Ausgabe 01/2013
www.agilereview.de

agile review

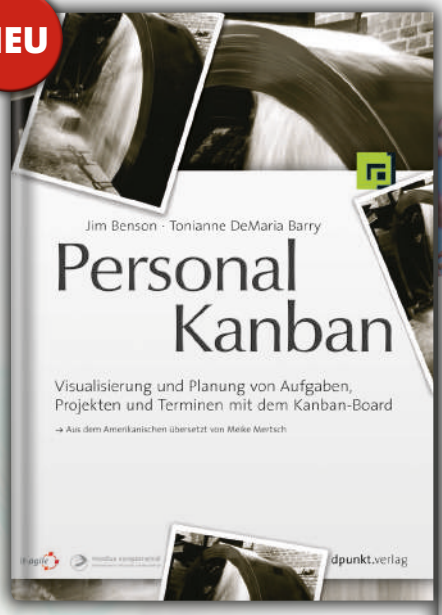
Das Kundenmagazin von it-agile

Wir lieben es, wenn ein Team funktioniert!

dpunkt.bücher für Softwareentwickler

dpunkt.verlag

NEU



2013, 164 Seiten · € 29,90 (D)
ISBN 978-3-89864-822-6

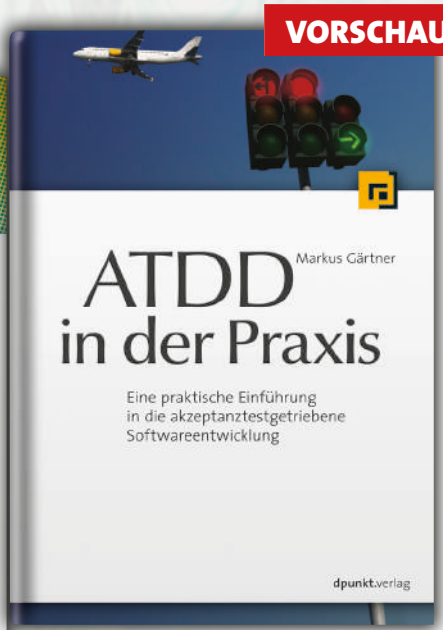
2011, 302 Seiten · € 34,90 (D)
ISBN 978-3-89864-730-4

2011, 216 Seiten · € 29,90 (D)
2. Auflage
ISBN 978-3-89864-701-4

2012, 222 Seiten · € 34,90 (D)
ISBN 978-3-89864-752-6

2011, 184 Seiten · € 32,90 (D)
ISBN 978-3-89864-719-9

VORSCHAU



03/2013, ca. 260 Seiten · € 32,90 (D)
ISBN 978-3-86490-046-4

dpunkt.verlag GmbH · Ringstraße 19 B · D-69115 Heidelberg
fon: 0 62 21 / 14 83 40 · fax: 14 83 99 · e-mail: bestellung@dpunkt.de

