Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Agil bis zur Unternehmensstrategie – ein »Big Picture«

»Lean« und »Agil« in einem Unternehmen einzuführen ist eine schwierige Aufgabe. Es bedeutet, Anforderungs- und Designspezifikationen und »Gate«-orientierte Steuerungsmodelle zu reduzieren oder gar zu eliminieren und stattdessen »work in progress limits« und Ähnliches einzuführen.

In diesem Beitrag erfahren Sie:
- etwas über das »Big Picture« des Skalierten Agilen Frameworks als Übersicht,
- etwas über dieses neue, agile, schlanke und dennoch voll skalierbare Modell in klarer Gestalt,
- einige Aspekte auf Team-, Programm- und Portfolioebene.

Dean Leffingwell (übersetzt von Hans-Peter Korn)

Dieser Beitrag beruht auf dem (vom Übersetzer im Oktober 2012 auf Basis von scaledagileframework.com aktualisierten) Inhalt des zweiten Kapitels des Buches »Agile Software Requirements« von Dean Leffingwell [1].

Prinzipien und Praktiken von »lean« und »agil« in einem Projektteam, für ein Programm oder auf Unternehmensebene einzuführen ist anspruchsvoll. Sogar die Sprache ist anders und irgendwie merkwürdig: »User Stories«, »Sprints«, »Velocity«, »Story Points«, »Epics«, »Backlog« …

Zusätzlich werden vertraute und bisher als unumstößlich geltende Praktiken wie etwa Spezifikationen, Abnahmen und Quality Gates durch neue, zunächst paradox erscheinende Praktiken und

»Skaliertes Agiles Framework« ist die in diesem Beitrag benutzte Übersetzung von »Scaled Agile Framework™«, einer geschützten Wortmarke des Autors Dean Leffingwell.

Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Prinzipien (wie beispielsweise die Begrenzung der Anzahl paralleler Arbeiten) ersetzt.

Selbst dann, wenn all diese Veränderungen vollständig begrüßt werden, kann es sechs Monate bis zu einem Jahr dauern, nur um die grundlegendsten neuen Praktiken einzuführen und wirksam werden zu lassen. Und noch viel länger dauert es, um jene signifikante Produktivitätssteigerung und deutliche Verbesserung der Qualität zu erreichen, die sich letztlich auf die Kundenzufriedenheit und den Ertrag auswirkt und einen größeren Marktanteil verspricht. Um diesen Nutzen zu erreichen, muss vieles verändert werden. Andererseits werden viele der bis jetzt existierenden Artefakte, Milestones usw. als Absicherung gegen die oft erlebten Probleme von Softwareprojekten als unverzichtbar betrachtet. Wir stehen also vor einem Dilemma: Wie schaffen wir diesen Hochseilakt ohne Netz, wenn das Netz – die bisherige Praxis – selbst das Problem darstellt?

Glücklicherweise haben bereits einige Organisationen diesen Wechsel hinter sich und es kristallisieren sich langsam etliche Erfolgsfaktoren des von »lean« und »agil« bestimmten Softwareentwicklungsprozesses heraus. In unseren Gesprächen mit Teams, Managern und Geschäftsführern während solcher Wechsel hatten wir oft Schwierigkeiten, die passende Sprache und die passenden Abstraktionen und Grafiken zu finden, um zu beschreiben, wie das Unternehmen nach einem solchen Übergang zur Agilität aussehen und funktionieren wird.

Denn dazu müssen wir die neuen Mechanismen der Softwareentwicklung und -auslieferung, die neu gestalteten Teams und Organisationseinheiten und einige neue Rollen der »Keyplayer« auf Basis des neuen Paradigmas beschreiben können. Ein solches Big Picture muss die Praktiken dieses neuen Modells beleuchten, denn seine Artefakte sind die Proxies der Softwareentwicklung, betrachtet als Wertschöpfungskette.

Letzten Endes konnten wir mithilfe anderer (speziellen Dank an Matthew Balchin und andere von Symbian Software Ltd. und Juha-Markus Aalto von Nokia Corporation) ein für diesen Zweck einiger-

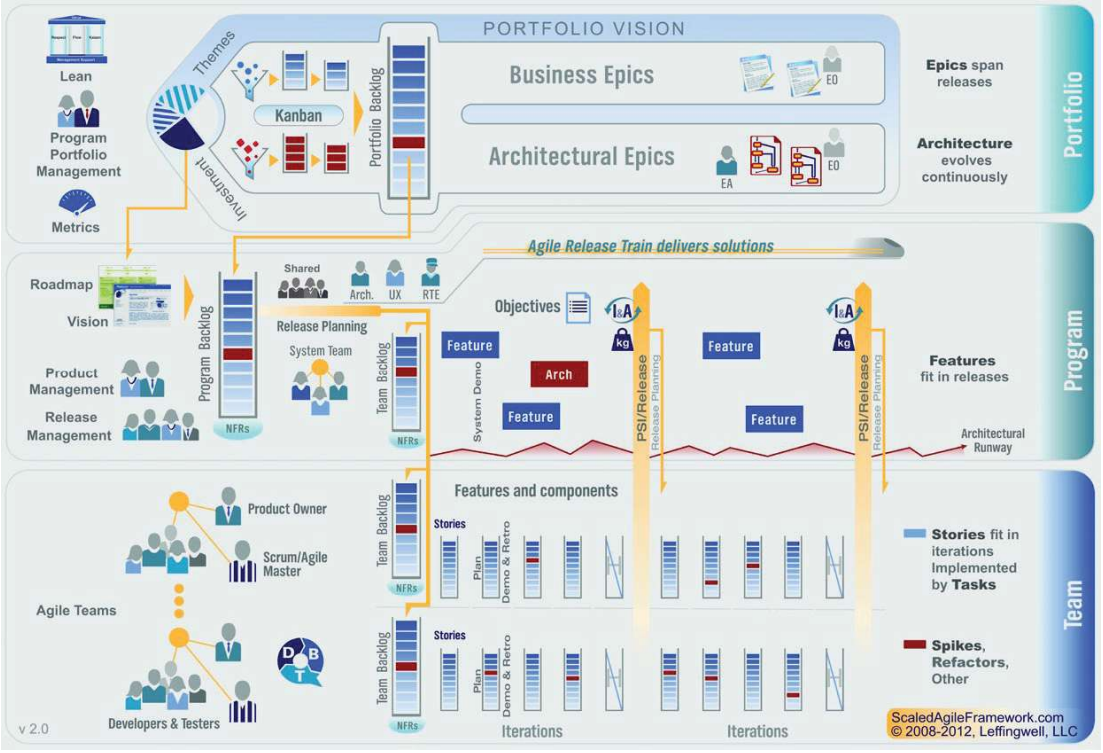

Abb. 1: *Das Agile Enterprise Big Picture*

maßen gut funktionierendes Bild entwickeln. Wir nennen es das Agile Enterprise Big Picture (siehe Abbildung 1).

