Uwe Vigenschow

APM – Agiles Projektmanagement

Anspruchsvolle Softwareprojekte erfolgreich steuern

→ Unter Mitarbeit von Andrea Grass, Alexandra Augstin und Michael Hofmann

dpunkt.verlag

Uwe Vigenschow ist Abteilungsleiter bei Werum IT Solutions. In das Buch sind über 25 Jahre Erfahrung in der Softwareentwicklung als Entwickler, Berater, Projektleiter und Führungskraft eingeflossen. Mit agilen Konzepten befasst er sich seit Ende der 1990er-Jahre und hat APM, Scrum und XP bei verschiedenen Firmen eingeführt und an besondere Rahmenbedingungen angepasst.

Andrea Grass arbeitet als Trainerin und Beraterin für die oose Innovative Informatik eG. Sie führt Agilitätschecks durch, unterstützt Teams darin, Agilität zum Leben zu erwecken und im Unternehmen zu verankern. Mit großer Begeisterung und Engagement begleitet sie Gruppen auf ihrem Weg, ein eingespieltes und lauffähiges Team zu werden.

Alexandra Augstin coacht agile Teams bei einem der weltweit führenden Unternehmen der Gamesbranche. Zuvor war sie lange Zeit als Trainerin und Beraterin mit den Schwerpunkten agiles Projektmanagement, Kommunikation und Change Management tätig und einige Jahre als Entwicklungsingenieurin in der Halbleiterindustrie beschäftigt.

Dr. Michael Hofmann ist Arbeits- und Organisationspsychologe. Seit 2010 hilft er als Trainer und Berater für die oose Innovative Informatik eG Unternehmen dabei, Lean, Agile und Scrum einzuführen.

APM - Agiles Projektmanagement

Anspruchsvolle Softwareprojekte erfolgreich steuern

Unter Mitarbeit von Andrea Grass, Alexandra Augstin und Michael Hofmann

Uwe Vigenschow

dpunkt.verlag

Uwe Vigenschow
uwe@vigenschow.com

Lektorat: Christa Preisendanz
Copy Editing: Ursula Zimpfer, Herrenberg
Satz: Uwe Vigenschow, Hamburg
Herstellung: Frank Heidt
Umschlaggestaltung: Helmut Kraus, www.exclam.de
Druck und Bindung: M.P. Media-Print Informationstechnologie GmbH, 33100 Paderborn

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.deabrufbar.

ISBN:
Buch 978-3-86490-211-6
PDF 978-3-86491-631-1
ePub 978-3-86491-632-8

1. Auflage 2015

Wieblinger Weg 17
69123 Heidelberg

5 4 3 2 1 0

Vorwort

Warum und wann ist es für Sie sinnvoll, sich mit *APM* zu befassen? Darauf gibt es eine kurze, plakative Antwort, wie sich *APM* in die Welt der agilen Projektmanagementvorgehensweisen einordnet:

> Wenn Sie Scrum machen können, machen Sie Scrum! Wenn Ihnen Scrum als agiles Framework zu umfangreich erscheint, setzen Sie Kanban ein! Wenn Sie jedoch aus unterschiedlichen Gründen mehr brauchen als Scrum, dann ist *APM* die richtige Methode für Sie!

Was bedeutet das? *APM* steht für ein **A**giles **P**rojekt**m**anagement von anspruchsvollen Softwareprojekten. *APM* kann wesentlich umfassender eingesetzt werden, als es Scrum in seiner reinen Form ermöglicht. Sie erhalten in diesem Buch einen praxisorientierten Überblick über *APM* und seinen effizienten und erfolgreichen Einsatz in der Projektarbeit. Sie erfahren, wie von der Projektvorbereitung bis zu einem agilen Requirements Engineering und einer durchgängigen Softwarearchitektur agil entwickelt werden kann. Dabei skaliert *APM* von Sieben-Personen-Teams bis hin zu Großprojekten mit mehreren 100 Mitarbeitern in verteilten Teams. *APM* wird auch für die Softwareentwicklung in regulierten Umfeldern wie der Luftfahrt- und Automotive-Industrie oder der Medizin- oder Pharmatechnik eingesetzt, bei denen besonders hohe Sicherheitsvorgaben einzuhalten sind. *APM* kann auf Ihre spezifischen Anforderungen angepasst werden und ermöglicht Lösungswege, die in der Praxis funktionieren. Hinter dem aktuellen Stand von *APM* stecken mittlerweile über zehn Jahre Erfahrung.

APM beschreibt eine Methodik für agile Projekte, die erstmals 2006 von Bernd Oestereich und Christian Weiss veröffentlicht wurde [118]. Seitdem haben wir *APM* kontinuierlich weiterentwickelt, was letztendlich zu diesem aktuellen Buch zur *APM*-Methodik geführt hat.

Bernd und Christian haben sich in den letzten Jahren anderen Aufgaben gewidmet, sodass ich als ein Wegbegleiter seit der ersten Stunde diese Aufgabe gerne übernommen habe.