www.dpunkt.de

editorial

There is no i in team! It takes teamwork to make a dream work! 11 Freunde müsst ihr sein! Die Reihe an Sprüchen, die Teams über den grünen Klee loben, ließe sich fast endlos fortführen. Aber was genau sind die großen Vorteile von Teams? Bekommen wir sie umsonst oder welcher Preis ist dafür zu zahlen? Und wie sieht ein guter Teamschnitt aus? Diese Fragen werden besonders relevant, wenn man einmal hinter die Schlagworte schaut und überlegt, was es mit cross-funktionalen Teams eigentlich auf sich hat, wie Scrum sie propagiert.
Im Titel-Artikel auf Seite 14 beleuchten die Roock-Brüder cross-funktionale Teams aus unterschiedlichen Blickwinkeln und kommen dabei zu durchaus interessanten Erkenntnissen.
Ein weiterer Punkt, der sich zuerst simpel anhört, aber in der Praxis oft zu erheblichen Schwierigkeiten führt, betrifft die Rolle des Product Owners. In vielen Kontexten wirft insbesondere die Zusammenarbeit mit dem User Experience Design Probleme auf. Diesem Thema widmet sich Doreen Timm in ihrem Artikel auf Seite 38.
Natürlich beschäftigt sich die aktuelle agile review nicht nur mit Scrum. Die Frage, wie man Anforderungen sinnvoll klein schneidet, stellt sich für immer mehr Teams – unabhängig von der eingesetzten Methode. Hierfür bietet Dimensional Planning eine interessante Möglichkeit, wie der Artikel von Stefan Roock zeigt (Seite 4).
Im Interview ab Seite 26 erklärt Dave Snowden, was es mit komplexen adaptiven Systemen auf sich hat und was der Zweck des Cynefin-Frameworks ist. Etwas technischer wird es dann, wenn es darum geht, wie man Refactorings als Teamaktivität gestalten kann und wie sich problematische Codestellen anschaulich visualisieren lassen (Seite 44).
Die Bedeutung von Smartphone-Apps nimmt immer weiter zu. Dass die Erstellung von Apps ernsthafte Software-Entwicklung ist, beschreiben Holger Bohlmann und Sven Günther eindrucksvoll in ihrem Erfahrungsbericht zur Entwicklung der Top-5-iPhoneApp von mobile.de (Seite 34). Auch in dieser Ausgabe beschäftigen wir uns wieder mit Lean Startup. Wie man die Ideen von Eric Ries auch in bestehenden Unternehmen sinnvoll einsetzen kann, erfahren Sie auf Seite 50. Außerdem erwartet Sie eine kleine Sammlung nützlicher Low-Tech-Tools (Seite 22), und natürlich haben wir auch diesmal wieder Buchtipps für Sie zusammengestellt (Seite 12).
Aufgrund des positiven Feedbacks zu den früheren Ausgaben der agile review haben wir uns übrigens entschieden, ab sofort zwei Mal im Jahr zu „releasen“. Die nächste agile review wird dann wieder im September erscheinen. ■

Viel Spaß beim Lesen wünscht Ihnen

Dr. Arne Roock, Chefredakteur der agile review

PS: Dass es sehr wohl ein „i in Team“ gibt, beweist diese kleine Skizze ;-)

INHALT

Dimensional Planning

der Universalhäksler für User Storys, Iterationen und Releases

Von Stefan Roock

Kleine User Storys haben viele Vorteile. Leider ist nicht immer offensichtlich, wie man eine größere Anforderung in mehrere kleine User Storys zerlegt. Dieser Artikel stellt Dimensional Planning vor: eine relativ unbekannte, aber umso mächtigere Technik zum Zerlegen von User Storys, die sich auch auf Iterationen und Releases anwenden lässt – und sogar auf Organisationsentwicklung.

Buchtipps

X-teams: How to Build Teams That Lead, Innovate and Succeed

Deborah Ancona, Henrik Bresman (2007), Harvard Business Review Press, 260 Seiten.

Das Buch von Deborah Ancona und Henrik Bresman basiert auf Studien über 45 Produktteams, in denen die Autoren untersuchten, was Teams erfolgreich macht. Die Teams, die aktiv nach neuen Ideen außerhalb des Teams suchten, Feedback zu den eigenen Arbeitsergebnissen einholten, sich mit der Umwelt koordinierten und Managementunterstützung hatten, entwickelten schneller innovativere Produkte als die Teams, die sich nur um die effektive Zusammenarbeit im Team kümmerten.

Erfolgreiche Teams zeichnen sich also vor allem durch ihre Beziehungen zu ihrer Umwelt aus: Kontakte zu den Stakeholdern, wechselnde Teammitgliedschaften, umfangreiche externe Bindungen und erweiterbare Grenzen. Das Buch erläutert im Detail, was diese Eigenschaften jeweils bedeuten und wie sie hergestellt werden können.

Ancona und Bresman schreiben viel von dem, was z. B. auch in Scrum verankert ist, nur mit anderen Worten. Allerdings bringen sie eine zusätzliche Perspektive ein, die Fehlinterpretationen von Scrum deutlich macht. Denn diese mitunter beobachtbaren Auslegungen des Scrum-Frameworks widersprechen sehr deutlich dem, was Ancona und Bresman fordern:

- Product Owner, die alleine den Kontakt zur Außenwelt halten.
- ScrumMaster, die den Kontakt von Anwendern, Kunden und Managern zu dem Team unterbinden.
- Sprint-Reviews, in denen keine Anwender, Kunden oder Manager anwesend sind.
- Product Owner, die das Product Backlog in ihrem „stillen Kämmerlein" füllen, anstatt zusammen mit dem Team nach Ideen außerhalb des Teams zu suchen.

Das Buch ist daher gut geeignet, um über die eigenen Teams nachzudenken – ganz allgemein oder auch im Kontext agiler Methoden im Speziellen.

