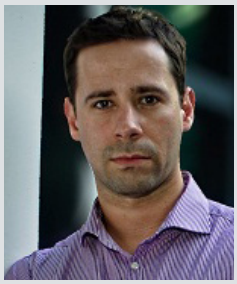

Dr. Thorsten Keuler

(KeulerThorsten@JohnDeere.com)
arbeitet bei John Deere und ist dort verantwortlich für Strategien und Maßnahmen zur nachhaltigen Sicherung der Softwarequalität in der Displayentwicklung. Zuvor hat er zehn Jahre am Fraunhofer Institut für Experimentelles Software Engineering (IESE) in Kaiserslautern gearbeitet und dort vorwiegend Industrieprojekte und Schulungen im Bereich System- und Softwarearchitekturen geleitet. Sein Interesse gilt insbesondere der Skalierbarkeit von agilen Entwicklungsmethoden im Kontext verteilter Entwicklung komplexer Systeme.

objektspektrum themenspecial: agility

„Der Worte sind genug gewechselt, …

… laßt mich auch endlich Taten sehen! Indes ihr Komplimente drechselt, kann etwas Nützliches geschehn.“ Würde dieses berühmte Zitat nicht zwangsläufig mit Goethes Faust in Verbindung gebracht, wäre es durchaus denkbar, dass es hier um einen hochgestochenen Disput zwischen einem Agilisten und einem klassischen Softwareingenieur geht. Die Diskussion um die Frage, wann es Zeit ist, theoretische Überlegungen abzuschließen und entschlossen zu handeln, ist damit belegbar älter als die Disziplin der Softwareentwicklung und damit auch des agilen Manifests.

Als das agile Manifest um die Jahrtausendwende entstand, schienen die revolutionären Ideen hochgradig inkompatibel mit den bis dahin allgemein gängigen Softwareentwicklungsmethoden. Vom jeweiligen Standpunkt aus, von dem beide Welten damals aufeinander schauten, haben sich über die Zeit im Wesentlichen zwei Strömungen entwickelt.

Die Vertreter klassischer Entwicklungsmethoden versuchen seit geraumer Zeit, Ansätze und Praktiken aus der agilen Welt in traditionelle Vorgehensweisen zu übernehmen. In diesem Lager würde mittlerweile wohl niemand mehr behaupten, dass schnelles Feedback nicht erstrebenswert ist, oder dass „over-engineering“ einen Nutzen hat. Einigkeit herrscht offenbar auch darin, dass mit agilen Praktiken Ziele wie eine höhere Geschwindigkeit und verbesserte Flexibilität erreichbar sind, getreu dem Motto: Wir bleiben bei dem, was wir kennen, aber vielleicht sind einige Ideen und Konzepte aus der agilen Welt doch nicht so schlecht.

Genauso gibt es aus dem Lager der Agilisten immer häufiger Anzeichen dafür, Softwaretechniken aus etablierten, klassischen Herangehensweisen zu adaptieren und im agilen Umfeld einzusetzen. Besonderes Augenmerk liegt dabei meist auf der Skalierung von agilen Methoden. Viele Firmen, die agil Software entwickeln, reichern ihr Vorgehen mit Ansätzen aus dem klassischen Software-Engineering an, und profitieren so von bewährten Softwaretechniken. Hier lautet das Motto: Wir haben zwar erstmal alles neu und ganz anders gemacht, haben dann aber gemerkt, dass das ein oder andere aus den „altmodischen“ Entwicklungsansätzen vielleicht doch nicht so unsinnig war.

Das Streben nach Verbesserung

Mittlerweile ist allerdings davon auszugehen, dass überwiegend nicht mehr im Grundsatz über Philosophieunterschiede zwischen agiler und traditioneller Entwicklung diskutiert wird. Interessant ist daher zu sehen, wie Firmen für sich ausprobieren, was funktioniert und wie sie sich verbessern können. Dabei geht es nicht um Glaubensfragen, sondern um die einfach anmutende Zielstellung: Projekte „on time“ und „in budget“ abzuschließen.

Gerade bei der Umstellung auf agile Entwicklung übersehen viele Firmen schnell wichtige Zusammenhänge, die letztlich über Erfolg oder Misserfolg entscheiden. Eine Möglichkeit, den agilen Weg für sich auszuprobieren, liegt nämlich darin, es einfach mit einem kleinen Team auszuprobieren. Dazu führt in der Regel ein Coach das Team in die Welt der agilen Prinzipien und Praktiken ein, um dann ein überschaubares Projekt innerhalb weniger Monate zum Abschluss zu bringen. Ist das Pilotprojekt erfolgreich,

kommt nach dem unternehmensweiten „Rollout“ dann schnell das böse Erwachen. Niemand hat offenbar damit gerechnet, dass die Koordination und Abstimmung von vielen agilen Teams einiges mehr an Organisations- und Kommunikationsfähigkeiten verlangt.

In diesem Kontext beschreibt **Lutz Malburg** in seinem Artikel „Agil – Die mentale Revolution“, dass bei der Einführung agiler Methoden der Schlüssel zum Erfolg in einer stark veränderten Denkweise und einer anderen Kultur liegt.