Via negativa

Michelangelo soll auf die Frage, wie er seine meisterhafte Skulptur des David erschaffen habe, geantwortet haben, dass er einfach nur alles entfernt habe, was nicht nach David aussah. Man bezeichnet dieses aus der Theologie entlehnte Vorgehen als *Via negativa*, also den Weg des Verzichts und unbedingten Reduzierens auf das Minimum [45]. Das bereits den alten Griechen und Römern bekannte Prinzip beschreibt für mich sehr gut, was agiles Projekmanagement bedeutet. Ohne genau zu wissen, wie wir in einer bestimmten Situation optimal agieren, können wir zumindest mit unserem Vorgehen schädliches Handeln vermeiden. Wir besinnen uns auf die agilen Werte, Prinzipien und Praktiken.

Das klingt überschaubar. Dennoch ist die Umsetzung in der Praxis nicht trivial. *APM* unterstützt Sie dabei, die relevanten Techniken einzusetzen. Somit bietet *APM* einen gut gefüllten Werkzeugkasten für viele unterschiedliche Situationen im agilen Projektmanagement.

Begriffe und geschlechtsspezifische Bezeichnungen

In der globalen Softwareentwicklung dominiert eine englische Begriffsbildung. Diesem Fakt tragen wir Rechnung. Daher werden geläufige Begriffe in *APM* entsprechend auf Englisch benutzt. Die Bezeichnung von Rollen erfolgt wie in der Praxis oft nur in der männlichen Form, obwohl natürlich Frauen ebenso alle *APM*-Rollen erfolgreich ausfüllen.

Danksagung

Mein Dank gilt zahlreichen Personen. In alphabetischer Reihenfolge nach Vornamen sind dies: Bernd Oestereich, Carsten Bierans, Christian Weiss, Guido Zockoll, Heinrich Hambloch, Henning Landgrebe, Jan Gentsch, Jan Wegener, Klaas Reineke, Marcel Ecks, Marcus Winteroll, Markus Wittwer, Matthias Ferdinand, Nils Aue, Nils Bokholt, Oliver

Busch, Oliver F. Lehmann, Stefan Toth, Stefan Zörner und Stephan Roth.

Mit Andrea Grass und Alexandra Augstin habe ich *APM* während unserer gemeinsamen Zeit bei oose intensiv weiterentwickelt. Michael Hofmann hat mich bei Kapitel 3 maßgeblich unterstützt und das Kapitel 23 geschrieben. Dafür gebührt den Dreien mein großer Dank! Ganz besonders möchte ich auch den Projektleitern, Product Ownern, Scrum Mastern, Entwicklern, Architekten und Testern, die ich in ihren Projekten beraten durfte, danken. Ines Meyrose danke ich für die Skizzen und Zeichnungen, die ich in den Abbildungen verwenden durfte.

Das Buch wäre nichts geworden ohne die tolle Unterstützung durch den dpunkt.verlag und Christa Preisendanz sowie Ursula Zimpfer. Insbesondere danke ich Prof. Dr. Heidi Heilmann und den anderen Reviewern des Manuskripts für ihre wertvollen und konstruktiven Anregungen.

Uwe Vigenschow, Hamburg, im Dezember 2014

Aufbau und Überblick

Das Projektmanagement-Framework *APM* bietet umfangreiche Praktiken, die in diesem Buch beschrieben werden. Als Navigation und Orientierung finden Sie hier eine Übersicht über die Struktur des Buchs mit seinen fünf Teilen und insgesamt 23 Kapiteln.

Teil I - Agilität und APM-Einführung

1 Die Architektur von APM Hier lernen Sie das Fundament und die fünf Säulen von *APM* kennen, und wir führen einige zentrale Begriffe ein, die für das Verständnis von *APM* essenziell sind. Diese Begriffe werden später noch genauer definiert. Dieses Grundverständnis benötigen wir für die ersten Darstellungen der Abläufe in agilen Projekten.

2 Was ist Agilität? Es gibt unterschiedliche Sichtweisen auf Agilität sowie agile Werte und Prinzipien und darauf, was das in der Praxis bedeutet. Wir werfen einen Blick auf die *APM*-Sichtweise und die systemtheoretischen Hintergründe agiler Vorgehensweisen.

3 Die wirtschaftliche Sicht Letztendlich zählt der wirtschaftliche Erfolg. Ein Projekt muss sich rechnen. An diesem Punkt hat sich jeder Managementansatz zu bewähren. Und genau dort können agile Ansätze Erfolg versprechend wirken. Mit *APM* steht Ihnen eine Möglichkeit bereit, Agilität auch in größeren Projekten bzw. im Umfeld großer Organisationen zu etablieren und gleichzeitig wirtschaftlich erfolgreich zu sein.

