle bieten hierzu wenig Hilfestellung, d. h. es bleibt den Unternehmen oder sogar den Projekten überlassen, wie diese Herausforderungen zu lösen sind. Zu betrachten ist hierbei nicht nur das Projektmanagement, sondern ebenso weitere Querschnitts-Disziplinen, z. B. Anforderungsmanagement und Qualitätssicherung.

Agile Vorgehensmodelle haben sich in den letzten Jahren in verschiedensten Formen und Auslegungen auch in großen Unternehmen etabliert (3). Basis dieses Modells ist die iterative Software-Entwicklung: Das Vorgehensmodell ist dadurch geprägt, dass die klassischen (Basis-) Phasen Entwurf/Design, Implementierung und Test in mehreren Iterationen durchlaufen werden. Je nach Lehre sind die Phasen innerhalb der Iterationen klar erkennbar oder verschmelzen miteinander.

Mit der Einführung von Iterationen wird ein sogenannter horizontaler Schnitt angelegt. Dieser Schnitt des umzusetzenden Gesamtwerks wird durch die Strukturierung nach Funktionsblöcken, Applikationsbereichen oder anderen gebildet. Bei der Definition der Iterationen wird dabei bewusst in Kauf genommen, dass zwar die Anforderungen in Gänze bekannt sind, jedoch nicht zwangsläufig in der für die Implementierung erforderlichen Tiefe detailliert vorliegen.

Eine der bekanntesten Ausprägung agiler Software-Entwicklung ist SCRUM (4). Hierbei werden Anforderungen im Product Backlog verwaltet, das als priorisiertes Sammelbecken für Anforderungen dient. Iterationen werden als Sprints bezeichnet und sind zeitlich klar begrenzt. Die in einem Sprint umzusetzende Funktionalität wird gemäß der Priorisierung aus dem Product Backlog in einem Sprint Backlog dokumentiert. Das Entwickler-Team verpflichtet sich dabei in einem Ritual zur Umsetzung. Die Umsetzung erfolgt dann möglichst unterbrechungsfrei und ohne äußere Einflüsse.

Durch verschiedene Maßnahmen und Regeln wird besonders großer Wert auf Verbindlichkeit für einen überschaubaren Zeitraum gelegt und dass das erzeugte Ergebnis ein möglichst direkt nutzbares Produkt ist. Maßgeblich für das Resultat und die inhaltliche Steuerung des Projekts ist der Product Owner. Die Organisation des Projekts und die Einhaltung der Methodik ist Aufgabe des SCRUM-Masters. Eine im klassischen Sinne definierte Projektleiter-Rolle ist nicht vorgesehen. Damit ändert die Steuerung von Projekten gemäß SCRUM-Methodik das klassische Rollenverständnis in Projekten ganz massiv.

In der Praxis wird dieses Vorgehensmodell von Entwicklerteams aufgrund der damit erklärten Wertschätzung und Verbindlichkeit positiv aufgenommen und erweist sich als stabiler Rahmen für die Team-Arbeit. Allerdings zeigt sich immer wieder, dass das Thema Anforderungsreife zwar durch die überschaubaren Sprints weniger dramatisch, jedoch keineswegs trivial ist. Wichtig ist, die Anwendung agiler Methoden nicht mit Beliebigkeit und Willkür zu verwechseln – es besteht ein klarer Unterschied zwischen diffus-unbeständigen Anforderungen und einer

definierten Anforderungsmenge unterschiedlicher Granularitäten. Ebendiese definierte und verlässliche Anforderungsmenge wird im Product Backlog vorgehalten.

Bevor jedoch jedwede Aktivitäten in Richtung Umsetzung von Anforderungen gestartet werden, wird in einem Ausblick die gesamte Menge der Anforderungen und die Vision des Anforderers (hier durch den Product Owner vertreten) im gesamten Team besprochen und Konsens darüber hergestellt. Dies bildet das Divide-and-conquer-Prinzip ab: Das große Ganze wird betrachtet, um die Richtung festzulegen – die einzelnen „Tages-Etappen" werden dann in den Sprints besprochen („wenn es konkret werden muss") (5). In der Praxis wird dieser Teil oft vergessen (ursächlich hierfür ist wahrscheinlich mangelnde Ausbildung und Erfahrung) oder nicht ernst genommen – dies ist aus unserer Sicht eine der Ursachen für das geringe Vertrauen in agile Vorgehensmodelle und insbesondere SCRUM. Die teils naive Begeisterung der SCRUM-Befürworter löst dabei nur weiteres Misstrauen bei Vertretern der traditionellen Vorgehensmodelle aus.

Gerade die Punkte Anforderungsreife und Rollenverständnis erzeugen bei der Einführung von SCRUM in großen Unternehmen großen Erklärungs- und Steuerungsbedarf. Die häufig unzutreffenden Freigabe-Prozesse können mit einer Standard-Umsetzung meist nicht bedient werden, sodass neben den direkten Projektaktivitäten auch bei den begleitenden Prozessen Schwierigkeiten auftreten.