Das Big Picture kurz erklärt

Als Erstes erklären wir das Big Picture aus der Vogelperspektive, damit der Leser eine kompakte Sicht dieses neuen, agilen, schlanken und dennoch voll skalierbaren Modells für das Management der Softwareanforderungen erhält.

Das Big Picture als Bestandteil dieses Buches zeigt die bei Drucklegung aktuelle Version 2.0. Die neueste Version finden Sie unter http://scaledagileframework.com in Form eines interaktiven »HIPper«-Dokuments als Wissensbasis zur Implementierung agiler Praktiken im gesamten Unternehmen. Jedes Icon des Big Pictures in http://scaledagileframework.com ist mit einem Link zu einer Beschrei-

bung in Form einer Kurzfassung und (falls bereits erarbeitet) einer Detailbeschreibung hinterlegt.

Das Big Picture aus der Vogelperspektive

Die Teamebene

Agile Teams mit 7 ± 2 Mitgliedern definieren, bauen und testen User Stories in eine Abfolge von Iterationen und Releases. In sehr kleinen Unternehmen gibt es allenfalls nur wenige solcher Teams. In großen Unternehmen hingegen arbeiten mehrere im »selben Boot sitzende« agile Teams gemeinsam an der Realisierung umfassender Funktionalitäten für komplette Produkte, Features, Komponenten, Subsysteme etc. Die jeweiligen Product Owner dieser Teams sind verantwortlich für das Managen des Backlogs bestehend aus User Stories und anderen Dingen, die das Team zu erledigen hat.

Die Programmebene

Auf der Programmebene erfolgt die teamübergreifende Koordination der Funktionalitäten umfangreicher Systeme auf Basis eines synchronisierten Agile Release Train (ART). Dieser ART ist eine festgelegte Abfolge stets gleich langer Iterationen und von Milestones mit fixierten Terminen und Qualitätsanforderungen, jedoch mit veränderlichem Umfang (kein »eisernes Dreieck«). Ergebnisse des ART sind häufige, in der Regel terminfixierte Releases oder Potentially Shippable Increments (PSIs) jeweils alle 60 bis 120 Tage. Diese überprüfbaren Inkremente können (müssen aber nicht) dem Kunden ausgeliefert werden. Die Auslieferung ist abhängig einerseits davon, ob der Kunde die Kapazität hat, das neue Produkt entgegenzunehmen, und andererseits von externen Ereignissen, die das Timing bestimmen.

Wir benutzen die allgemein gebräuchliche Bezeichnung Product Manager für jene Personen, die auf dieser Ebene die Features des Systems festlegen, sind uns aber bewusst, dass es viele andere Bezeichnungen für diese Rolle gibt.

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Die Portfolioebene

Beim »Portfolio« sprechen wir über die Gesamtheit jener Investment Themes, die die Investitionsprioritäten auf Unternehmensebene steuern. Mit diesem Konstrukt wird erreicht, dass die geleistete Arbeit für die Umsetzung der gewählten Geschäftsstrategie auch tatsächlich notwendig ist. Diese Investitionsthemen sind Treiber der Portfolio Vision, dargestellt als eine Zusammenstellung umfangreicherer Vorhaben in Form von Epics. Sie fließen im Lauf der Zeit in verschiedene Agile Release Trains ein.

In den weiteren Abschnitten dieses Beitrags werden einzelne zentrale Elemente des Big Picture beschrieben. Damit wird eine systemische Sicht auf unseren schlanken und agilen Anforderungsprozess geboten, der einerseits auf der Teamebene funktioniert, andererseits aber auch auf alle übergeordneten Anforderungen eines Unternehmens skalierbar ist.

Big Picture – Teamebene

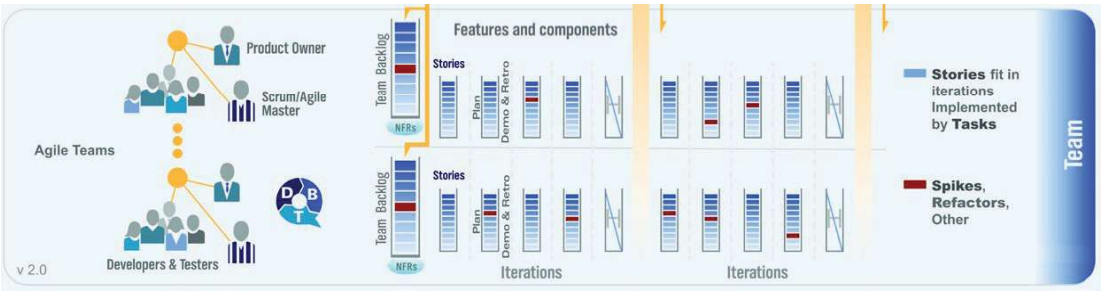

Abb. 2: *Teamebene des Big Picture*

Das agile Team

An vorderster Linie der Softwareentwicklung stehen die agilen Teams. Sie implementieren und testen Code und arbeiten zusammen am Gesamtsystem. Mit dem Team zu beginnen ist berechtigt: Denn beim agilen Vorgehen ist das Team das zentrale Element. Im Team wird jener Code geschrieben und getestet, der die für den Nutzer wertvollen Ergebnisse liefert. Ein agiles Team besteht aus 7 bis 9 Mitgliedern.

Abb. 3: *Das agile Team*

Es umfasst alle Rollen, die für das Definieren, Bauen und Testen seiner Software-Features oder Software-Komponenten nötig sind (siehe Kapitel 6 in [1]). Die Rollen sind u. a. der Scrum/Agile Master, der Product Owner und ein kleines Team ihm fest angehörender Entwickler, Tester und (hoffentlich) mit einem Experten der Testautomatisierung und möglicherweise auch einem »technical lead«.
In seiner täglichen Arbeit wird das Team von Architekten, externen QA-Leuten, Dokumentations- und Datenbank-Spezialisten, Mitarbeitern aus dem Supply-Chain-Management-/Build-/Infrastruktur-Bereich, dem internen IT-Betrieb und weiteren Personen als Shared Resources auf der Programmebene so unterstützt, dass das Team in der Lage ist, für das Gesamtsystem funktionierende Software zu definieren, zu entwickeln und zu testen.

Das Testen der Software ist ein unabdingbarer Bestandteil effektiv wertvoller Lieferungen – ungetesteter Code wird auf keinen Fall berücksichtigt. Deshalb sind die Tester Teil des Teams. Oft gehören sie zur Organisationseinheit »QA«, dennoch sollten sie auch dann physisch einem bestimmten agilen Team zugeordnet sein. In dieser Matrixorganisation sind sie primär Mitglied des Teams, als Mitarbeitende der QA-Gruppe haben sie aber unmittelbaren Zugriff auch auf andere Mitarbeitende und Manager der QA zur Kompetenzverbesserung, für Automatisierungsbelange und all die anderen spezifischen Testaspekte, die auf Systemebene erforderlich sind. Stets aber muss klar sein, dass das agile Team allein für die Qualität seiner Arbeitsergebnisse verantwortlich ist und dass es diese Verantwortung an keine andere Organisationseinheit innerhalb oder außerhalb des Unternehmens delegieren (oder sie gar aufheben!) kann.