4 Was kann Agilität leisten? Mit Agilität erreichen wir nachweislich einen schneller nutzbaren Stand eines Softwareprodukts und wir reduzieren aktiv Risiken in der Softwareentwicklung, wodurch u.a. die Erfolgswahrscheinlichkeit erhöht wird. Hier blicken wir hinter die

offensichtlichen Aussagen über Agilität und betrachten den individuellen Verbesserungsprozess und die Flexibilität, die über die Umsetzung verschiedener Konzepte gewonnen wird, ebenso wie die Risiken, die mit einer agilen Vorgehensweise verbunden sind.

5 Die Konzepte und Methoden von APM Sie lernen in diesem zentralen Kapitel des ersten Teils die Konzepte und Methoden in sich geschlossen im Zusammenhang kennen. Wir betrachten die Struktur und den Umgang mit Anforderungen, die notwendigen Artefakte und die grundlegenden Meetings mit Projektbeteiligten und/oder Stakeholdern zur Sicherstellung der internen und externen Informationsflüsse und der Rückkopplungsschleifen, die essenziell sind für den wirkungsvollen Einsatz von *APM* und somit für den Projekterfolg.

6 Das Rollenmodell *APM* besitzt ein skalierbares und flexibles Rollenmodell, um unterschiedlich große Projekte adressieren zu können. Es gibt dafür ein einfaches Rollengrundmodell und verschiedene Ausbaustufen für unterschiedliche Projektgrößen und Rahmenbedingungen.

7 Das Phasenmodell In *APM* werden zwei Varianten eines Phasenmodells unterschieden, um Neuentwicklungen und Wartungsprojekte angemessen abbilden zu können. Über diese Phasen wird eine zeitliche Einteilung und Struktur des *APM*-Projekts ermöglicht.

Teil II - Das Projekt aufsetzen

8 Ein agiles Projekt vorbereiten In diesem Kapitel erfahren Sie, was alles zur Vorbereitung eines agilen Projekts gehört. Sie lernen außerdem das Fallbeispiel kennen, auf das wir in den folgenden Kapiteln immer wieder zurückkommen, um einzelne Aspekte zu illustrieren.

9 Die Projektvorbereitung im Detail Wir gehen durch die Artefakte von der Vision über die erste Releaseplanung bis zum agilen

Projektstrukturplan. Hinzu kommen ein erster Blick auf das Risikomanagement und die Durchführung eines Projekt-Kick-off.

10 Umfang und Aufwand schätzen Die Prinzipien und Techniken der agilen Schätzweise werden erläutert und die zugrunde liegenden Konzepte aus Teil I weiter verfeinert. Konkrete Techniken zur Vorab-Projektschätzung werden ebenso beschrieben wie die Ursachen für die Schwierigkeiten dabei und das Einplanen von Risikoreserven.

Hier gehen wir primär auf die initale Schätzung ein. In agilen Projekten ist *Planung* ein regelmäßiger, beinahe schon kontinuierlicher Vorgang, der uns in jeder Iteration begleitet. Planung basiert auf Schätzungen. Damit gehören Schätzmethoden zur Projektmanagementbasis. Mit dem iterativen Schätzen beschäftigen wir uns weiter in Kapitel 15.

11 Agile Projekte und Verträge Agile Projekte legen den Fokus auf hohe gegenseitige Transparenz und enge, direkte Zusammenarbeit von Auftraggeber und Auftragnehmer. Das schlägt sich in der Gestaltung von Verträgen nieder und wird hier mit allen seinen möglichen Konsequenzen beschrieben. Zusätzlich ergeben sich diverse rechtliche Aspekte, für deren Klärung Sie bitte einen Experten hinzuziehen, bevor Sie einen ersten Vertrag für ein agiles Projekt abschließen!

Teil III - Agile Architektur

12 Softwarearchitektur Was bedeutet Softwarearchitektur für agile Projekte? Der Begriff *Architektur* wird in diesem Sinne geschärft und in den Kontext agiler Projekte gestellt. Dazu wird das Modell der *Architektur-Brezel* als Prozess- und Entscheidungsmodell erläutert. *APM* zeichnet sich durch regelmäßige Workshops zur Bewertung der Architektur aus, um technische Schulden auf der Architekturebene über das Mittel der *Tech-Story* zeitnah zu tilgen. Des Weiteren werden Techniken für gemeinsame Architekturentscheidungen vorgestellt und der Bezug zwischen Architektur und Qualität über Szenarien hergestellt.

13 Der Architekt und die Architekturphase Die Rolle des Architekts in einem agilen Projekt als *Architecture Owner* wird im Detail beschrieben. Er schafft einen Rahmen, um die Mitglieder der Teams direkt in die Entscheidungsfindung einzubinden, und treibt die Architekturthemen aktiv voran. Damit bedarf es für die Rolle des Architecture Owner sowohl der Fähigkeiten eines technischen Experten als auch eines Moderators und Organisators.