3 Szenario für den Artikel

Die im Folgenden beschriebenen Konzepte sollen anhand eines Beispielprojekts illustriert werden. Das Beispiel ist an vielen Stellen vereinfacht, um auf die hier relevanten Aspekte eingehen zu können. Im ersten Schritt wird hierbei eine klassische Projektorganisation beschrieben. Das Projekt mit seinen Teilprojekten wird z. B. nach dem Vorgehensmodell des RUP abgewickelt.

In dem Entwicklungsprojekt wird eine Legacy-Host-Anwendung, wie sie in vielen Branchen zu finden ist, über ein Webfrontend zugänglich gemacht. Hierzu wird eine klassische Multi-Tier-Architektur implementiert, d. h. es werden gekapselte Schichten definiert. Die Schichten dienen zur Realisierung der Benutzerschnittstelle (Front-End), zur Implementierung der Sicherheitsanforderungen (Middle-Tier) und schließlich zur Anbindung an die existierende Funktionalität in der Altanwendung (Backend-Tier).

Zur Umsetzung dieses Projekts wird die in Abbildung 1 dargestellte Projektorganisation aufgebaut. Der Lenkungsausschuss (LA) delegiert mittels Projektauftrag die Projektverantwortung an den Gesamtprojektleiter. Dieser steuert einige Aktivitäten, z. B. das Anforderungs-, Stakeholder- und Risikomanagement, auf der obersten Ebene. Andere Aktivitäten werden über weitere Projektaufträge an die Teilprojektleiter delegiert. Dies kann z. B. das Konfigurationsmanagement sein, das vollständig an ein Teilprojekt delegiert wird. Insbesondere werden aber die Aktivitäten im Bereich Design und Implementierung beauftragt, also die Entwicklungsaufgaben. In dem Beispiel in Abbildung 1 wird jede der drei oben beschriebenen Schichten durch ein Teilprojekt realisiert, z. B. wird die Benutzerschnittstelle mit allen Masken und deren Navigation an das Teilprojekt Frontend beauftragt. Ein unternehmensweit zentralisiertes Testcenter koordiniert in diesem Beispiel schließlich die Testaktivitäten und stellt die Abschluss-Testate für die Launch-Freigabe aus.

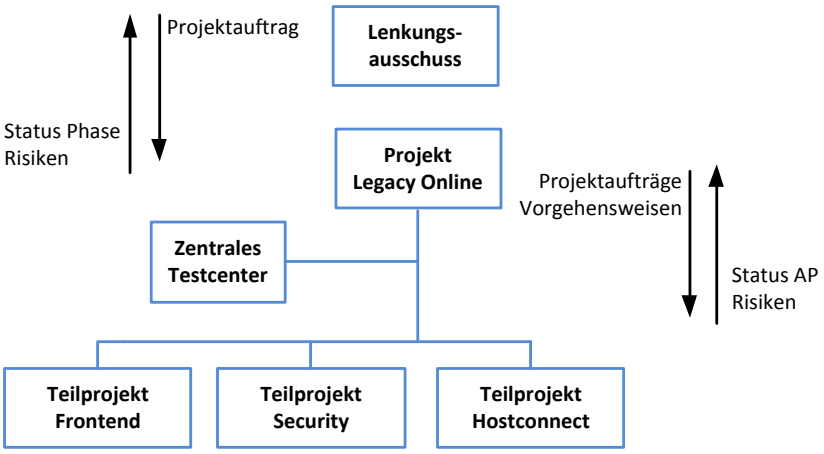

Abbildung 1: Organigramm für Beispielprojekt

Die Synchronisation zwischen Gesamtprojekt und den Teilprojekten erfolgt, wie in Abbildung 2 dargestellt, über Meilensteine und Phasenabschlüsse. Ausgehend von ihren Projektaufträgen planen die Teilprojektleiter ihre Arbeitspakete detailliert aus und melden die geplanten Meilensteine (bottom-up) in aggregierter Form an den Gesamtprojektleiter. In Zusammenarbeit mit dem Auftraggeber legt der Gesamtprojektleiter die Ziele für die jeweiligen Phasen fest.

Während der Projektdurchführung wird über Statusberichte diese Planung mit dem tatsächlichen Projektfortschritt abgeglichen. Dieses Controlling wird schrittweise (top-down) bis auf die Ebene des Teilprojekts fortgeführt. Veränderte Anforderungen werden, wegen des zentralisierten Anforderungsmanagements, ebenso schrittweise an die jeweilige Ebene weiter gereicht.