Teams sind in der Regel so organisiert, dass sie entweder Features oder Komponenten der Software liefern. In den meisten Firmen werden wir eine Mischung davon antreffen: Einige Komponententeams kümmern sich um die gemeinsame Infrastruktur, Subsysteme und die

vielfach genutzten serviceorientierten Komponenten der Architektur und die Featureteams liefern vertikale, nutzerorientierte Lösungen. Agile Teams organisieren sich selbst und reorganisieren sich, wenn es die Aufgaben im Program Backlog erfordern. Mit der Zeit wird die Zusammensetzung der Teams eher dynamisch als statisch sein; statisch genug für die zur Teambildung (»norm, storm, perform« [2]) nötige Zeitdauer und dynamisch genug, um den sich ändernden Prioritäten des Unternehmens zu genügen.

Gruppen »im selben Boot sitzender« agiler Teams

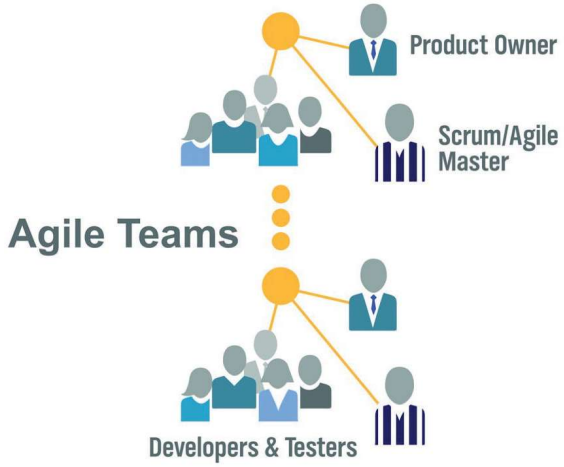

Abb. 4: *Gruppen »im selben Bott sitzender« agiler Teams*

In großen Unternehmen gibt es oft einige (3 bis 10) agile Teams, die gemeinsam ein größeres Feature, System oder Subsystem (siehe Programmebene im Big Picture) entwickeln. Es ist zwar keine feste Regel, aber eine häufige Erfahrung, dass – gerade bei sehr umfangreichen Systemen – die logischen Strukturen, die sich aus architekturorientierten System- und Produktfamilien ergeben, dazu führen, dass sich um diese verschiedenen Bereiche Entwicklergruppen organisieren. Etwa 50 bis 100 Personen arbeiten dann sehr eng zusammen, um ihr in der Produkthierarchie »nächstgrößeres Ding« zu realisieren. Das nennen wir ein »Programm«. Damit ist auch die Grenze einer »Face-to-Face«-Zusammenarbeit bei der Release-Planung erreicht.

Selbst wenn das eine allzu vereinfachende Sicht auf wirklich große Systeme ist, die eine Reihe solcher Programme umfassen, so trägt doch jedes Programm zum gesamten Portfolio (Produktportfolio, Anwendungslandschaft, Gesamtsysteme aus Systemen) bei.

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Rollen im agilen Team

Product Owner

Abb. 5: *Product Owner*

»Scrum« ist die gebräuchlichste agile Methode. Dort liegt auch der Ursprung der – wenn auch etwas willkürlichen – Definition der Rolle des Product Owners. Bei Scrum ist der Product Owner dafür verantwortlich, die Anforderungen zu ermitteln und zu priorisieren und das Product Backlog zu pflegen. Nach unserer Erfahrung kann selbst dann, wenn Scrum nicht verwendet wird, die Einführung der Rolle des Product Owners – im Sinne von Scrum – einen echten Durchbruch für die Vereinfachung der Teamarbeit bedeuten und bewirken, dass sich das ganze Team auf Basis des priorisierten Backlogs organisiert.
Allerdings endet die Verantwortung des Product Owners nicht hier. Im Sinne des vierten Prinzips des Agilen Manifests: »nutzerseitige Fachexperten und Entwickler müssen während des Projekts täglich zusammenarbeiten« hat der Product Owner seinen Arbeitsplatz im Teamraum, nimmt am Daily Meeting teil und beteiligt sich an den Arbeiten des Teams.

Scrum/Agile Master

Abb. 6: *Scrum Master*

Für Scrum-Teams ist der Scrum Master eine bedeutende – wenn auch eine gelegentlich vorübergehende – Rolle. Wenn das Team den agilen Prozess beherrscht, wird diese Rolle weniger wichtig. Einige sehr agile Teams, auch solche, die Scrum verwenden, haben keinen dezidierten Scrum Master. Jeder im Team kennt und beachtet die Spielregeln und das Team ist selbstbestimmt.

Der Scrum Master ist der im Team verankerte Repräsentant (»proxy«) des Managements und der Führung und hat die Aufgabe, das Team beim

Übergang zur neuen Vorgehensmethode zu unterstützen und die Entwicklung des Teams in Richtung Leistungsmaximierung zu begleiten. In agilen Teams, die nicht auf Basis von Scrum arbeiten, wird eine vergleichbare Führungsrolle üblicherweise vom Teamleiter, einem internen oder externen Coach oder vom Linienvorgesetzten des Teams wahrgenommen. Viele dieser Agile Master werden, wenn sie ihre Fähigkeiten weiterentwickeln, später zu Führungskräften und stellen dann ihre Fähigkeit unter Beweis, Werte für die Nutzer zu schaffen und die Verbesserung der agilen Praktiken fortlaufend voranzutreiben.

Entwickler und Tester

Abb. 7: *Entwickler und Tester*

Das restliche Kernteam besteht aus Entwicklern und Testern. Sie schreiben und testen Code. Bei agilen Teams ist die Teamgröße üblicherweise auf 3 bis 4 Entwickler plus 1 bis 2 Tester begrenzt. Sie arbeiten – im Idealfall – im selben Raum gemeinsam, um Stories zu definieren (D), zu bauen (B), zu testen (T) und in die schon vorhandene Codebasis zu integrieren.

Iterationen

Beim agilen Entwickeln werden neue Funktionalitäten in kurzen Zeitabschnitten (»Timeboxes«), sogenannten Iterationen (bei Scrum die Sprints), realisiert. In größeren Unternehmen einigen sich die Teams in der Regel auf eine standardisierte Dauer der Iterationen und synchronisieren deren Beginn und Ende, damit die Reife des Codes quer über alle Teams bei jedem Iterationsende als Systemintegrationspunkt vergleichbar ist.

Jede Iteration liefert ein nutzbares Inkrement einer neuen Funktionalität mithilfe eines stetig wiederholten (also iterierten) Standardmusters: Planen der Iteration – Bauen und Testen der Stories – Demons-

tration der neuen Funktionalität bei den Stakeholdern – »inspect and adapt« – Wiederholen.