Teil IV - Konstruktion und Releases

14 Struktur und Dynamik einer Iteration In der Konstruktionsphase entfalten agile Vorgehensweisen vollends ihre Stärken und Möglichkeiten. Die Iterationen laufen nach einem festen Schema ab und wir erstellen eine Abfolge nutzbarer Releases. Die innere Struktur einer Iteration wird mit ihren Aktivitäten beschrieben. Die Planung einer Iteration wird ebenso detailliert erläutert wie der Entwicklungsmikroprozess und die testgetriebene Entwicklung. Abschließend wird die Durchführung von Retrospektiven am Ende jeder Iteration und ihr Wert für die kontinuierliche Prozessverbesserung dargestellt.

15 Fortlaufende Backlog-Arbeit Wir gehen in diesem Kapitel auf die fortlaufende und hierarchische Zerlegung eines Epos bis zur verfeinerten User Story über eine Reihe von *Splitting Patterns* ein. Auch die Themen agiles Schätzen, Refactoring und Planungsreserven werden wieder aufgegriffen und eingehend betrachtet.

16 Regelmäßige hochwertige Releases Das Schneiden von Releases sowie die Unterscheidung zwischen internen und externen Releases und deren Planung werden im Detail beschrieben. In diesem Zusammenhang gehen wir auch auf Qualitätsaspekte ein.

17 Projektcontrolling und -steuerung Das auf der Velocity basierende agile Projektcontrolling wird mit seinen Visualisierungen beschrieben. Dabei geht es um die Steuerung des Projekts und der Aktivitäten innerhalb einer Iteration. Metriken für agile Projekte und

ihr Nutzen bei der Selbstorganisation und -steuerung der Teams werden erläutert

18 Kanban und Lean Management Hier werden Kanban und Lean Management als eine der fünf Säulen von *APM* genauer beschrieben. Der Einsatz dieser Methoden hilft besonders bei der Ergebnisorientierung und der Früherkennung negativer Trends in der Projektumsetzung.

19 Der agile Coach im täglichen Einsatz Die vielfältigen und zahlreichen Aufgaben eines agilen Coaches werden beleuchtet und das Konzept der adaptiven Führung erläutert. Wie trägt ein agiler Coach zur Wertschöpfung bei? Welche Anforderungen werden an diese Rolle geknüpft? Was bedeuten Coaching, Mentoring und Moderation genau? Auch auf die agilen Werte und wie sie zum Leben erweckt werden können, wird in diesem Zusammenhang eingegangen.

Teil V - Agile Großprojekte

20 APM für große Projekte skalieren *APM* kann als vorskaliertes agiles Projektmanagement-Framework für mittelgroße Softwareprojekte mit zwei bis fünf parallel arbeitende Featureteams eingesetzt werden. In diesem Kapitel betrachten wir die Konzepte, um *APM* zu skalieren. Das Scrum-of-Scrums-Konzept wird auf *APM* übertragen und die Konsequenzen für die Featureteams und deren Zusammenarbeit sowie für das Projektleitungsteam werden analysiert. Die interne Struktur einer Iteration mit Fortschritts- und Orientierungsteil spielt dabei eine besondere Rolle für die Kommunikation in Großprojekten ebenso wie die Communities of Practice sowie der Einsatz von Persona und Szenarien.

21 Verteilte Teams Zuerst analysieren wir verschiedene Arten der Verteilung von Teams und betrachten mögliche Risiken, die sich daraus ergeben. Vorgehensweisen und Strategien werden erläutert und wie sie sich einzeln oder kombiniert umsetzen lassen.

22 Besonderheiten im regulierten Umfeld Erprobte agile Vorgehensweisen sind mittlerweile auch bei der Softwareentwicklung unter besonders hohen Sicherheitsvorgaben üblich und werden von den entsprechenden Richtlinien unterstützt. Wir betrachten exemplarisch an GAMP 5 und AAMI TIR45 die Konsequenzen, die sich daraus ergeben, und ergänzen die risikoorientierte Vorgehensweise in *APM* um entsprechende Vorgehensweisen.

23 Agilität im Unternehmen einführen Zum Abschluss betrachten wir die Einführung von Agilität in einem mittelständischen oder großen Unternehmen. Die Einführung von agilem Projektmanagement ist ein Change-Management-Projekt und die entsprechende Organisationsentwicklung ist nach agilen Prinzipien zu gestalten.

Inhaltsverzeichnis

Teil I
Agilität und APM-Einführung

APM besitzt ein skalierbares und flexibles Rollenmodell, um unterschiedlich große Projekte unter verschiedenen Rahmenbedingungen adressieren zu können.

In *APM* werden zwei Varianten eines Phasenmodells unterschieden, um Neuentwicklungen und Wartungsprojekte angemessen abbilden zu können.

1 Die Architektur von APM

Agile Vorgehensweisen sind im grundsätzlichen Ansatz einfach zu verstehen, jedoch wird die innere Komplexität ihrer Umsetzung sofort deutlich, wenn wir versuchen, detailliert zu erklären, wie ein agil durchgeführtes Projekt konkret abläuft. Bevor wir in die Tiefen von *APM* eintauchen, finden Sie hier einen kurzen Überblick über die zentralen Elemente von *APM*. Danach klären wir Begriffe aus dem agilen Projektmanagement, die wir später noch genauer definieren werden, aber bereits für die ersten Darstellungen der Zusammenhänge in ihrer grundlegenden Bedeutung benötigen.