Stefan Roock

Was haben Ocean's Eleven, das A-Team oder auch die Ice-Age-Truppe rund um Manni, das Mammut, gemeinsam? Es handelt sich um interdisziplinäre - oder neudeutsch: cross-funktionale – Teams. Die Teammitglieder bringen unterschiedliche Fähigkeiten mit ein und ergänzen sich gegenseitig, um die anstehenden Aufgaben zu meistern und innovative Lösungen für ihre Probleme zu finden.
Was im Film gilt, gilt in diesem Fall auch in der Geschäftswelt: Wie Takeuchi / Nonaka bereits in einem Artikel von 1986 beschreiben (vgl. [1]), bieten cross-funktionale Teams im Business zwei bestechende Vorteile:
1) Geschwindigkeit und
2) Innovation durch gegenseitigen Austausch über Fachgrenzen hinweg („cross-fertilization").

Schneller sind crossfunktionale Teams deshalb, weil sie nicht auf Zuarbeit anderer Teams angewiesen sind und tendenziell alle Probleme selbst angehen und umgehend lösen können. Und Innovation entsteht häufig genau dann, wenn das Denken die ausgetretenen Pfade verlässt und ungewohnte Wege geht – und die Wahrscheinlichkeit hierfür ist prinzipiell umso höher, je mehr unterschiedliche Disziplinen zusammenarbeiten. In der agilen Welt, insbesondere in Scrum, spielen cross-funktionale Teams eine zentrale Rolle. Ken Schwaber und Jeff Sutherland schreiben dazu im Scrum Guide (vgl. [2]): „Entwicklungs-Teams sind interdisziplinär besetzt und verfügen als Team über alle nötigen Kompetenzen, die für die Erzeugung eines Produktinkrements erforderlich sind" (vgl. [2]).

Wie cross-funktional soll mein Team sein?

Agile Teams sollen cross-funktional sein. Punkt. Oder ist es vielleicht doch nicht ganz so einfach? Was genau bedeuted Cross-Funktionalität eigentlich? Und gilt generell die Regel: Je cross-funktionaler, desto besser? Von Arne Roock und Stefan Roock

Das Ziel nicht aus den Augen verlieren

Aber wann genau ist ein Team eigentlich cross-funktional? Diese Frage scheint einfach, lässt sich so pauschal aber gar nicht beantworten, denn entscheidend ist, welches Ziel das Team erreichen soll. In Scrum besteht das Minimalziel des Teams darin, in jedem Sprint ein Produktinkrement fertigzustellen. Und was genau es bedeutet, ein Produktinkrement fertig zu haben, wird in der Definition of Done in jedem Scrum-Team neu festgelegt. Es kann sein, dass es reicht, Funktionalität entwickelt zu haben und diese in einer Testumgebung zu präsentieren. Dann bräuchte man im Scrum-Team vermutlich Entwickler und evtl. Tester. Wenn die Definition of Done jedoch weitreichender ist und vorgibt, dass die neue Funktionalität bereits auf einen Live-Server deployed sein muss, dann werden zusätzlich Admin-Skills benötigt. Vielleicht reicht jedoch auch dies nicht aus, und das Teams braucht zusätzlich Textübersetzungen und es müssen rechtliche Fragen geklärt werden. Dann kann es sein, dass ein Lektor und ein Anwalt mit ins Team gehören. Neben dem erwarteten Ergebnis spielt auch die Input-Seite (in Scrum häufig durch eine Definition of Ready ausgedrückt) eine wichtige Rolle: Bekommt das Team in der Sprintplanung recht genau spezifizierte Anforderungen, die es umsetzen soll – mitsamt den dazugehörigen Designs? Oder steht das Design am Spintanfang noch nicht genau fest, und auch die Anforderungen sind noch recht vage? Im zweiten Fall reicht es vermutlich nicht, wenn das Team nur aus Ent- ▶