Die Iteration ist für das Team der »Herzschlag der Agilität« und die Teams sind nahezu vollständig darauf konzentriert, neue Funktionalitäten in diesen kurzen »Timeboxes« zu entwickeln. Im Big Picture ist die Iterationsdauer für alle Teams identisch. Das ist das einfachste Organisations- und Steuerungsmodell. Es gibt zwar keine vorgeschriebene Länge, die meisten Teams praktizieren jedoch die empfohlene Länge von zwei Wochen.

Anzahl der Iterationen pro »Release«

Eine Abfolge von Iterationen dient dazu, umfangreichere, systemumspannende Funktionalitäten für ein Release (oder ein potenzielles Release) für den externen Nutzer zusammenzufassen. Im Big Picture werden pro Release vier Entwicklungs-Iterationen (dargestellt mit jeweils einem vollen Iterations-Backlog) gezeigt, gefolgt von einer fünften »hardening« (oder Stabilisierungs-)Iteration (dargestellt mit einem leeren Backlog) als Abschluss des Release-Inkrements.

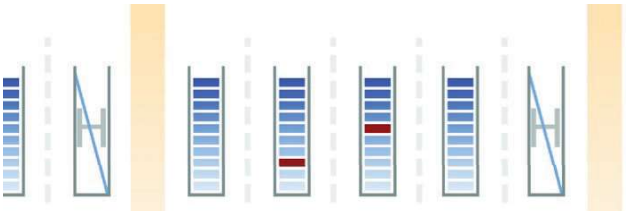

Abb. 8: *Iterationen pro Release*

Dieses Muster ist willkürlich. Es gibt keine feste Regel, wie viele Iterationen zu einem Potentially Shippable Increment (PSI) führen. Etliche Teams jedoch praktizieren 4 bis 5 Entwicklungsiterationen und eine »Hardening Iteration« (Stabilisierungsiteration) pro Release und liefern ein Shippable Increment jeweils alle 90 Tage. Das ist ein recht vernünftiger Produktionsrhythmus entsprechend einer den Kunden angemessenen Frequenz externer Releases und er passt gut zur üblichen Vierteljahresplanung auf Unternehmensebene.

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Stets aber entscheidet das Unternehmen über die Dauer und Anzahl der Iterationen pro Release-Inkrement und den Zeitpunkt, wann ein Release-Inkrement ausgeliefert wird.

User Stories und Team Backlog

User Stories

User Stories (kurz: »Stories«) entsprechen in der agilen Welt dem, was traditionell als »Software Requirement« bezeichnet wird.

User Stories, ursprünglich als Konstrukt von XP (Extreme Programming) entwickelt, sind heute ein untrennbarer Bestandteil der agilen Entwicklung im Allgemeinen und werden in Schulungen für Scrum, XP und für die meisten anderen agilen Ansätze angeboten. Bei der agilen Entwicklung sind die User Stories die wichtigsten Objekte, um die Nutzeranforderungen durch die gesamte Wertschöpfungskette der Softwareentwicklung von der Anforderungsanalyse bis zur Codierung und Implementierung zu transportieren.

Im Gegensatz zu »Requirements« (die im verbreiteten Verständnis das repräsentieren, wie das System eine Geschäftsanforderung oder vertragliche Verpflichtung zu erfüllen hat) sind User Stories kurze Absichtserklärungen, die beschreiben, was das System für einen Nutzer tun sollte. Üblicherweise wird die User Story als Aussage aus der Sicht des Nutzers formuliert:

Als ›Rolle des Nutzers‹ *kann ich* ›Aktivität‹, *damit* ›geschäftlicher Nutzen‹.

Durch diese Form lernt das Team, seinen Fokus auf den Geschäftsnutzen der neuen Funktionalität für eine bestimmte Nutzerrolle zu legen. Dieses Konstrukt ist von zentraler Bedeutung für die agile Intention, Nutzen zu liefern.

Team Backlog

Das Backlog des Teams (in der Regel Projekt- oder Produkt-Backlog genannt) enthält alle Stories, die das agile Team für die Implementierung identifiziert hat. Jedes Team hat sein eigenes Backlog. Es wird vom Product Owner des Teams gepflegt und priorisiert. Das Team

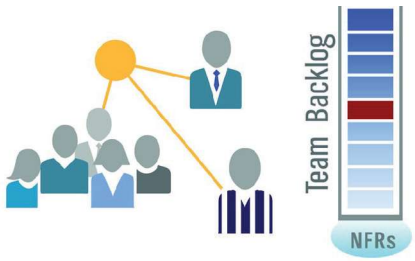

Abb. 9: *Team Backlog*

ist in erster Linie fokussiert auf die im gerade laufenden Sprint zu implementierenden User Stories, ungeachtet dessen, dass im Backlog des Teams auch anderes wie z. B. Fehlerbehebungen, Refactorings und Infrastrukturarbeiten enthalten sein können:

Das Identifizieren, Pflegen, Priorisieren, Planen, Entwickeln, Implementieren, Testen und Abnehmen der User Stories ist der primäre Prozess des Requirement Managements eines agilen Unternehmens.

Tasks

Um alle zur Lieferung einer Story nötigen Aktivitäten detaillierter verfolgen zu können, zerlegen die Teams die Stories in der Regel in jene Tasks, die von einzelnen Teammitgliedern bearbeitet werden müssen, damit die Story insgesamt erledigt werden kann. In einigen agilen Trainings werden die Tasks (und nicht die Stories) als Basisobjekte für das Schätzen und für die Fortschrittsverfolgung betrachtet.

Dennoch: Die Fortschrittsverfolgung innerhalb einer Iteration als solcher sollte stets auf den Stories als Ganzen beruhen. Damit bleibt das Team eher auf den Geschäftsnutzen fokussiert als auf die einzelnen Tasks. Tasks hingegen liefern eine granulare Aufteilung der Aufgaben, die den Teams möglicherweise die Koordination, das Schätzen, die Fortschrittsverfolgung und die Aufteilung der individuellen Verantwortung zur Erledigung der einzelnen Stories ermöglicht und damit die Erledigung der Iteration erleichtert.

Big Picture – Programmebene

Auf der Programmebene finden sich zusätzliche organisatorische Konstrukte, Rollen, Prozesse und Anforderungsartefakte, die auf die Realisierung umfassender Systeme, Anwendungen, Produkte und Produktfamilien abgestimmt sind.