Die Granularität verändert sich demnach entsprechend der jeweiligen Ebene. Während der Lenkungsausschuss primär auf die Umsetzung der zur Auslieferung führenden Inkremente (i.S.v. Release-Paket) achtet, verfeinert der Gesamtprojektleiter diese Inkremente zu Arbeitspaketen, die weiter delegiert werden können. Der Teilprojektleiter verfeinert die delegierten Arbeitspakete weiter bis zu einer Granularität, die die Steuerung der Umsetzung ermöglicht. Damit ist die Skalierbarkeit bei großen Projekten ermöglicht.

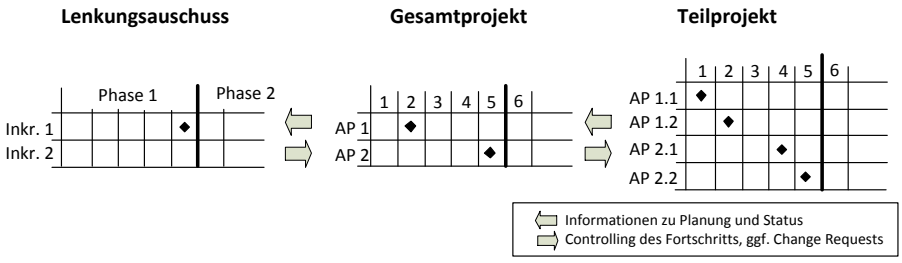

Abbildung 2: Verfeinerung der Aufgaben in der mehrstufigen Projekthierarchie

	Projekt-management	Anforderungs-management	Qualitäts-management	Konfiguration-management
Projekt Legacy Online	Verantwortlich	Verantwortlich	Berichten, Durchführen	Durchführen
Zentrales Testcenter	Verantwortlich	Berichten	Verantwortlich	Durchführen
Teilprojekt Frontend	Verantwortlich	Umsetzen	Berichten, Durchführen	Verantwortlich
Teilprojekt Security	Verantwortlich	Umsetzen	Berichten, Durchführen	Durchführen
Teilprojekt Hostconnect	Verantwortlich	Umsetzen	Berichtend, Durchführen	Durchführen

Abbildung 3: Aufteilung der Verantwortlichkeiten zwischen Gesamtprojekt und Teilprojekt

In diesem Beispiel wird die in Abbildung 3 dargestellte Aufteilung der Zuständigkeiten festgelegt. Für die Disziplin Projektmanagement sind die zentrale Projektsteuerung und die einzelnen Teilprojekte für die im Projektauftrag spezifizierten Leistungen verantwortlich. Die Gesamtverantwortung gegenüber dem Auftraggeber liegt zwar primär bei dem Gesamtprojektleiter, durch die Delegation und Vergabe von Teilprojekt-Aufträgen sind die Teilprojektleiter aber hier ebenso in der Verantwortung.

Das Anforderungsmanagement wird in diesem Beispiel zentralisiert durch die Gesamtprojektleitung verantwortet. Während der Projektdurchführung wird die Gesamtheit aller fachlichen Anforderungen demnach zentral erhoben und gepflegt. Die Teilprojekte haben die Aufgabe, die fachlichen Anforderungen im Systemdesign und Implementierung umzusetzen.

Die zentrale Verantwortung und Koordination für das Qualitätsmanagement übernimmt das Testcenter. Die Durchführung der Qualitätssicherungsmaßnahmen und Rückmeldung von Ergebnissen wird entsprechend durch die Teilprojekte geleistet.

4 Kombination der Ansätze

Das in Abschnitt 3 eingeführte Beispiel kann wie beschrieben auch ohne agile Ansätze gesteuert und umgesetzt werden. Dieses vielfach bewährte Vorgehen ermöglicht eine strikte Trennung der Verantwortlichkeiten und Zuständigkeiten. Gerade für das Anforderungsmanagement bietet die zentrale Steuerung auf Gesamtprojektebene die Vorteile einer gebündelten externen Kommunikation und einheitlichen Beschreibung der Kundenwünsche.

In dem dargestellten Beispiel kann es dennoch Vorteile bringen, in einigen Bereichen ein flexibleres Modell anzubieten. Gerade für das Teilprojekt Frontend sind häufige Iterationen mit dem Auftraggeber sinnvoll: Durch die Durchdringung der Anforderungen in mehreren Durchgängen bei gleichzeitiger Beschränkung auf eine sinnvolle Teilmenge werden Abweichungen von einer formalen (und vermeintlich vollständigen und korrekten) Anforderungsdefinition vermieden. Um diese intensive Interaktion mit dem Auftraggeber für das Projektvorgehen vorzusehen, bietet es sich an, das Teilprojekt Frontend agil zu steuern.

Wir erlauben hier explizit die Kombination verschiedener Vorgehensmodelle und bezeichnen dieses als Hybrides Vorgehensmodell. Mit derartigen Kombinationen lassen sich die jeweils optimalen Vorgehensmodelle für die Teilprojekte (auch im Sinne von Gewerken) wählen, was sich bereits in unseren Projekten in der Praxis bewährt hat. Wir werden insbesondere auf die unserer Meinung nach erfolgskritischen Aspekte der Integration in das Gesamtprojekt und auf die Zusammenarbeit zwischen Teilprojekten eingehen.

