mehr zum thema:
http://blog.codecentric.de/
http://www.meettheexperts.de/

AGILITÄT GESTERN, HEUTE UND MORGEN: EINE BESTANDSAUFNAHME UND EIN BLICK IN DIE ZUKUNFT

In diesem Artikel zum Schwerpunktthema „10 Jahre Agiles Manifest" lasse ich die vergangenen Jahre Revue passieren und mache mir ein paar Gedanken darüber, wie es weitergehen könnte. Ich möchte erzählen, wie die Agilität Einzug in die IT gehalten hat, wie es heute um die Agilität steht und in welche Richtung sich das Thema weiterentwickeln könnte. Der Artikel deckt sicherlich nicht alle Aspekte vollständig ab und untersucht nicht streng wissenschaftlich alles bis ins letzte Detail, aber ganz im Sinne der Agilität erhebe ich auch gar nicht diesen Anspruch.

der autor

Uwe Friedrichsen
(E-Mail: uwe.friedrichsen@codecentric.de)
hat langjährige Erfahrungen als Architekt, Projektleiter und Berater. Aktuell ist er bei der codecentric AG für die strategische Entwicklung neuer Paradigmen und Technologien verantwortlich und beschäftigt sich in dem Kontext insbesondere mit agilen Verfahren und neuen Architekturansätzen.

Als ich Anfang der 90er Jahre meine IT-Begeisterung nach erfolgreich abgeschlossenem Studium zum Beruf machte, waren *Structured Analysis (SA)* und *Structured Design (SD)* noch die vorherrschenden Konzeptionsmethoden. Die seit Anfang der 70er Jahre immer weiter ausgefeilten Prinzipien der Softwaretechnik bzw. des Software-Engineerings sorgten für einen ordentlich geregelten Rahmen.

Die Welt vor der Agilität

Die IT-Welt war damals scheinbar in Ordnung. Allerdings säte die immer noch nicht überwundene Softwarekrise ihre Zweifel und der Siegeszug der PCs in den Büros dieser Welt seit Ende der 80er Jahre sorgte für Unruhe. Insbesondere die damit häufig verbundenen Ad-hoc-Entwicklungen, die vielfach so erfolgreich waren, dass sie den klassischen Methoden den Schneid abzukaufen drohten, sorgten für einige Unruhe. Es brodelte, aber noch konnte man den Deckel drauf halten.

Dazu kam, dass der Durchbruch der Objektorientierung zu Beginn der 90er Jahre nach neuen Konzeptionsmethoden verlangte. Dieses Bedürfnis wurde übererfüllt, indem die Methoden gleich im Dutzend auf den Markt geworfen wurden: Peter Coad und Ed Yourdon, Sally Shlaer und Stephen Mellor, Rebecca Wirfs-Brock, Jim Odell, Grady Booch, James Rumbaugh, Ivar Jacobson, Donald Firesmith, Brain Henderson-Sellers und Ian Graham – um nur die wichtigsten Protagonisten zu nennen. Sie alle verbreiteten ihre Vorstellungen von der „einzig richtigen" Methode und Notation zur Konzeption und Erstellung objektorientierter Softwaresysteme. Diese Zeit wurde unter dem etwas martialischen Begriff „Methodenkrieg" bekannt. Einen Überblick gibt Abbildung 1.

Das Ende dieses „Krieges" wurde Mitte der 90er Jahre mit der von Grady Booch und Jim Rumbaugh entwickelten *Unified Method 0.8* eingeläutet. Kurz danach kam dann noch Ivar Jacobson mit dazu und weil sich die „drei Amigos" offenbar doch nicht so recht auf eine einheitliche Methode einigen konnten, wurde aus der „Unified Method" kurzerhand die *Unified Modeling Language (UML)*, die uns bis heute begleitet. Nach ein paar Jahren konnten sich die drei dann unter Führung von Philippe Kruchten doch noch auf den *Rational Unified Process (RUP)* einigen.

Ähnlich wie das parallel in den 90er Jahren entstandene und populär gewordene *V-Modell* zeichnet sich der RUP durch einen sehr hohen Detaillierungs- und Formalisierungsgrad aus. Man konnte damit einfach alles regeln. Unterstützt wurden diese Methoden von in der gleichen Zeit populär gewordenen Qualitätsstandards wie *DIN EN ISO 9000* und das *Capability Maturity Model (CMM)*.