Der etwas andere Werkzeugkoffer

Nützliche Helferlein für agile Teams

Von Norbert Hölsken

Kennen Sie diese Küchenmaschinen, mit denen man tausend tolle Sachen machen kann? Mit zig Zusatzteilen für alles Mögliche, bei denen man sich oft fragt, wofür das Teil, was man da gerade in der Hand hat, eigentlich gedacht war? Beim Kaufen konnten es nicht genug Funktionen sein – mehr Features fürs Geld sozusagen! Im Alltag verwendet man meistens nur 20% davon. Hat man dann noch Pech, werden diese nicht einmal so gut unterstützt, wie das bei anderen, spezialisierteren Modellen oder einem einfachen Küchenmesser der Fall gewesen wäre.
Mit dem gleichen Spagat zwischen einfachen, spezialisierten Werkzeugen und breiter ausgerichteten Alleskönnern hat man leider bei jeder Werkzeugauswahl zu kämpfen. Ob Sie nun schon eine (agile) Küchenmaschine haben oder nicht, die folgenden fünf „Tools“ könnten für Sie nützlich sein. Diese Vorstellung stellt kein abgeschlossenes Set dar, sondern ist eher als eine kleine Auswahl aus verschiedensten Tools zu verstehen.

Delegation Poker

Delegation Poker bildet eine Technik, die Jurgen Appelo in seinem Buch „Management 3.0“ vorstellt [1]. Eine Erkenntnis aus diesem Spiel lautet: Bei der Delegation von Aufgaben gibt es mehr Optionen als „entweder ich oder du“. Delegation Poker hilft, gemeinschaftlich und spiele-

Abb. 1: Die sieben Stufen der Delegation

Es gibt nichts Besseres als
vollständige Transparenz,
um menschliche Kreativität zu töten.

Florian Eisenberg unterhält sich mit Dave Snowden über das Cynefin-Framework, OpenSpace, Scrum und Kanban sowie überschwemmte Häuser

Würdest du dich bitte vorstellen?

Mein Name ist Dave Snowden, ich bin Chief Scientific Officer bei Cognitive Edge. Ich verließ IBM, um das zu tun, was ich wirklich wollte: eine Firma gründen, die auf Komplexität und Erzählungen basiert.

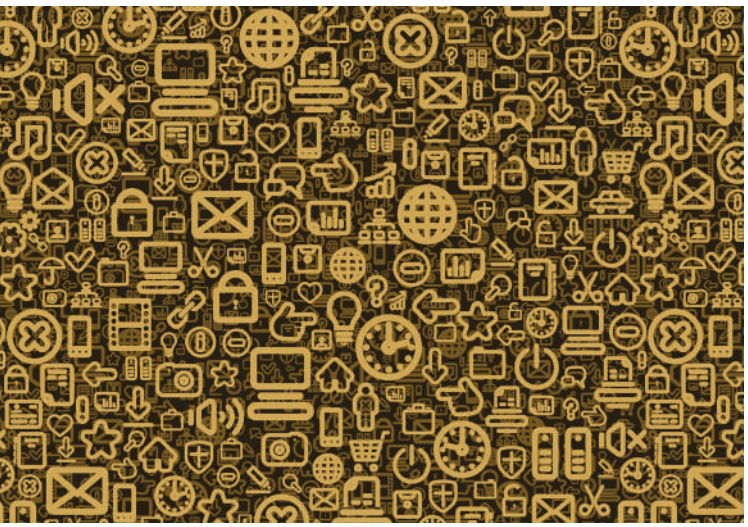
Wie kamst du zur Komplexitätstheorie?

Es ist ein ziemlich gewundener Weg gewesen. Vor IBM war ich bei Data Sciences. Dort bestand meine Rolle darin, strategische Blicke in die Zukunft zu werfen und neue Ideen zu entwickeln. Wissensmanagement hat Mitte der 90er ziemlich abgehoben. Dies habe ich dann genommen und mit dem zusammengesteckt, was ich damals gemacht habe: Operations- und Supportsysteme designen. Als IBM uns dann übernommen hat, habe ich in eine Vollzeit-Forschungsstelle bekommen – ohne operative Verantwortung, aber mit allen Freiheiten. Das hat mich zu zwei Dingen gebracht. Das eine: Wir haben entdeckt, dass Wissen hauptsächlich durch Erzählungen und Geschichten transferiert wird. Wir sind also zu den Erzählungen vom Punkt des Wissenstransfers gekommen und nicht von der Kommunikation. Die Kommunikation war zweitrangig. Als wir uns darin vertieft haben, stießen wir auf Komplexität, was die zweite wichtige Sache war. Es wurde mehr und mehr offensichtlich, dass das an vielen Stellen Relevanz hatte. Was mich hingegen nie überzeugt hat, war der Kram, der damals aus Santa Fe kam. Ich bin schon immer zutiefst misstrauisch gewesen gegenüber den Modellen mit Ameisen und Vogelnestern. Also haben wir mit einer Gruppe von hauptsächlich europäischen Akademikern diese Sache gefunden, die wir dann kognitive Komplexität genannt haben. Kognitive Komplexität vertritt die Auffassung, dass menschliche, komplexe, adaptive Systeme ein vollkommen neues Studienfeld darstellen. Wenn ich mal ganz ehrlich sein soll, ist mein gesamtes Leben ungefähr so gelaufen: Man findet interessantes Zeug, man taucht tief ein und erforscht es, dann schafft man Verbindungen zu dem, was man bereits kennt, und daraus entstehen dann weitere Dinge.