Für das Gesamtprojekt resultiert daraus ein hybrides Vorgehen: die Gesamtprojektsteuerung sowie die Teilprojekte Security und Hostconnect arbeiten nach dem klassischen Vorgehensmodell, z. B. RUP, das Teilprojekt Frontend agil, z. B. mit Scrum. Die Interaktion und Aufgabenteilung zwischen den Einheiten des Projekts muss neu definiert werden, um die Stärken des jeweiligen Vorgehens nutzen zu können.

Im Folgenden werden die Auswirkungen auf die hybride Projektsteuerung anhand der Disziplinen Anforderungsmanagement, Projektmanagement und Qualitätsmanagement untersucht. Hier wird sowohl die Sichtweise von der Gesamtsteuerung auf die Teilprojekte (top-down) berücksichtigt, als auch die erforderlich Integration der Teilprojekteinformationen zum Ganzen (bottom-up).

4.1 **Projektmanagement**

Durch die Verwendung unterschiedlicher Vorgehensmodelle wird die Projektorganisation auf den ersten Blick komplizierter, da in den Teilprojekten unterschiedliche Fähigkeiten gefordert und ein anderes Zusammenleben praktiziert wird. Da nur ein Teilprojekt nach SCRUM entwickelt wird, ist keine übergeordnete „SCRUM-Koordination“ erforderlich (Stichwort: Scrum of Scrums). Für die Steuerung des jeweiligen Teilprojekts ergeben sich bzgl. der Projektmanagement-Disziplin keine gesonderten Anforderungen aus dem hybriden Vorgehen im Gesamtprojekt.

Bei der Einbindung in das Gesamtprojekt ist eine Änderung der Kommunikation bzgl. des Arbeitsauftrags und des Reportings erforderlich. An dieser Stelle wird das Stage-Gate-Prinzip verwendet: Die Teilprojekte werden bzgl. des Projektmanagements als Black-Box betrachtet. Die Erteilung eines Arbeitsauftrags an das Teilprojekt erfolgt dabei in den SCRUM-üblichen Ritualen, es ist hier also eine engere Einbindung des Auftraggebers erforderlich. Idealerweise wird die Rolle des Product Owners durch den Leiter des übergeordneten Teilprojekts bzw. des Gesamtprojekts (wie hier in unserem Beispiel) besetzt. Kritisch für den Projekterfolg ist dabei, dass der Product Owner während eines laufenden Sprints nur Anforderungen (er)klärt und nicht verändert. Es ist die Aufgabe des Product Owners, die mitunter variierenden Interessen der Stakeholder zu kapseln und zu kanalisieren – auf keinen Fall darf die Variabilität in die Entwicklung durchschlagen.

Das Reporting basiert dabei auf dem Grad der Zielerreichung, der in diesem Teilprojekt durch Aufsummierung der verbleibenden Story-Points auf Basis der ursprünglichen Schätzung berechnet wird. Erst der Abschluss von Arbeitspaketen wird dann wieder als Meilenstein-Erreichung berichtet. Diese Art des Reportings muss dann auf der Ebene des Gesamtprojekts berücksichtigt werden – in den aggregierten Berichten an den Projekt-Auftraggeber treten diese Unterschiede nicht mehr in Erscheinung.

Diese Art der Fortschrittsmessung hat sogar Vorteile. Beim Messen auf Basis von Meilensteinen und Untermeilensteinen wird in aller Regel schlicht die Anzahl der erreichten Meilensteine als Maß für den Fortschritt angenommen – und damit implizit, dass das Erreichen jedes Meilensteins einen gleichen Anteil an der zu leistenden Arbeit darstellt. Bei der hier beschriebenen Messung fließen aber bereits die Expertenschätzungen des Teams (vgl. Burn-Down-Charts etc.) mit in die Fortschrittsmessung ein – was sicherlich ein höheres Maß an Genauigkeit darstellt.

In der Zusammenarbeit mit anderen Teilprojekten hat es sich bewährt, die Meeting-Strukturen für den formalen und informellen Austausch a priori festzulegen. In Tabelle 1 wird ein geeigneter Fahrplan für die Zusammenarbeit innerhalb des Projekts vorgestellt: Nach dem Kick-off bearbeiten die Teilprojekte die übertragenen Arbeitsaufträge, die Aggregation und Abstimmung erfolgt immer im 4-Wochen-Rhythmus. Dabei finden die angegebenen Meetings der Teilprojekte immer mindestens einen Tag vor den Besprechungen auf Gesamtprojekt-Ebene statt.