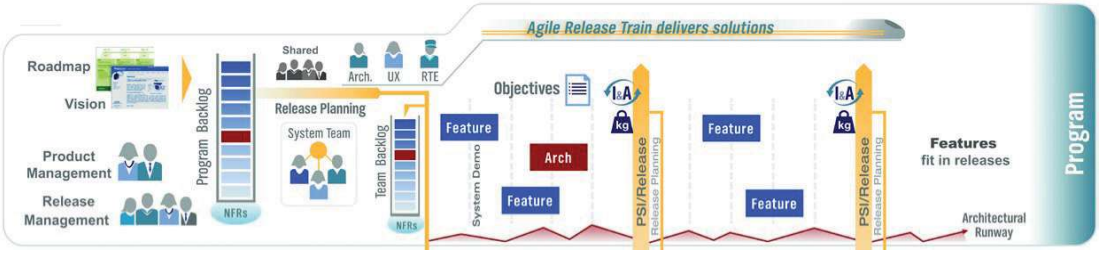

Abb. 10: *Die Programmebene des Big Picture*

Agile Release Train

Der »Agile Release Train« (ART) ist der wichtigste Skalierungsmechanismus des Frameworks. Auf der Programmebene ist ein Release des ART das, was eine Iteration für ein Team bedeutet. Er folgt demselben Muster: Planen – Vereinbaren – Ausführen – Vorführen – Reflektieren. Auf Programmebene entstehen mit dem ART pro Release (nach jeweils vier bis fünf Iterationen) »Potentially Shippable Increments« der Software, auf der Ebene der Iterationen entsteht alle zwei Wochen funktionierende Software auf Systemebene.

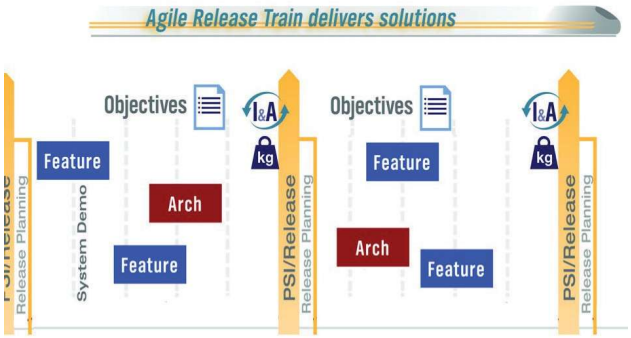

Abb. 11: *Agile Release Train*

Um das zu erreichen, skalieren wir

- ⇨ das Backlog, indem wir Features und nichtfunktionale Anforderungen (»Nonfunctional Requirements (NFR)«) als Artefakte der übergeordneten Ebene einführen,

Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

- die Teams, indem wir die Rollen Release Train Engineer, Product Management, System Team, Release Management, UX Designer und System Architect einführen sowie
- die Timebox, indem wir ein Release-Intervall von 8 bis 10 Wochen einführen.

Die Gliederung in Release Trains orientiert sich an den signifikanten Wertschöpfungsketten des Unternehmens, also an jenen langlebigen Wertströmen, bei denen eine optimierte Produktentwicklung und Lieferstrategie den größten wirtschaftlichen Nutzen bringt.

Aus Sicht des Lean Managements richtet der ART die Teams, die »im selben Zug sitzen« auf ein gemeinsames Ziel und einen gemeinsamen Zeitplan und Lieferrhythmus – die Haltestellen des ART – aus und unterstützt so den gleichmäßigen Fluss der Produktentwicklung. Anzumerken ist, dass – obwohl er »Agile Release Train« heißt und die Teams oft im Rhythmus der »Haltestellen« releasen – es sich eigentlich um einen »Agile Development Train« handelt und die Programme (und die einzelnen Teams eines Programms) jegliches Ergebnis zu jeder beliebigen Zeit releasen können, wenn die Voraussetzungen des Marktes, der Qualität und anderer Kriterien der Produktsteuerung erfüllt sind. In anderen Worten: Die Teams sind nicht daran gebunden, »externe« Releases nur am Ende einer Releasebearbeitungsperiode zu liefern. Sie können das seltener oder häufiger tun, so wie es der Markt erfordert. Siehe dazu auch Abb. 6 im Beitrag von Hans-Peter Korn: »Redesigned Agile in großen Unternehmen« im vorliegenden Buch.

Releases und »Potentially Shippable Increments«

Obwohl es das Ziel jeder Iteration ist, ein auslieferbares Software-Inkrement zu erzeugen, erachten es Teams – insbesondere in großen Unternehmen – oft als schlichtweg undurchführbar oder unangemessen, am Ende jeder Iteration ein Inkrement auszuliefern.

So kann ein Team im Verlauf mehrerer Iterationen etliche »technische Schulden« anhäufen, die vor der Auslieferung des Inkrements

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

eliminiert werden müssen. Technische Schulden sind beispielsweise zu behebende Fehler, größere oder kleinere Refactorings, aufgeschobene systemweite Tests der Performance oder der Zuverlässigkeit sowie die Einhaltung von Standards oder der Abschluss der Benutzerdokumentation.

»Hardening Iterations« werden im Big Picture auf der Teamebene eingeführt (dargestellt als Iterationen mit einem leeren Backlog), um Zeit für diese zusätzlichen Aktivitäten zur Verfügung zu haben. Zusätzlich gibt es berechtigte Geschäftsinteressen, die dagegen sprechen, dem Kunden jedes Inkrement auszuliefern. Dazu gehören

- ⇨ mögliche Konflikte mit Lizenz- und Serviceverträgen des Kunden,
- ⇨ möglicher Overhead und Geschäftsunterbrechung auf Kundenseite infolge von Installationen, Nutzerschulungen usw. sowie
- ⇨ das Risiko der Störung des kundenseitigen operativen Betriebs infolge kleinerer Installations-Rollbacks oder Fehler.

Aus diesen und anderen Gründen fassen die meisten Programme eine Reihe von Iterationen zu »Potentially Shippable Increments« zusammen, die dann ausgeliefert werden, wenn es zur jeweiligen Geschäftssituation passt.

Vision, Features und das Program Backlog

Im Unternehmen ist vor allem das Produktmanagement (oder das Programmmanagement oder die Business Analyse) verantwortlich für die Vision der Produkte, Systeme oder Anwendungen ihres Zuständigkeitsbereichs.

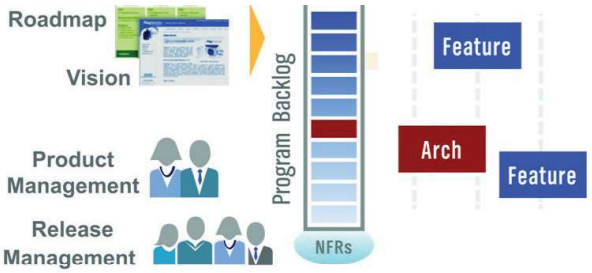

Abb. 12: *Vision, Features und Program Backlog*

Die Vision beantwortet folgende zentrale Fragen für das System, die Anwendung oder das Produkt:

- ⇨ Welches Problem wird damit gelöst?
- ⇨ Welche Features und welchen Nutzen liefert die Lösung?
- ⇨ Wem dient sie?
- ⇨ Wie gut ist die gelieferte Performance, Zuverlässigkeit usw.?
- ⇨ Welche Plattformen, Standards, Anwendungen usw. unterstützt sie?

Die Vision besteht in erster Linie aus einem Satz von Features

Die Vision kann in einem Dokument, in einer Backlog-Liste oder als simple Information oder Präsentation festgehalten werden. Unabhängig von der Form beinhaltet das Visionsdokument insbesondere eine priorisierte Zusammenstellung von Features, die den Anwendern Nutzen liefern.