> Viel zu viele Menschen, besonders in der Agilen Bewegung, verwenden Cynefin als 2x2-Matrix und verstehen nicht, wozu es eigentlich da ist.

Du bist der Vordenker des Cynefin-Frameworks [vgl. Abb. 1]. Kannst du in 2–3 Sätzen beschreiben, worum es da geht?

Das fundamentale Prinzip hinter Cynefin besteht darin, dass unterschiedliche Typen von Systemen unterschiedliche Arten erfordern, wie Entscheidungen getroffen werden. Man muss also diese verschiedenen Systeme herausarbeiten und ordnen. Und tatsächlich können widersprechende Methoden in unterschiedlichen Domänen funktionieren. Als ich Philosophie studiert habe, hat ▶



Eine App bauen kann jeder! Oder doch nicht?

Erfahrungen mit dem Um- und Neubau der Top-5-iPhoneApp von mobile.de

Von Holger Bohlmann und Sven Günther

Welche ist die richtige Smartphone-App für meine Kunden? Welche Arten von Smartphones möchte ich unterstützen? Gibt es überhaupt Kunden da draußen, die sich für eine App oder eine Web-App interessieren? Diese Fragen stellen sich heute immer mehr Unternehmen, wobei die Antwort auf die letzte Frage heutzutage meistens mit „Ja“ zu beantworten ist. Schwieriger wird es hingegen, wenn man sich die Frage stellt: „Was ist eine gute App, die sich an den Bedürfnissen der Kunden orientiert?“

Auch mobile.de – Betreiber von Deutschlands größtem Fahrzeugmarkt – erkannte vor einigen Jahren das Potenzial mobiler Anwendungen. Allerdings war damals noch unklar, ob es wirklich genügend Interessenten für eine mobile und eine native App geben würde. Oder geben sich die Kunden vielleicht auch mit einer Web-App zufrieden, mit der man nach Fahrzeugen suchen kann? mobile.de probierte es aus und ließ sich eine Web-App entwickeln, die allerdings sehr generisch konzipiert war und alle möglichen Smartphones unterstützen sollte. Eine spezielle Anpassung der UI für den Platzhirsch iPhone war in diesem Ansatz nicht vorgesehen, wodurch es wahrscheinlich erschien, dass iPhone-Besitzer von dieser Lösung enttäuscht sein könnten. Gleiches gilt natürlich auch für den stetig wachsenden Markt von Android-Phones.

Eine erste native App

Also musste eine native iPhone-App her! Und weil die Zeit knapp war und intern nicht die nötige technische Erfahrung vorhanden war, beauftragte mobile.de einen externen Dienstleister mit der Entwicklung. Es lag ein günstiges Angebot vor und man wollte nicht allzu viel investieren, da es durchaus sein konnte, dass eine native App ein Randprodukt bleiben würde. mobile.de war sich bewusst, damit insgesamt ein gewisses Risiko einzugehen. So entstand relativ schnell eine App, die dann, wie es so schön heißt, in den App-Store ging.