Für die direkte Koordination zwischen den Teilprojekten dienen die Teilprojektleiter und im Fall des agilen Teilprojekts der Product Owner als dedizierte Ansprechpartner. Über das hier nicht im Fokus stehende Konfigurationsmanagement sind ohnehin die produzierten Artefakte für alle Projektbeteiligten transparent und verfügbar.

	PW 0	**PW 1**	**PW 2**	**PW 3**	**PW 4**	**PW 5**	**PW 6**	**PW 7**	**PW 8**	**PW 9**	**PW 10**
GP	Kick-Off	JF	JF	JF	LA	JF	JF	JF	LA	JF	…
TP 1		Sprint Planing	Dailies	Dailies	Sprint Review + Planing	Sprint Planing	Dailies	Dailies	Sprint Review + Planing	Dailies	…
TP 2		JF	JF	JF	JF	JF	JF	JF	JF	JF	…
TP 3			JF	JF	JF	JF	JF	JF	JF	JF	…

Tabelle 1: Koordination des Gesamtprojekts (GP) mit den Teilprojekten (TP) in den Projektwochen (PW)

4.2 **Anforderungsmanagement**

Im Bereich des Anforderungsmanagement liegen die Stärken des agilen Vorgehens. Dies gilt insbesondere, wenn die Anforderungen nicht vorab vollständig erhoben werden können oder der Auftraggeber keine genaue Vorstellung von dem zu entwickelnden Produkt hat. Für die Entwicklung von Benutzerschnittstellen werden deshalb oftmals agile Vorgehensweisen favorisiert. Für die Gesamtprojektleitung und den Auftraggeber ergeben sich durch das hybride Vorgehen elementare Veränderungen.

Ein agiles Vorgehen setzt die enge Mitarbeit des Auftraggebers im Projekt voraus. Insbesondere die Anforderungsdefinitionen für einzelne Sprints werden zusammen mit dem Auftraggeber erarbeitet, aber auch die Ergebniskontrolle durch Review und Test erfordert das Mitwirken des Auftraggebers. Als Sprint wird hier ein Entwicklungszyklus von vier Wochen angenommen, der mit den üblichen Managementtechniken des agilen Manifests gesteuert wird.

Damit erfolgt das Anforderungsmanagement auf zwei Ebenen: 1. Die grundlegenden funktionalen und nicht-funktionalen Anforderungen für das Gesamtsystem werden weiterhin zentral durch die Gesamtprojektleitung erhoben und verwaltet. Über spezifische Projektaufträge erfolgt die Beauftragung der Teilprojekte. 2. Die spezifischen Anforderungen für die Benutzerschnittstelle werden von dem Teilprojekt Frontend in enger Abstimmung mit dem Auftraggeber erarbeitet. Diese definieren bzw. ergänzen den Projektauftrag dieses Teilprojekts.

Für den Projekterfolg ist es elementar, dass bei der verteilten Anforderungsentwicklung ein konsistentes Gesamtbild der Anforderungen entsteht. Es gilt zu vermeiden, dass (a) keine mehrfachen oder sich überschneidenden Anforderungen spezifiziert und ggf. implementiert werden und dass (b) die Anforderungsmenge vollständig ist, d. h. keine Anforderungen durch die Verteilung vergessen werden.

In der Projektorganisation muss diese Konsistenz organisatorisch und technisch abgesichert werden. Organisatorisch muss ein Vertreter der Gesamtprojektsteuerung bei der agilen Anforderungsdefinition im Teilprojekt als weiterer (interner) Auftraggeber mitarbeiten. Dieser Anforderungsmanager hat somit die anspruchsvolle Aufgabe, die inhaltliche Konsistenz der beiden Betrachtungsebenen zu gewährleisten.

Um den Anforderungsmanager hierbei zu unterstützen und eine Durchgängigkeit für folgende Projektphasen zu gewährleisten, müssen die verteilt erhobenen Anforderungen in einem einheitlichen Format gespeichert und einem gemeinsamen Werkzeug abgelegt werden.

4.3 Qualitätsmanagement

Bezüglich der Qualitätssicherung entsteht der größte Unterschied zwischen den Teilprojekten. Während in klassischen Vorgehensmodellen die Qualitätssicherung eine eher untergeordnete Rolle spielt und dies leider viel zu häufig tatsächlich ein Schwachpunkt in vielen IT-Projekten ist, wird in agilen Verfahren (insbesondere SCRUM) eine unumstößliche „Definition of Done“ postuliert: Ein Arbeitsauftrag darf nur dann vom Team freigegeben bzw. fertig gemeldet werden, wenn eine tatsächliche Qualitätssicherung aus der Verwender-Sicht (technischer oder humaner Verwender) durchgeführt wurde (6)(3).