Nichtfunktionale Anforderungen

Zusätzlich muss die Vision auch diverse nichtfunktionale Anforderungen für die Erfüllung der Zielsetzung des Systems enthalten, beispielsweise für Zuverlässigkeit, Genauigkeit, Performance, Qualität, Kompatibilitätsstandards usw.

Noch nicht implementierte Features bilden das Program Backlog

Ähnlich wie beim Backlog des Teams, das vor allem Stories enthält, besteht das Program (oder Release) Backlog aus den noch nicht implementierten, künftig aber benötigten und priorisierten Features. Das Program Backlog kann auch Schätzungen für den Umfang der Features

enthalten. Allerdings sind alle Schätzungen auf dieser Ebene grobgranular und unpräzise und es sollte jeglicher Versuchung widerstanden werden, allzu viel in die zu frühe Definition und Schätzung von Features zu investieren.

Releaseplanung

Jede Timebox für ein Releaseinkrement beginnt – in Übereinstimmung mit den agilen Praktiken – mit einem Planungstreffen für das Kick-off des Releases. Das Unternehmen nutzt dieses Kick-off, um für das Releaseinkrement den betrieblichen Kontext zu klären und die Teams auf die Geschäftsziele des Releases auszurichten. Die Vorgabe für das Releaseplanungstreffens ist die aktuelle Vision zusammen mit den Zielen des kommenden Releases und der Zusammenstellung der dafür benötigten und priorisierten Features.

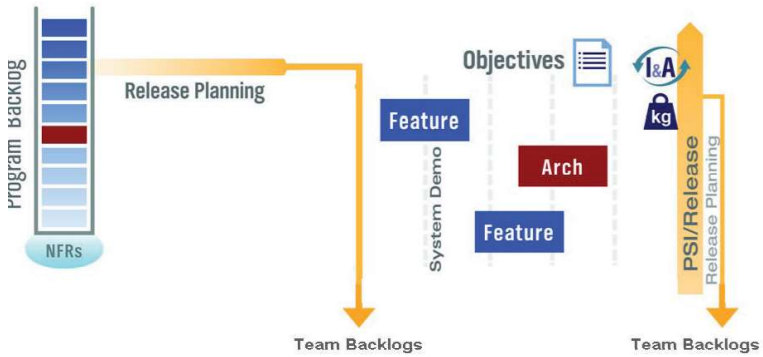

Abb. 13: *Releaseplanung*

Die agilen Teams planen das Release üblicherweise als zweitägigen (Groß-)Gruppen-Workshop aller Mitglieder der zum nächsten PSI beitragenden Teams am Ende des »Hardening-Sprints« des vorangehenden Releases. Sie zerlegen die Features in Stories und stützen sich dabei auf die vereinbarte Iterations-Kadenz und die ihnen bekannte »velocity«. Während dieses Prozesses klären die Teams ihre Schnittstellen und entwerfen den Releaseplan, indem sie die Stories den verfügbaren Iterationen der Release-Timebox zuordnen. Sie verhandeln in Kenntnis dessen, was sie leisten oder nicht leisten können (auf Basis ihrer be-

kannten Velocity und der Umfangsschätzungen für die neuen Stories), auch über Veränderungen des Releaseumfangs mit dem Produktmanagement. Zusätzlich zu dieser Planung verpflichten sich die Teams als weiteres wichtiges Resultat dieses Prozesses auf die zu erreichenden Releaseziele und priorisierten Features.

Die Teams setzen sich im Anschluss daran dafür ein, die wichtigsten Ziele des Releases – für die sie sich ja verpflichtet haben – zu erreichen. Und das auch dann, wenn sich herausstellt, dass nicht jedes einzelne Feature zeitgerecht fertig wird.

Die Ergebnisse der Releaseplanung fließen in die Aktualisierung der Produkt- oder Lösungs-Roadmap ein. Diese liefert eine Übersicht darüber, welchen zunehmenden Nutzen das Unternehmen auf der Zeitachse realisiert.

Die Roadmap

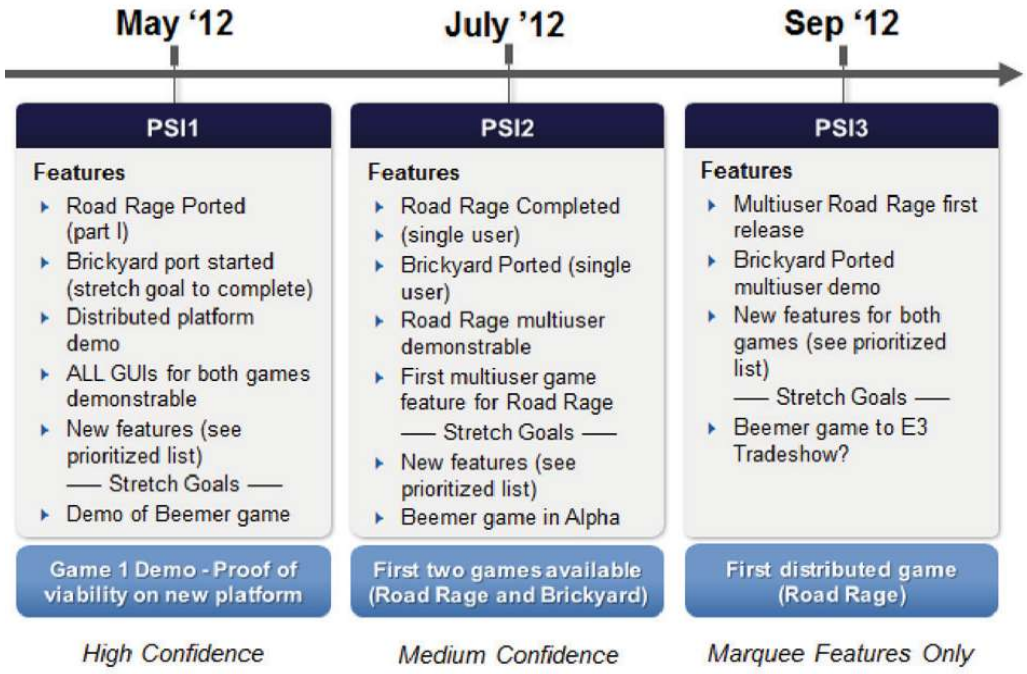

Abb. 14: *Roadmap*

Die Roadmap zeigt pro zukünftig geplanten Releasetermin das Thema, die jeweiligen Ziele und die geplanten priorisierten Features. Die Beschreibung des unmittelbar nächsten Releases der Roadmap stellt eine

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Verpflichtung gegenüber dem Unternehmen, basierend auf den Ergebnissen des letzten Releaseplanungstreffens, dar. Für die Ausgestaltung der späteren Releases gibt es keine verbindlichen Zusagen, deren Umfang ist bestenfalls unscharf.
Die Roadmap repräsentiert in Form eines Plans, was das Unternehmen für die aktuellen und künftigen Releases beabsichtigt. Sie ist stets Änderungen unterworfen, da sich Entwicklungsergebnisse, Geschäftsprioritäten und Kundenbedürfnisse ändern können. Deshalb sollen Pläne für Releases nach dem unmittelbar nächsten grundsätzlich nicht für unternehmensexterne Verpflichtungen genutzt werden.