Dass agile Softwareentwicklung aber nicht nur in einem kleinen Team funktioniert, sollte sich mittlerweile herumgesprochen haben. Allerdings sehen sich Firmen in der Praxis trotzdem immer wieder mit Herausforderungen konfrontiert, wenn es um die Umsetzung guter Teamarbeit geht. Einen konkreten Ansatz, wie verteilte agile Teams effizient zusammenarbeiten können, zeigen **Baris Güldali** und **Masud Fazal-Baqaie** in ihrem Artikel „Skalieren von großen agilen Projekten mit verteilten Backlogs“.

Sind die Anforderungen geklärt und abgestimmt, gilt es diese mithilfe einer passenden Produktvision, einer System-Metapher oder einer High-level Architektur in Form eines „Big Pictures“ in die Teams zu tragen. Im Artikel „Vom ‚Big Picture‘ zur Umsetzung“ demonstrieren **Philip Stolz**, **Bertil Muth** und **Markus Eberhardt** an einem Beispiel, wie der Blick auf das „große Ganze“, trotz vieler kleiner User Stories und häufiger Änderungen, nicht verloren geht – ohne am Anfang der Entwicklung eine umfangreiche Spezifikation zu schreiben.

Konkrete Unterstützung zur Umsetzung des Big Pictures geben **Dominik Rost**, **Balthasar Weitzel**, **Matthias Naab** und **Torsten Lenhart** in ihrem Artikel „Durch kleine Bausteine direkter zum Ziel – Architektur-Best-Practices für Agile Entwicklung“. Die Autoren stellen den Bezug zwischen agiler Entwicklung und Architekturarbeit her und präsentieren ihre Kernidee, Best Practices aus klassischen Architekturmethoden zu extrahieren und leichtgewichtig für agile Projekte aufzubereiten.

Spätestens wenn sich die ersten Features in der Umsetzung befinden, stellt sich die Frage nach der Qualitätssicherung. Das Thema Qualität erfährt in der agilen Entwicklung nach wie vor eine eher stiefmütterliche Behandlung. Bei komplexen Softwaresystemen ist die geforderte Systemqualität kein Zufallsprodukt, und auch die qualitativ besten Softwareentwürfe entstehen nicht einfach von selbst. Die Grundvoraussetzung zur Erlangung von Softwarequalität ist zu wissen, was mit der Qualität gemeint ist. Erst wenn diese Frage geklärt ist, kann man zielführende Strategien zur Qualitätssicherung entwickeln. Dazu beschäftigt sich **Dominique Mühlbauer** in seinem Artikel „Das Q in Agile - Qualitätssicherung in agilen Projekten“ mit der Frage, was zu einem ganzheitlichen Qualitätssicherungsansatz gehört und wie dieser in einen agilen Rahmen eingepasst werden kann.

Hieraus ergibt sich dann schnell die Anforderung, die traditionelle Rolle eines Software-Testers in die agile Qualitätssicherung zu überführen. Wie das konkret aussehen kann, damit beschäftigt sich der Artikel „Schnelles Feedback für agile Teams! Mit der neuen ISTQB®-Ausbildung zum Certified Agile Tester“. Die neue ISTQB®-Ausbildung zum Certified Agile Tester erleichtert Softwaretestern, die aus traditionellen Projekten in agile Entwicklungsprojekte wechseln, den Umstieg, und sie vermittelt in kompakter Form das methodische Rüstzeug für erfolgreiches, agiles Testen.

Selbst wenn ein System bereits implementiert und im Einsatz ist, so stellen sich in nachgelagerten Phasen des Produktlebenszyklus durchaus interessante Fragen, die sich oft nur schwer mit einem Blick in den Code beantworten lassen. Wie dann das Ziel mit Hilfe von klassischen Methoden erreicht werden kann, beschreibt **Andrea Herrmann** in ihrem Artikel „Agiles Requirements-Engineering statt Konzeption? Wie Sie Ihre agil entwickelte Software mit RE refaktorieren“. Der Artikel illustriert anhand einer Fallstudie, wie mit Hilfe von Requirements-Engineering-Techniken die Qualität einer existierenden Patientensoftware verbessert wird.

Wo kommen wir her? Wo gehen wir hin?

Auch wenn so mancher vor rund 15 Jahren noch in den Schubladen „wir“ und „die anderen“ gedacht hat, würde ich behaupten, dass die Übergänge von traditioneller zu agiler Softwareentwicklung mittlerweile fließend sind. Also, wo kommen wir her? Die Antwort ist einfach: Es spielt eigentlich keine Rolle. Wir mögen vielleicht das Produkt unserer eigenen Vergangenheit sein, aber viel wichtiger ist, dass wir alle letzten Endes dasselbe Ziel verfolgen: Bessere Software in kürzerer Zeit zu erstellen. Es ist sehr wahrscheinlich, dass wir das Ideal nie erreichen werden. Aber die Tatsache, dass wir aus unterschiedlichen Richtungen kommen, lässt mich hoffen, dass wir das Ziel zumindest umzingeln können. ■

Und jetzt wünsche ich Ihnen viel Spaß beim Lesen dieses Online Themenspecials!

Ihr

Dr. Thorsten Keuler