Es hat sich in der Praxis bewährt, einen dedizierten Tester in das Entwicklungsteam zu integrieren. Für beste Ergebnisse ist hier die unmittelbare räumliche Nähe streng zu empfehlen – dies macht das Erleben von Fehlern beim Testen für das Team zu einem einschneidenden Ereignis und wird gemäß der menschlichen Psyche versucht zu vermeiden – durch die räumliche Nähe nicht mehr durch Flucht und Verdrängen, sondern durch Vermeidung der Entstehung minderwertiger Qualität. Hinzu kommt der Effekt, dass durch das frühe Erstellen der Testfälle die Programm-Abläufe bereits mental vorweggenommen werden. Dadurch setzen sich pro Entwicklungsteam mindestens zwei Personen mit den Anforderungen sehr detailliert auseinander und können Unklarheiten und Interpretationsspielräume deutlich einfacher (und auch früher) identifizieren und eine Klärung, z. B. durch den Product Owner, einfordern.

Neben der üblichen möglichst automatisierten Qualitätssicherung der Entwicklung, z. B. durch Unit-Test, wird die Betrachtungsebene auch auf die Ende-zu-Ende-Funktionalität und die Integration der erzeugten Artefakte erweitert. Dies erfolgt in unserem Modell immer in Ergänzung zu den etablierten Test- und Freigabeprozessen.

Bei der Zusammenarbeit mit anderen Teilprojekten bezogen auf die Qualitätsmanagement-Disziplin ergeben sich im hybriden Modell keine Änderungen. Es steht den anderen Teilprojekten natürlich frei, das Konzept des dedizierten Testers ebenfalls umzusetzen. Auch im hybriden Modell setzen wir einen dedizierten Tester ein. Dieser ist als Ergänzung zum zentralen Testcenter zu sehen. Die im Teilprojekt entstehende Test-Dokumentation verbleibt im Teilprojekt und wird nur bei Bedarf veröffentlicht (muss also nicht Teil des zentralen Konfigurationsmanagements sein).

4.4 Resümee

Mit den hier beschriebenen Konzepten und Arbeitsweisen gelingt es, die bisher erlebten Brüche bei unabgestimmten Vorgehensweisen zu überwinden. Mithilfe des etablierten Stage-Gate-Prinzips werden in unserem hybriden Vorgehensmodell die unterschiedlichen Vorgehensweisen effizient kombiniert, sodass ein aufgabenspezifisches Modell gewählt werden kann. Bei den wichtigsten drei Projekt-Disziplinen (Projektmanagement, Anforderungsmanagement und Qualitätsmanagement) haben wir im Detail gezeigt, wie diese Kombination in die Praxis umgesetzt werden kann.

5 Related Work

Die Autoren beschreiben in (7) den Unterschied zwischen dem inneren und äußeren Projektmanagement. Als inneres Projektmanagement wird hier die Abstimmung und Verteilung der Aufgaben innerhalb eines Teams verstanden, die – bei Anwendung agiler Methoden – nicht von einem Projektleiter sondern vom Team eigenständig organisiert wird. Unter dem Begriff des äußeren Projektmanagements beschreiben die Autoren alle eher formal als inhaltlich geprägten Aktivitäten. Die Synchronisation erfolgt wie in dem vorliegenden Beitrag über Meilensteine, die

an die Sprints angepasst sind. Das wichtige Konzept, inkrementelle Abnahmen über Quality Gates zu erreichen, wird hierbei allerdings noch nicht berücksichtigt.

Der Autor von (8) motiviert, dass die Umsetzung agiler Methoden mithilfe des Rational Unified Process (RUP) keinesfalls einen Widerspruch darstellt. Hierzu stellen die Autoren die Gemeinsamkeiten zwischen dem Agilen Manifest (6) und den Grundkonzepten des RUP (2) gegenüber. Es wird deutlich, dass insbesondere Kundennähe und Qualität in beiden Ansätzen elementare Ziele sind. Als Schwächen agiler Methoden werden insbesondere die fehlende Systematik beim Projekt-, Anforderungs- und Qualitätsmanagement sowie die Sicht auf die Gesamtarchitektur kritisiert. Es werden zwar die im RUP hierfür vorgesehen Disziplinen dargestellt, eine Kombination beider Ansätze wird allerdings nicht beschrieben.

Müller argumentiert in (9), dass die Kombination von klassisch und agil organisierten Teilprojekten sinnvoll ist und dass zertifizierte Projektmanager erforderlich sind, um insbesondere die Skalierbarkeit und durchgängige Kommunikation sicher zu stellen. Die Verzahnung der heterogenen Teilprojekte soll hierbei durch Entwicklungswerkzeuge erreicht werden.

Dass bei richtigem Projektzuschnitt auch klassische Vorgehensmodelle, z. B. der RUP, ihre Agilität behalten, wird in (13) motiviert. Entscheidend ist hierbei die richtige Besetzung der Projektrollen, insbesondere im Übergang von Fachkonzept zum Systemdesign. Die Autoren beschreiben in (5), dass eine hohe Prozessreife des Unternehmens sowie große Motivation und Disziplin bei den Projektmitarbeitern erforderlich sind, um mit agilen Vorgehensweisen erfolgreich zu sein.