Produktmanagement

Abb. 15: *Produktmanagement*

Beim agilen Vorgehen kann die offensichtliche Überlappung der Verantwortlichkeiten des Produktmanagers mit jenen des Product Owners eine Herausforderung sein. So ist in Scrum der Product Owner »für die Maximierung des Wertes des Produkts … verantwortlich. … Der Product Owner ist als einzige Person für die Verwaltung des Product Backlog verantwortlich. … nur dem Product Owner (ist es) erlaubt, die Fertigstellungsreihenfolge der Einträge des Product Backlog zu verändern … Die Entscheidungen des Product Owners müssen von der gesamten Organisation respektiert werden« [3].

In manchen kleineren Organisationen ist diese Beschreibung gut anwendbar und ein oder zwei Product Owner reichen aus, um alle Anforderungen zu definieren und zu priorisieren. In größeren Unternehmen jedoch sind die Verantwortlichkeiten eines Product Owners i. S. von Scrum oft so breit angelegt, dass sie auf die Teams, auf technologisch orientierte Product Owner und die auf den Markt oder auf das Programm fokussierten Produktmanager verteilt sind. Letztere nehmen ihre Verantwortlichkeiten traditionell sowohl bezüglich der Produkt-

Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

definition als auch der Präsentation der Lösung gegenüber dem Markt wahr. Auch stellen wir fest, dass die Bezeichnung für diese Rolle je nach Art des Unternehmens unterschiedlich ist (Tabelle 1):

Tabelle 1: Unterschiedliche Titel für die Rolle des Produktmanagers

Art des Unternehmens	Übliche Rollenbezeichnung
Informationssysteme/ Informationstechnologie	Business Owner, Business Analyst, Project Manager, Program Manager
Embedded Systems	Product Manager, Project Manager, Program Manager
Unabhängiger Software-Lieferant	Product Manager

Verantwortlichkeiten des agilen Produktmanagers im Unternehmen

Unabhängig von der Bezeichnung der Rolle (wir verwenden weiterhin den Titel »Produktmanager«) nimmt die diese Rolle ausfüllende Person vor allem folgende Verantwortungen wahr:

1. Ownership der Programmvision und des Program (Release) Backlogs
2. Managen des Release-Umfangs
3. Pflegen der Produkt-Roadmap
4. Aufbau eines effektiven Produktmanager-/Product-Owner-Teams

Die »Architekturpiste« (Architectural Runway)

Die »Architekturpiste« (angedeutet mit der roten Zickzacklinie unten auf der Programmebene) repräsentiert den fortlaufenden Erhalt und Ausbau der technologischen Infrastruktur auf Basis der im Unternehmen verfügbaren Plattformen, um die wichtigsten Business Epics des Portfolio Backlogs (siehe den folgenden Abschnitt »Portfolioebene«) ohne allzu aufwendiges und Verzögerungen verursachendes Redesign implementieren zu können. Für das Etablieren einer solchen Architekturpiste muss das Unternehmen sowohl in ein kontinuierliches Refactoring und Erweitern der vorhandenen Plattformen investieren als auch jene neuen Plattformen aufbauen und in Betrieb nehmen, die für die jeweils neuen Geschäftsanforderungen benötigt werden.

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Big Picture – Portfolioebene

Die oberste Ebene des Big Picture bildet das Portfoliomanagement. Es umfasst jene Personen, Teams und Organisationsteile, die sich darauf konzentrieren, die Investitionen des Unternehmens mit seiner Geschäftsstrategie in Einklang zu bringen. Wir finden auf dieser Ebene auch zwei neue Arten von Artefakten, die Investment-Themen und die Epics, die zusammen die Portfolio-Vision ergeben.

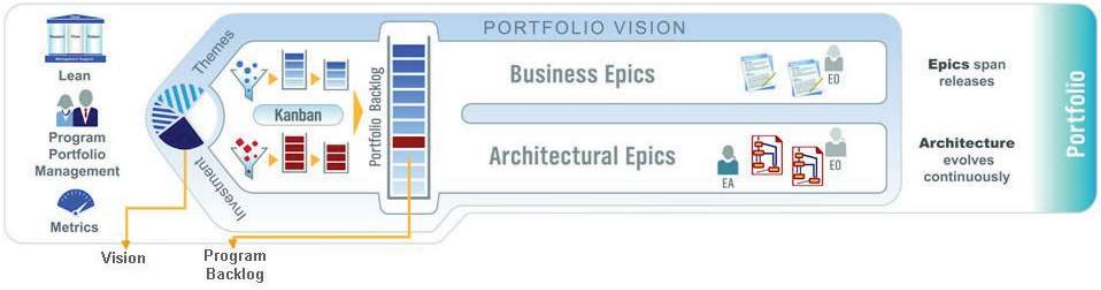

Abb. 16: *Portfolioebene des Big Picture*

Investment-Themen

Die Investment-Themen umfassen die Investitionsziele des Unternehmens oder eines Geschäftsbereichs. Diese Themen sind die Treiber der Vision für alle Programme. Zukünftige Epics werden aus diesen Themen abgeleitet. Die Verantwortung für die Erarbeitung der Themen und Epics tragen die Portfoliomanager. Das sind entweder die Verantwortlichen für die Geschäftsbereiche oder die Produktgremien und andere, die gegenüber ihren Stakeholdern treuhänderisch verantwortlich sind.

Die vom Portfoliomanagement erarbeiteten Themen erlauben es, auf dem Markt differenzierte und konkurrenzfähige Kernprodukte anzubieten. Diese Themen haben eine weitaus längere Lebensdauer als die Epics. Einzelne Themen können bis zu einem Jahr oder länger unverändert bleiben.

Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Epics und Portfolio Backlog

Im Framework werden zwei Arten von Epics unterschieden: Business-Epics und Architektur-Epics.

Business-Epics sind quasi verdichtete Nutzerbedürfnisse. Architektur-Epics repräsentieren große technologische Vorhaben als Voraussetzung für die Entwicklung umfassender Lösungen, welche die laufenden und zukünftigen Unternehmensbedürfnisse unterstützen. Anforderungen und Design (Architektur) sind die zwei Seiten derselben Münze, das »Was« und das »Wie«. Im Buch »Agile Software Requirements« [1] wird dieses Thema ausführlich behandelt. Jene Unternehmen, die über so etwas wie eine – weiter oben bereits beschriebene – sorgfältig gepflegte »Architekturpiste« verfügen, werden die Gewinner im Markt sein. Deshalb ist die Systemarchitektur ein »Ehrenbürger« im Big Picture und erfordert in einem agilen Unternehmen regelmäßige Investitionsüberlegungen.