1.1 Was ist APM?

APM steht für **A**giles **P**rojekt**m**anagement und beschreibt ein Framework, um Softwareprojekte mit der notwendigen Flexibilität umzusetzen und der hohen Dynamik des Projektumfelds angemessen begegnen zu können. Es fußt auf fünf Säulen und nutzt eine Vielzahl einzelner Best Practices aus unterschiedlichen methodischen Vorgehensweisen (Abb. 1-1):

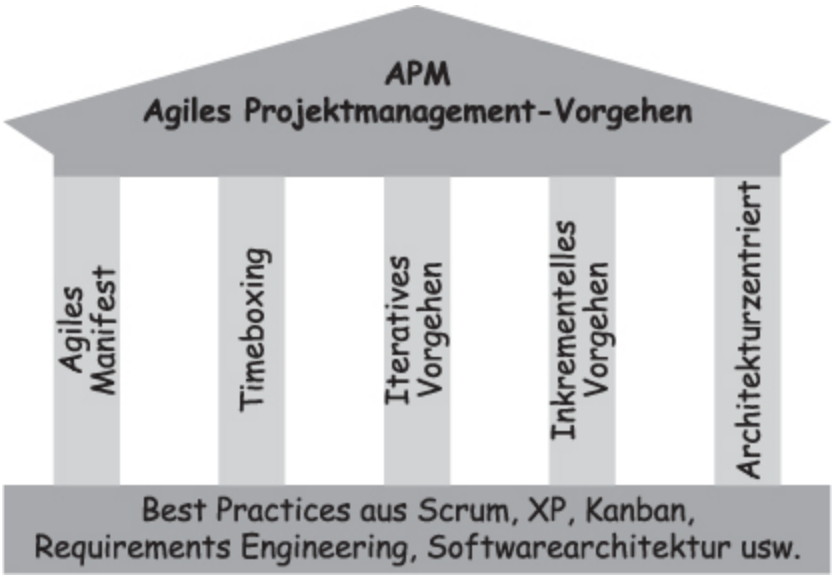

Abbildung 1-1: Das Fundament und die fünf Säulen von *APM*

- *APM* orientiert sich am Agilen Manifest, also an den beschriebenen vier Wertepaaren, und fußt auf den dort genannten zwölf Prinzipien als grundlegende Basis für das Vorgehen, die Führung und die Zusammenarbeit innerhalb des Projekts sowie mit den Stakeholdern [12].
- *APM* implementiert ein durchgängiges Timeboxing zur Planung und Steuerung von Projekten. Eine *Timebox* ist ein fester, vorab definierter Zeitabschnitt, in dem wir planen, ein inhaltliches Ergebnis zu erreichen, das wir am Ende überprüfen können. Dieses Konzept durchdringt *APM* von kleinen Abschnitten der täglichen Zusammenarbeit wie in täglichen kurzen Meetings über feste Zeitabschnitte von wenigen Wochen, Iterationen genannt, bis auf die Ebene der Auslieferungen.
- *APM* ist ein iteratives Vorgehen, d. h., der zeitliche Ablauf ist in gleich lange Iterationen von wenigen Wochen unterteilt, was einem zeitlichen Prüfraster mit festem Rhythmus entspricht.
- *APM* beinhaltet ein inkrementelles Vorgehen. In einer Iteration erarbeiten wir ein prüfbares Ergebnis von direktem Nutzen für den Kunden bzw. die Anwender, also ein Stück funktionierender Software. Das Ergebnis eines Projekts wird schrittweise erarbeitet. Ein Zwischenergebnis bezeichnen wir als Inkrement. Die Inkremente bauen aufeinander auf, sodass wir uns mit jedem

weiteren Inkrement dem Projektergebnis konkret nähern. Damit ein inkrementell-iteratives Vorgehen funktioniert, strukturieren wir unser angestrebtes Projektergebnis derart, dass wir ausreichend kleine, sinnvolle Teilstücke als Inkremente innerhalb einer Iteration umsetzen können.

- *APM* ist ein architekturzentriertes Vorgehen. Die Arbeit an der Softwarearchitektur findet kontinuierlich statt, wobei regelmäßig im Projektverlauf grundlegende Architekturfragen im Vordergrund stehen.

Diese fünf zentralen Säulen von *APM* sind eingebettet in eine Vielzahl von Best Practices aus unterschiedlichen Frameworks und Methoden. Im Wesentlichen sind das Scrum, eXtreme Programming (XP), Kanban, agiles Requirements Engineering und Softwarearchitektur (Abb. 1-1). In *APM* wenden wir nicht nur neueste Techniken an, sondern nutzen die Erfahrung aus mehreren Jahrzehnten mit agilem Projektmanagement, Lean Management und inkrementell-iterativen Vorgehensweisen.