Nach kurzer Zeit gab es zwei Überraschungen: Die angenehme war, dass der „Andrang“ doch erstaunlich groß war. Viele Kunden hatten sich die App installiert, um damit nach Fahrzeugen zu suchen. Es gab also offenbar

User Experience Design in Scrum

Von Doreen Timm

Zeit und Leben schreiten voran, die virtuelle Welt ist omnipräsent und hat einen viel stärkeren Einfluss als noch vor zehn Jahren. Während die Offline-Welt von harten Deadlines geprägt ist, bedingt durch mechanische und industrielle Produktion, ist die Online-Welt eher eine organische, stetig wechselnde, sich anpassende Welt – fast möchte man sagen: ein lebender Organismus [1]. Die agile Softwareentwicklung ist durch die Arbeit in Sprints flexibel und kann z. B. schnell reagieren und die Time-to-market verkürzen. Dies ist ein wichtiger Wettbewerbsvorteil.
Die agile Bewegung hat die Softwareentwicklung nachhaltig beeinflusst: bessere Software kommt schneller auf den Markt.
Außerdem hat User Experience Design (UX-Design oder einfach UX) in den letzten Jahren eine immer größere Bedeutung erlangt. Mit steigender Verbesserung der Online-Welt haben sich auch die User verändert. Ungeduldiger als früher wird sowohl eine kontinuierliche Verbesserung erwartet als auch eine zeitnahe Anpassung des Produktes – Hopp oder Top entscheidet der User mit einem Klick oder einem Tweet. UX ist heute häufig ein entscheidendes Differenzierungsmerkmal für Produkte.
Agil und UX sind unabhängig voneinander entstanden und daher erst einmal nicht integriert. Viele Unternehmen spüren allerdings, dass dieses Nebeneinander von Agilität und UX-Design nicht optimal ist und eine Integration notwendig wird. Dieser Artikel beschreibt anhand der grundsätzlichen Problemstellung, warum diese Notwendigkeit besteht und diskutiert die Vor- und Nachteile verschiedener Integrationsmöglichkeiten von Agilität und UX.

Scrum kommt nach UX?

Die einzelnen Techniken des UX-Designs haben sich im Laufe der Jahre wenig verändert. Und glücklicherweise finden sich im UX-Design einige Parallelen zu agiler Entwicklung: Auch bei UX wird z. B. mit kurzen Feedbackzyklen gearbeitet und direktes Feedback vom Endkunden eingeholt. Allerdings gibt es in der Praxis eine sequenzielle Zweiteilung in UX-Design und Softwareentwicklung. UX ist in sich ein iterativer Prozess, der Anwender einbezieht. Wie auch bei Scrum ist jeweils zu entscheiden, wie man mit dem Feedback der Anwender umgeht. Für diese Entscheidung wird häufig ein spezieller Product ▶

Was ist UX?

User Expererience (UX) beschreibt das Nutzererlebnis bei der Interaktion mit einer Software, einem Dienst oder einem Produkt. Die positive Erfahrung bei der Benutzung z. B. einer Website oder einer App wird gemessen und soll möglichst verbessert werden. Das Aussehen, die Usability und die Funktionalität gehören zu den Aspekten einer User Experience und basieren auf einer Reihe von Interaktionen des Benutzers mit dem Produkt. Ein Beispiel für eine Messung ist Eye–Tracking (Blickerfassung): die Aufzeichnung und Analyse der Blickbewegung des Users. Das positive Erlebnis oder überhaupt Emotionalität zu messen, ist komplex und wird zum Beispiel auch durch Interviews versucht einzufangen.

Türme sprengen im Team

Visuell unterstütztes Refactoring als Team-Aktivität

Von Andreas Havenstein

Der „Big Ball of Mud“ [1] ist ein Muster, das sich in vielen länger laufenden Projekten zeigt. Die Strukturen werden unleserlich, zu viel Funktionalität und Daten werden an wenigen Stellen aggregiert, es bilden sich „God Classes“. Als God Classes werden große Klassen bezeichnet, die sehr viel Funktionalität beinhalten und viele Daten verwalten. Ausgerechnet diese Stellen sind es dann, die häufigen Änderungen unterliegen. Und diese Änderungen werden teurer und teurer, da der Code immer schwerer zu verstehen ist.

Aufgabe des Teams muss es sein, diese Stellen nicht immer weiter zu verschlacken, sondern die Strukturen regelmäßig durch Refactoring wieder lesbar und änderbar zu gestalten.

Im Folgenden stellen wir verschiedene Strategien vor, mit denen Teams ein gemeinsames Verständnis über Problemstellen entwickeln und ein angemessenes Vorgehen ausprobieren können. Diese Strategien haben wir bereits in mehreren Projekten ausprobiert.