Epics sind Entwicklungsvorhaben zur Lieferung wertschaffender Ergebnisse für ein Investment-Thema und werden im Portfolio Backlog festgehalten, priorisiert, geschätzt und laufend gepflegt. Vor der Releaseplanung werden sie in spezifische Features aufgeteilt, aus denen ihrerseits detailliertere Stories für die Implementation abgeleitet werden. Epics können dargestellt werden in Stichworten, in Form nutzersprachlicher »Stories«, in ein oder zwei Sätzen, als Video, als Prototyp oder in irgendeiner anderen Form, die geeignet ist, die Absicht der Produktinitiative auszudrücken. Ziel des Epics ist die Beschreibung der strategischen Absicht, nicht die Genauigkeit. Mit anderen Worten: Ein Epic muss nur so weit im Detail beschrieben werden, als damit eine zukünftige Diskussion darüber ausgelöst werden kann, welche Features ein Epic impliziert.

Bussines Epics werden zunächst im Kanban-System der Business Epics, Architektur-Epics zunächst im Kanban-System der Architektur-Epics erfasst. In beiden Kanban-Systemen erfolgt die weitere Bearbeitung unter Einhaltung der für Kanban typischen »Work in Progress(WIP)«-Begrenzungen.

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Literatur

[1] Dean Leffingwell: *Agile Software Requirements – Lean Requirements Practices for Teams, Programs, and the Enterprise (Agile Software Development). Addison Wesley, 2010*

[2] Bruce Tuckman: *en.wikipedia.org/wiki/Forming-storming-norming-performing*

[3] Jeff Sutherland, Ken Schwaber: *Scrum Guide 2011; http://www.scrum.org/Scrum-Guides*

Dean Leffingwell: Agil bis zur Unternehmensstrategie - ein "Big Picture"
Beitrag für Buch: Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez ISBN 978-3-86329-442-7 Symposion Publishing, Düsseldorf 2013

Zusammenfassung

In diesem Beitrag haben wir das »Big Picture« des Skalierten Agilen Frameworks (SAF) als grundlegendes Anforderungsartefakt und als Basismodell für den Prozess und die Organisation eines schlanken und agilen Managements von Softwareanforderungen vorgestellt.

Einerseits beschränkt sich dieses Framework auf der Ebene des agilen Teams auf jenes Minimum an Artefakten, Rollen und Praktiken, welche für die effektive Arbeit des Teams nötig sind.

Andererseits bietet dieses Framework die auf der Programm- und Portfolioebene erforderlichen Erweiterungen, die jedoch stets so schlank wie möglich gehalten werden. Damit sollen – auch im Fall aggregierter Softwareanforderungen eines »Teams von Teams« – die Anforderungen für immer größere »Systeme von Systemen« managebar bleiben.

Agiles IT-Management in großen Unternehmen

Agiles IT-Management in großen Unternehmen
Hrsg.: Hans-Peter Korn, Jean Pierre Berchez
Hardcover, ca. 400 Seiten
ISBN 978-3-86329-442-7
Preis 69,– EUR (inkl. MwSt. und Versandkosten)
Symposion Publishing, Düsseldorf
In Vorbereitung, Buch erscheint Ende 2012
www.symposion.de

Über dieses Buch

Agiles IT-Management und agile Softwareentwicklung sind en vogue. Scrum, das populärste Rahmenwerk der agilen Produktentwicklung, verzeichnet ein steigendes Interesse, gerade bei großen SW-Häusern und in den IT-Abteilungen größerer Unternehmen.

Der Charme des agilen Vorgehens und von Scrum ist groß, aber auch tückisch. Denn deren leicht verständliche Grundprinzipien wirken sehr »simpel«. Doch die konkrete Umsetzung in komplexen Systemlandschaften und Organisationen ist alles andere als einfach.

Das Problem: Die Grundprinzipien beziehen sich in der Regel auf ein Team, das ein Produkt in einer relativ homogenen Systemumgebung so entwickelt, dass einzelne Produktinkremente tatsächlich innerhalb weniger Wochen in Produktion gehen und genutzt werden können.

Die Skalierbarkeit von agilen Vorgehensweisen auf Multi-Team- bzw. Multi-Produkt-Situationen in großen Unternehmen wird dabei kaum thematisiert. Aspekte wie:

- ⇨ agiles Portfoliomanagement in großen Unternehmen
- ⇨ Programm Management in großen Unternehmen
- ⇨ agile Requirements
- ⇨ Skalierung und Koordination von Teams
- ⇨ teamübergreifende Continuous Integration

bleiben offen und führen zum Scheitern der ersten agilen Gehversuche. Oder zu Implementierungen, welche viele der gewohnten Praktiken hinter einer formal agilen Fassade weiter leben lassen.

Was agiles IT-Management in großen Unternehmen wirklich bedeutet und wie man die Herausforderungen, die sich bei der der Einführung des agilen Vorgehens ergeben, erfolgreich bewältigt, schildern die Autoren in diesem Fachbuch.

Über die Herausgeber

Dr. **Hans-Peter Korn** ist Inhaber der Schweizer Beratungsfirma KORN AG und arbeitet als Consultant, Coach und Trainer für agiles Management, Leadership und Teamwork. Als promovierter Physiker war er zunächst in verschiedensten Linien- und »klassischen« Projektleitungsfunktionen im Großanlagen-Engineering tätig, dann im Informatikbereich (kommerzielle Großsysteme der Finanz- und Transportindustrie). Seit 1998 konzentriert er sich auf die sozialen Aspekte von Veränderungs-, Kooperations- und Kommunikations-Prozessen in Unternehmen und komplexen Projekten.
Veröffentlichungen unter:
http://www.korn.ch/publi.html#Publikationen

Jean Pierre Berchez beschäftigt sich seit mehr als 20 Jahren mit den Themen Projektmanagement, Agile Entwicklung, Software Engineering, UML und Application Life Cycle Management. Er ist Autor diverser Artikel insbesondere mit dem Schwerpunkt Scrum und »Collaborative Software Development« und Mitgründer des Scrum-Day's, der Scrum Community Konferenz www.scrum-day.de. Des Weiteren ist Jean Pierre Berchez als Lehrbeauftragter am FOM, an den BA's Stuttgart und Heidenheim sowie an der Universität Lichtenstein tätig.
Veröffentlichungen unter:
http://www.scrum-events.de/whatisscrum/index.php

Fax-Bestellformular
Bitte ausfüllen und faxen an

Fax Nr. 02 11/8 66 93 23

Symposion Publishing GmbH
Münsterstr. 304
40470 Düsseldorf
Tel. 02 11/8 66 93 10

Ich bestelle zur frühestmöglichen Lieferung:

Agiles IT-Management in großen Unternehmen
ISBN 978-3-86329-442-7
Buch ist in Vorbereitung, erscheint Ende 2012
zum Preis von 69,– EUR (inkl. MwSt. und Versandkosten)

Lieferadresse: _____

_____

_____

_____

INHALT