Damit ist *APM* besonders dafür geeignet, komplexe Projekte mit einem oder auch mehreren parallel arbeitenden Teams durchzuführen, in denen eine besonders hohe innere und äußere Qualität der Softwareprodukte gefordert ist. Ein wesentlicher Erfolgsfaktor liegt darin, dass die konkrete Vorgehensweise und die Ausprägung von *APM* im laufenden Projekt regelmäßig betrachtet und verbessert werden. Erfahrungen aus dem laufenden Projekt wirken so direkt auf das Projekt selbst und können sofort genutzt sowie den Projektteams kommuniziert werden.

APM stellt ein flexibles Rollenmodell bereit, über das wir weitgehend unabhängig vom Spezialisierungsgrad der einzelnen Teammitglieder cross-funktionale *Featureteams* bilden können, die jeweils gemeinsam verantwortlich einen eigenständig nutzbaren Teil des Softwareprodukts durchgängig und vollständig umsetzen [99]. So entkoppeln wir in *APM* die einzelnen Teams voneinander, sodass gegenseitig induzierte Wartezeiten zwischen den Teams minimiert werden und sich ein kontinuierlicher Projektfortschritt einstellt.

APM kann man auch als umfangreichen Werkzeugkasten um einen im Kern an Scrum angelehnten agilen Ablauf verstehen. Der Werkzeugkasten ist randvoll gefüllt mit zueinander kompatiblen, sich ergänzenden bzw. aufeinander aufbauenden Techniken, aus denen Sie sich zur Lösung bestimmter Aufgaben, Probleme oder Fragestellungen bedienen können. Die einzelnen Zutaten stammen ihrerseits aus unterschiedlichen Ansätzen. So finden sich z. B. auch traditionelle Techniken z. B. in der Projektvorbereitung oder im Risikomanagement wieder. Was zueinander passt, sich ergänzt, funktioniert und nicht den agilen Werten widerspricht, findet sich in *APM* wieder.

Daher passt *APM* erfahrungsgemäß besonders gut in Umfelder mit einer breiten Erfahrung in traditionellem Projektmanagement oder wo zusätzliche Anforderungen wie gesetzliche Auflagen oder andere Regularien zu erfüllen sind. Wenn Sie mit Scrum und Kanban an Grenzen gestoßen sind, weil die laufende Entwicklung stark gewachsen und an Komplexität zugenommen hat und Sie dafür eine Reihe ergänzender Praktiken etablieren möchten, kann Ihnen *APM* ebenfalls weiterhelfen.

Das offene Werkzeugkastenkonzept spiegelt auch die aktuelle Realität wider, in der die Gräben zwischen der agilen und der traditionellen Projektwelt langsam zugeschüttet werden und beeinträchtigende Dogmen beiseite geschoben werden. Wir können nur voneinander lernen.

Auch wenn *APM* auf den ersten Blick durch seinen umfangreichen Werkzeugkasten besticht, so ist es doch wesentlich mehr. *APM* basiert komplett auf der Umsetzung der agilen Werte und Prinzipien aus dem Agilen Manifest. *APM* ist kein traditionelles Vorgehen, das um ein paar Techniken aus Scrum erweitert wurde. Es ist ein durchweg agiles Vorgehen, bei dem vorurteilsfrei geschaut wird, was bei bestimmten Fragestellungen hilfreich sein kann. Was passt, wurde assimiliert.

APM zeichnet sich durch eine Reihe von Merkmalen gegenüber den meisten agilen Frameworks aus. Im Wesentlichen sind das:

- *APM* integriert die aktuellen Ansätze zu einem agilen Requirements Engineering sowie zu einer Use-Case-Analyse und -

Modellierung und nutzt damit bewährte und verbreitete Techniken.
- Es ist kompatibel zu verbreiteten Rollenmodellen und kann daher ohne organisatorische Probleme schnell aufgesetzt werden. Es bietet damit auch für die Mitarbeiter fachliche Karrieremöglichkeiten, die bei anderen agilen Vorgehensmodellen oft vermisst wird.
- Softwarearchitektur ist in ihrer Umsetzung und in die Entwicklungsprozesse vollständig integriert, sodass sie den Stellenwert erhält, der für langfristig erfolgreiche Produkte notwendig ist.
- *APM* beinhaltet neben einem testgetriebenen Vorgehen das agile Testkonzept nach Crispin und Gregory [37], das weit über ein rein testgetriebenes Vorgehen hinausgeht und in dem die Entwicklung begleitende Tests von Anfang an für eine hohe Qualität sorgen.
- Wege zur angemessenen Skalierung wachsender Projekte werden mitgeliefert und müssen nicht mühsam entwickelt und integriert werden.