Wenn ein Team seinen Code aufräumen möchte, sollte es sich diese Fragen stellen:

- Wie können wir Problemstellen identifizieren?
- Wie gelangen wir zu einem gemeinsamen Verständnis über Qualität und Codestrukturen?
- Wie können wir gemeinsam im Team die Qualität, Lesbarkeit und Änderbarkeit des Codes erhöhen?
- Mit welchen Refactoring-Praktiken gelangen wir ans Ziel, ohne Fehler an anderer Stelle zu verursachen? Wie können wir diese Praktiken im Team verbreiten?

Code Reviews

Eine klassische Strategie, um ein gemeinsames Verständnis vom Code herzustellen, besteht darin, regelmäßig Code-Reviews mit dem Team durchzuführen. Ein beispielhafter Ablauf sieht so aus:

Das Team kommt alle zwei Wochen für eine Stunde zusammen. Ein Teammitglied blättert durch Codestellen, während die anderen über einen Beamer zusehen. Gute Codeausschnitte werden exemplarisch als Vorbild präsentiert, während Codestellen, die Schmerzen verursachen, als Ausgangsbasis für Diskussionen über Code-Verbesserungen dienen.

Leider ist dieses Format nicht besonders interaktiv, denn es ist ermüdend, sich „Code-Wüsten“ auf einem Beamer anzusehen. Die Diskussionen sind für ein gemeinsames Verständnis zwar hilfreich, die Erfahrung zeigt jedoch, dass sich in der Regel nur wenige Teammitglieder beteiligen. Häufig denkt das Team dann nötige Umbauten nur theoretisch an, aber die konkrete Vorgehensweise und die Refactoring-Praktiken werden nicht demonstriert und erprobt.

Coding Dojos

Ein interaktiveres Format stellen Coding Dojos dar. Dieses ist besonders gut geeignet, um ein gemeinsames Verständnis für Entwicklungspraktiken wie Test Driven Development zu gewinnen. Auch hier sitzt das ganze Team für ein bis zwei Stunden zusammen. Alle können auf dem Beamer-Bild sehen, was auf dem zentralen Rechner vor sich geht. Nun geht es darum, eine klar definierte ▶

Lean Startup in bestehenden Unternehmen

Von Manuel Küblböck

Agile Teams werden immer besser darin, Produkte in guter Qualität iterativ zu entwickeln. Trotzdem sind viele Projekte kein durchschlagender Erfolg. Das Problem: Teams entwickeln am Markt vorbei. Zahlreiche Unternehmen reagieren darauf mit einer langen Analyse der Anforderungen vor der eigentlichen Entwicklung – also zu einem Zeitpunkt, zu dem sie am wenigsten über den Markt und das eigene Produkt wissen. Lean Startup dagegen fordert die Integration der Anforderungsanalyse in den iterativen Entwicklungsprozess. Dies stellt Teams in bestehenden Unternehmen vor Herausforderungen, die sich deutlich von dem Umfeld unterscheiden, in dem typische Startups agieren.

Was bedeutet Lean Startup?

„Lean" stammt aus dem Lean-Thinking-Ansatz und steht dort vor allem für Vermeidung von Verschwendung (muda) und Verkürzung der Durchlaufzeit (Lead Time). Durch Lean Startup soll die Produktentwicklung effektiver werden, indem Teams schneller herausfinden, wie ein auf dem Markt erfolgreiches Produkt beschaffen sein muss.

Steve Blank definiert Startups als temporäre Organisationen auf der Suche nach einem Geschäftsmodell, das wiederhol- und skalierbaren sowie profitabel ist. Ist dieses gefunden, muss „nur" noch skaliert und das Geschäftsmodell ausgeführt werden. Damit wird die temporäre Organisation zum Unternehmen.

Lean Startup kann aber nicht nur in echten Startups angewandt werden, sondern in jeder Umgebung, in der innovative Produkte entwickelt werden sollen – also auch innerhalb bestehender Unternehmen. Innovativ heißt in diesem Kontext, dass das Geschäftsmodell in irgendeiner Form neu ist. Dies kann daran liegen, dass wir mit einem neuen Produkt einen neuen Markt erschließen (z. B. Twitter), oder auch, dass wir ein neues Produkt in einem existierenden Markt platzieren wollen (z. B. Facebook).

▶

it-agile

Entwicklung, Beratung, Schulung

*Was? it-agile macht auch **Entwicklung?***

Agile Entwicklung mit bester Flexibilität, Time-to-Market und Qualität – z. B. die iPad-App von MyHammer.

MyHammer

Agile Entwickler von it-agile

www.it-agile.de

coders@it-agile.de