In (10) beschreibt Reiss die Anforderungen und Möglichkeiten zum Aufbau eines hybriden Vorgehensmodells. Als besondere Herausforderung wird hierbei die Zusammenführung der heterogenen Schnittstellen des Entwicklungsprozesses herausgestellt. Die Bewertung hybrider Vorgehensmodelle erfolgt bei ihm auf der Basis der allgemeinen Kriterien Effektivität und Effizienz und stellt dar, dass insbes. die Effizienz leidet („Effektiv, aber ineffizient“). In der Tat bergen hybrid-agile Vorgehensmodelle das Risiko, dass Projektergebnisse, die für den langfristigen und kosteneffizienten Betrieb von IT-Systemen erforderlich sind, dem schlanken Entwicklungsprozess zum Opfer fallen. Hier gilt es, die betrieblichen Erfordernisse von Beginn an in die Anforderungsmenge aufzunehmen. In dem hier beschriebenen hybriden Vorgehensmodell liegt dies typischerweise in der Verantwortung der Gesamtprojektsteuerung.

6 Zusammenfassung

Üblicherweise werden Projekte ausschließlich agil oder klassisch aufgesetzt und durchgeführt. Dabei stehen sich die jeweiligen Befürworter meist äußerst skeptisch gegenüber. Wir fokussieren hier auf die Kombination der unterschiedlichen Welten und zeigen einen Weg, um die unterschiedlichen Philosophien zusammen zu bringen.

Aktuell besteht eine Begeisterung für SCRUM, da viele Nachteile klassischer Modelle nicht mehr zu bestehen scheinen. Förderlich sind dabei einerseits der Buzz-Word-Charakter und das vermeintliche Abwerfen unangenehmer Formalismen. Agile Methoden versuchen von der vollständigen Komplexität zu abstrahieren und die Verbindlichkeit durch eine höhere Verantwortung und Motivation der Projektmitarbeiter auszugleichen. RUP und vergleichbare Vorgehensmodelle stehen dazu im Gegensatz mit dreistelliger Anzahl von Rollen und aufwändigen Tailorings. Die Strategie von RUP liegt darin, die Komplexität von Entwicklungsprozessen im Modell abzubilden. Die Dokumentation wird in der Folge sehr umfangreich und komplex.

In diesem Beitrag wird der RUP den klassischen Vorgehensweisen zugeordnet. Dies ist allgemein umstritten (11) (12), da durch mehrere Inkremente und fortlaufende Beteiligung des Auftraggebers agile Grundideen bereits realisiert werden. Dennoch beschreibt der RUP ein eher komplexes und detailliert festgelegtes Vorgehen, das an vielen Stellen im Gegensatz zu

dem agilen Manifest (6) steht. Für große Projekte ist ein sorgfältiges Tailoring der Methode, Festlegung der Verantwortlichkeiten für jede Disziplinen und Struktur der internen und externen Projektkommunikation elementar. Die Einbettung agiler Teilprojekte schafft hierbei einen elementaren Mehrwert an den Stellen, wo Agilität und Kundennähe erforderlich ist.

An einigen Stellen ist zusätzlicher Formalismus für das agil gesteuerte Teilprojekt erforderlich. Dieser Formalismus steht zwar im Widerspruch zur rein agilen Vorgehensweise, er sichert aber die Integration des agilen (Teil-)Prozesses in das traditionelle (Gesamt-) Prozessmodell:

- innerhalb der Disziplin Anforderungsmanagement können Abhängigkeiten zwischen Anforderungen, sowohl in der eher statischen Gesamtsicht auf das zu entwickelnde Projekt, als auch in der dynamischen Evaluation der Anforderungen zur Projektlaufzeit berücksichtigt werden.
- in der Integration der Disziplin Projektmanagement kann eine gesamtheitliche Planung erstellt, ausgeführt und überwacht werden. Die inhaltliche Bewertung des Projektfortschritts im agilen Teilprojekt ermöglicht ein aussagekräftiges Controlling sowie die Aggregation in die Gesamtprojektplanung.
- in der Integration der Disziplin Qualitätsmanagement können die gesammelt spezifizierten Anforderungen als Grundlage für Testplanung und –ausführung verwendet werden. Als positiven Nebeneffekt zur agilen Anforderungsbestimmung wirken die hierbei höher priorisierten Testanforderungen zusätzlich qualitätssteigernd auf die anderen Teilprojekte und das Gesamtprojekt.