1.2 Ein paar grundlegende Begriffe vorab

Beim Erklären und Darstellen agilen Vorgehens stoßen wir, wie wir bereits im vorherigen Abschnitt gesehen haben, schnell auf ein rekursives Problem. Wir brauchen einerseits definierte Begriffe, um das Zusammenspiel in agilen Projekten zu erklären, und andererseits das Zusammenspiel in agilen Projekten, um diese Begriffe zu definieren.

Wir versuchen auch für dieses Problem eine iterative Lösung zu finden. Daher beginnen wir mit kurzen Definitionen einiger Begriffe, um danach gleich in die Grundlagen von Agilität vorzudringen.

Eine **Iteration** ist ein vorab festgelegter, fester Zeitraum, in dem wir ein Zwischenziel erreichen wollen. Iterationen dienen uns dabei, ein Projektergebnis schrittweise zu erreichen und diese Schritte zeitlich zu gliedern. Den Teil des Projektergebnisses, der im Laufe einer Iteration neu hinzukommt, nennen wir **Inkrement**. Typische Längen einer

Iteration liegen in *APM* zwischen zwei und vier Wochen. In der Regel gibt es in einem Projekt mehrere Iterationen, die direkt aneinander anschließen.

Die Anforderungen an ein Projektergebnis halten wir in einer Liste fest, dem **Backlog**. Wörtlich übersetzt bedeutet Backlog »Auftragsbestand« oder »Arbeitsrückstand«. Das Besondere an einem agilen Auftragsbestand ist seine Ordnung. Die einzelnen Aufgaben, die im Backlog stehen bzw. dort referenziert sind, haben wir eindeutig geordnet. Das wichtigste Ordnungskriterium ist dabei die Bedeutung des Eintrags für den Wert des späteren Projektergebnisses aus Kunden- bzw. Anwendersicht. Des Weiteren ist jeder Eintrag im Backlog geschätzt. Ein Backlog ist also eine geordnete Liste von geschätzen Aufgaben bzw. Anforderungen für ein Projektergebnis.

Die einzelnen geordneten Einträge im Backlog beschreiben die Anforderungen an das Projektergebnis bzw. seine fachlichen oder technischen Produktmerkmale. Eine weitverbreitete und praktikable Form der Beschreibung eines Eintrags im Backlog ist die sogenannte **User Story**. Um eine solche Anforderung umzusetzen, ist eine Reihe von Tätigkeiten durchzuführen, die **Tasks** genannt werden. Dieser Unterschied zwischen User Stories im Backlog und Tasks ist deshalb wichtig, weil wir formale Kriterien an einen Backlog-Eintrag stellen, die nicht für die Tasks gelten. Wir benötigen beide Sichten, Backlog-Eintrag und Task, um ein agiles Projekt steuern zu können oder das Projektcontrolling durchzuführen.

Einen Zwischenstand unserer kompilierten und zusammengebauten Software wird **Build** genannt. Mindestens einmal pro Tag bzw. Nacht erstellen wir einen Build auf Basis der aktuellen Versionsstände in unserer Konfigurationsverwaltung (Daily bzw. Nightly Build). Noch vorteilhafter sind kontinuierliche Build-Managementsysteme wie z. B. Jenkins. Lässt sich ein Build nicht erstellen, liegt ein sogenannter Build Breaker vor. Auf einem der letzten Builds innerhalb einer Iteration beruht dann das Inkrement. Dieser Build dient damit als Besprechungsgrundlage im Ergebnisreview. Ausgewählte Builds werden in einem **Release** freigegeben und an den Kunden als **Lieferung** (engl.: Delivery) ausgeliefert.

Am Ende einer Iteration finden zwei Meetings statt, das Ergebnisreview und die Retrospektive. Im **Ergebnisreview** demonstrieren wir Mitarbeitern der Auftraggeber- bzw. Kundenseite und anderen Stakeholdern den Fortschritt aus der gerade ablaufenden Iteration. Diese Gelegenheit für Rückmeldungen zum aktuellen Projektergebnis sind essenziell für die konkrete Planung der nächsten Iteration und darüber hinaus für die Möglichkeit, das Projektergebnis für den Kunden bezüglich Kosten, Termin oder anderer Kriterien maßzuschneidern.

Die Durchführung einer **Retrospektive** ist die letzte Aktion in einer Iteration. Wir nutzen sie, um auf den Projektverlauf und die Zusammenarbeit zurückzublicken, oft auftretende Muster zu erkennen, daraus Erkenntnisse abzuleiten und kurzfristig umsetzbare Maßnahmen zu initiieren, um den Projektverlauf zur nächsten Iteration weiter zu verbessern. Während im Review das außen sichtbare Projektergebnis im Zentrum der Betrachtung steht, liegt der Fokus der Retrospektive auf den inneren Abläufen im Projekt.

Zur Steuerung kommen noch zwei verwandte, aber doch in Wirkung und Einsatzbereich unterschiedliche Konzepte hinzu: Timebox und Meilenstein. Beide verknüpfen einen gepanten Inhalt mit einer Dauer bzw. einem Endtermin. Bei einer **Timebox** ist dabei der Termin fest. So definieren wir feste Prüfpunkte, an denen wir den erreichten Inhalt bewerten und ggf. für fehlende Teile den Restaufwand schätzen und deren Bearbeitung einplanen. Eine Iteration ist ein Beispiel für eine Timebox.

Ein **Meilenstein** definiert das Ende einer Phase oder eines Schrittes, zu dem ein bestimmter inhaltlicher Umfang erreicht sein soll. Ist der gewünschte Umfang nicht erreicht, wird der Restaufwand geschätzt und der Meilensteintermin entsprechend verschoben.

Das wird erst einmal für die Begriffserläuterung genügen. Einen ersten Überblick über das zeitliche Zusammenspiel der Begriffe sowie der dahinter stehenden Konzepte finden Sie in Abbildung 1-2. Die Vorgänger- und Folgeiterationen sind angedeutet und grau eingezeichnet. Als Vorgriff finden Sie eine Verteilung der Aufgaben

bzw. Verantwortlichkeiten auf Rollen in *APM*, die links in Abbildung 1-2 dargestellt sind.

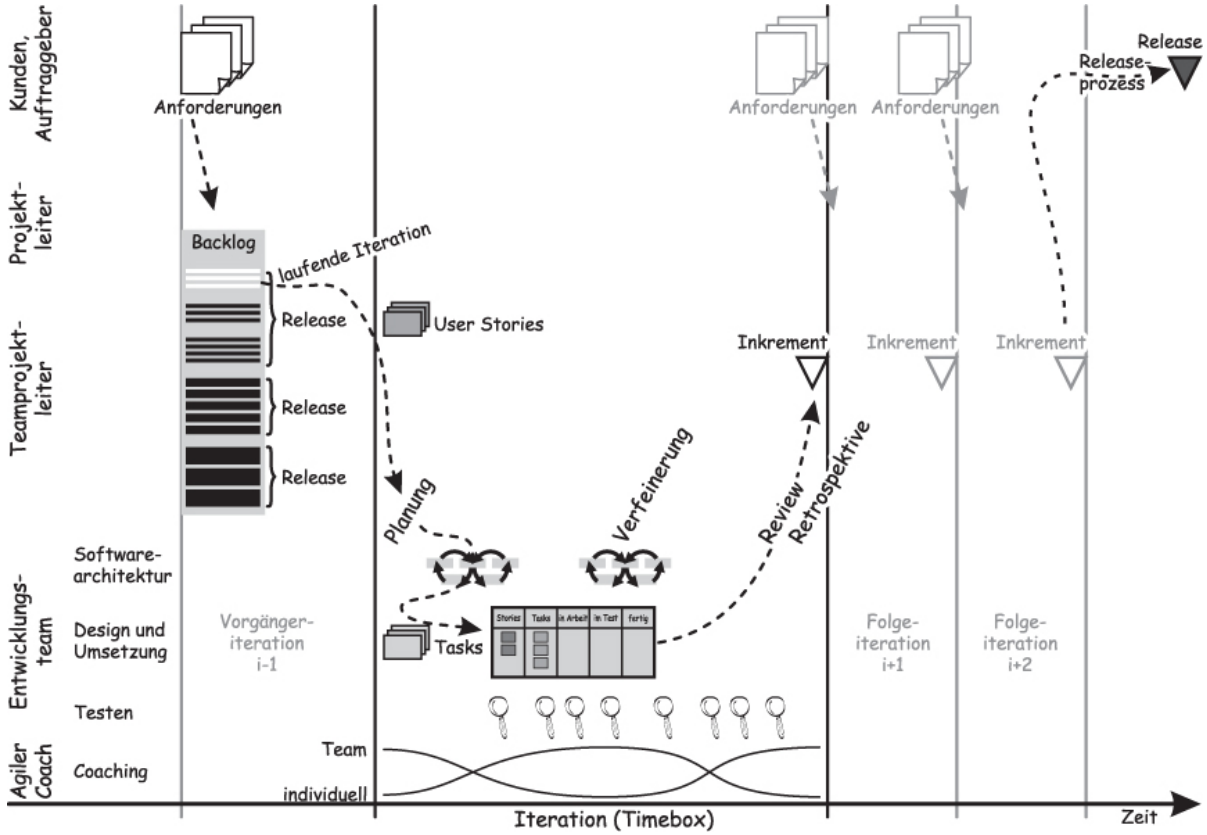

Abbildung 1-2: Das inkrementell-iterative Vorgehen in *APM*

Die **Planung** einer Iteration (engl. Planning) erfolgt in einer Timebox von wenigen Stunden. Sie kann und soll damit auch nicht vollständig sein. Um eine ausreichende Flexibilität zu erhalten, auch innerhalb einer Iteration auf unvorhergesehene Aspekte angemessen reagieren zu können, erfolgt in der zweiten Hälfte der Iteration meist eine weitere **Verfeinerung** der Planung (engl. Refinement) (Abb. 1-2).