Insgesamt betrachtet ist dieser zusätzliche Aufwand gerechtfertigt, da die erreichten positiven Effekte die Aufwände überkompensieren. Diese Betrachtung wird gestützt durch unsere Projekte, die wir gemäß dem hier beschriebenen hybriden Modell durchgeführt haben. Die Projektgröße variierte dabei, wobei sich die deutlichsten Verbesserungen bei mittleren und großen Projektvorhaben gezeigt haben. Dabei hat sich auch gezeigt, dass auch große Konzerne mit diesem hybriden Modell ausgezeichnet arbeiten können: sowohl die Adaption der verschiedenen Rollen und die wie oben beschriebene Anpassung der Projekt-Arbeitsweise als auch die Einbettung in Konzern-Strukturen (z. B. bzgl. Testaten, Freigaben und Prozess-Konformität, Metriken) erwiesen sich als sehr leicht und rasch umsetzbar. Als Hindernis haben sich hier lediglich bereits vorhandene Vorbehalte bei Projektbeteiligten gegenüber dem einen oder anderen Vorgehensmodell gezeigt.

Um dieses Hindernis zu umgehen, ist es essentiell, gegenseitiges Verständnis aufzubauen und keine harten Abgrenzungen zuzulassen: hier steht die Herausforderung „Mensch“ im Blickpunkt (Vertrauen statt Kontrolle, Motivation, Bildung, u.v.m.), die durch Regeln im Zusammenspiel gut zu meistern ist. Letztlich schlägt auch unser hybrides Modell in die Kerbe der Veränderungsprozesse in den Unternehmen, weg von der vollständigen Spezifikation hin zum wirklich inkrementellen Arbeiten.

Der hier beschriebene Ansatz bietet noch eine Reihe möglicher Erweiterungen, die hier als kurzer Ausblick beschrieben werden. So liegt beispielsweise eine Herausforderung bei der Anwendung umfassender Vorgehensmodelle darin, die in Artefakten und Ergebnistypen vorliegenden Informationen zu vernetzen. Die Motivation für die Vernetzung liegt einerseits darin, mehrfache Eingaben gleicher Sachverhalte zu vermeiden. Einen weitaus größeren Ertrag erzielt man allerdings durch die Nutzung der vernetzten Informationen, um die Abhängigkeiten im Projekt zu erkennen. Gängige Prozessframeworks wie CMMI fordern eine Vernetzung (Traceability) z. B. zwischen den Anforderungen und Elementen der Disziplinen Projektmanagement, Test und Design und Implementierung. Diese Vernetzung von Artefakten und Metriken ist als nächster Schritt zu untersuchen.

Basierend auf den gewonnenen Erfahrungen gehen wir davon aus, dass sich das hier beschriebene hybride Modell auf andere Projekt-Domänen ausweiten lässt, zumindest auf Technologie-Projekte. Hier ist dann insbesondere zu untersuchen, inwieweit das Projektverständnis und die gängige Praxis sowie etwaige regulatorische und rechtliche Vorgaben sich im hybriden Modell umsetzen lassen und mit welchem Adaptionsaufwand.

Literaturverzeichnis

1. Software Engineering: Report of a conference sponsored by the NATO Science Committee. Naur, P. und Randell, B. Garmisch, Germany : NATO , 1968. Scientific Affairs Division.
2. Jacobson, I., Booch, G. und Rumbaugh, J. The unified software development process. Addison-Wesley, 1999.
3. Schwaber, K. Agile Project Management with Scrum. Microsoft Press, 2004.
4. Schwaber, K und Beedle, M. Agile Software Development with Scrum. PEARSON STUDIUM, 2008.
5. Agil in die Sackgasse. Aghajani, B. und Kirchhof, M. 2010, manage it, S. 1-6.
6. Beck, K. Manifesto for Agile Software Development. [Online] [Zitat vom: 10. 06 2011.] http://agilemanifesto.org/.
7. Agile Softwareentwicklung im Kontext unternehmensweiter IT-Prozesse. Herzog, J., Holzmüller, G. und Schoeppe, W. 2010, Objektspektrum - Sonderbeilage Agilität, S. 6-7.
8. Agil(e) mit RUP: Gegensatz oder ideale Verbindung. Essigkrug, A. 2010, Objektspektrum - Sonderbeilage Agilität, S. 8-10.
9. Welcome to Reality! Agile vs. Klassisch. Müller, T. 2010, Objektspektrum - Sonderbeilage Agilität, S. 16-17.
10. Hybride Vorgehensmodelle. Reiss, M. In: O. Linssen, et al., et al. 2010: Integration von Vorgehensmodellen und Projektmanagement, S. 1-14.
11. Ambler, S. Agile Modeling - Effective Practices for Extreme Programming and the Unified Process. John Wiley & Sons, 2002.
12. Cohn, M. Succeeding with Agile: Software Development Using Scrum. Addison-Wesley Professional, 2009.
13. Projektzuschnitt für die inkrementelle Systementwicklung im Konzernverbund. Hacker, T., Kraft, B. und Zöll, A. In: Kuhrmann, M.; Linssen, O., 2011: 18. WSGIVM: Zusammenspiel von Vorgehensmodellen und Organisationsformen, S. 1-11.