Auch wenn viele Befürworter dieser Methoden und Standards später immer wieder betonten, dass man sie alle doch auch sehr leichtgewichtig verwenden könnte, wurde das in der Regel nicht getan. Es wurde geregelt, was das Zeug hielt. Zufall

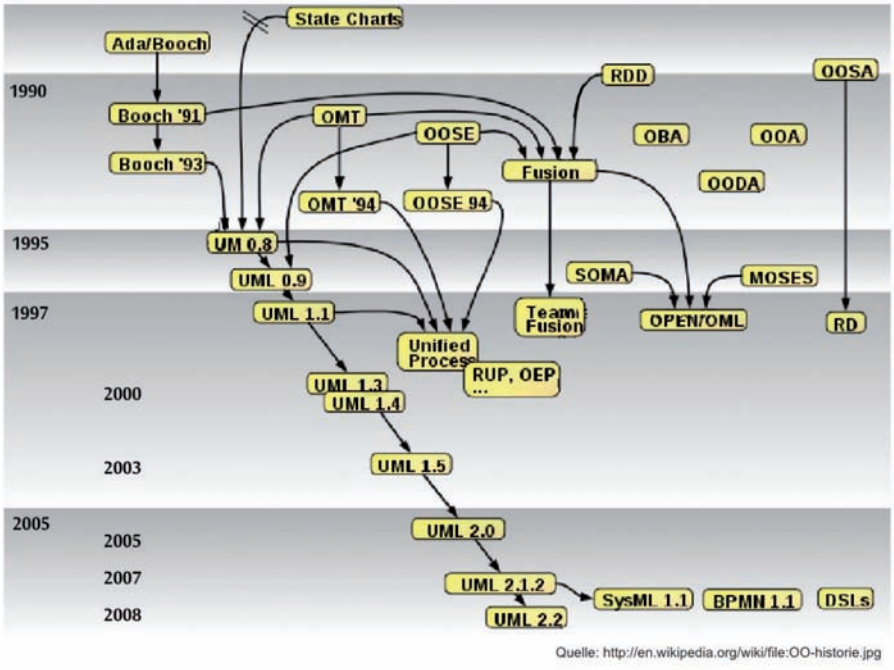

Abb. 1: Überblick über objektorientierte Notationen und Methoden.

gab es nicht mehr, alles war im Prozess geregelt. Treiber für das Vorgehen waren nicht die lästigen Kundenanforderungen, sondern die Architektur. Das Schlagwort von der architekturgetriebenen Softwareentwicklung war in aller Munde und die IT-Welt war wieder in Ordnung. Dass die Softwarekrise noch immer nicht gelöst war, wurde geflissentlich ignoriert, ebenso wie die Tatsache, dass die ganzen Prozesse und Methoden offenbar in keinster Weise den Projekterfolg sicherstellten, allerhöchstens eine Vervielfachung der Kosten.

Ich hatte zu der Zeit häufig ein schlechtes Gewissen, weil ich seit jeher eher einen Hang zum Pragmatismus und nicht so sehr zum Formalismus hatte und Dinge viel lieber so machte, wie es erfolgversprechend war, statt mich methodenkonform an die zahlreichen Vorgaben und Templates zu halten. Das durfte man zu der Zeit zwar nicht laut sagen, aber solange man gute Ergebnisse lieferte, wurde ein solches Verhalten zumindest geduldet.

Auftritt der Agilisten

In diesen trügerischen Frieden schlug die Agilität wie eine Bombe ein. Ich kann mich noch immer sehr gut an die Keynote von Kent Beck auf der OOP-Konferenz im Jahr 1999 erinnern, in der er die bis dahin etablierten Methoden unerbittlich für gescheitert erklärte und mit dem *eXtreme Programming (XP)* einen radikalen Gegenentwurf zu den planungs- und architekturzentrierten Methoden der 90er Jahre aufstellte: Weg mit den ewigen Planungen und ausufernden Konzeptionsphasen und stattdessen den Kunden und seine Wünsche, Geschäftswert und lauffähige Software in das Zentrum des Interesses stellen. Diese Forderung klang damals einfach unerhört. Und dann auch noch so exotische Ideen wie *Pair Programming*, den Kunden als Teil des Teams begreifen, testgetriebene Softwareentwicklung und lauffähige Software ab der ersten Iteration. Was war nur mit dem Mann los?

Ich weiß nicht mehr, was ich sonst noch auf der Konferenz gehört habe, denn die Diskussionen, die ab der Keynote die Konferenz beherrschten, haben alles andere komplett überdeckt. So umstritten die Ideen von Kent Beck in ihrer damaligen Radikalität auch waren: Jeder spürte, dass sich etwas Grundlegendes geändert hatte. Noch wusste keiner, wohin es uns treiben würde, aber allen war klar, dass die alten Ideale der 90er ausgedient hatten, egal ob man sie nun gut fand oder nicht.

Zu Kent Beck gesellten sich immer mehr weitere Protagonisten, zum Beispiel Ken Schwaber und Jeff Sutherland, Alistair Cockburn, Martin Fowler, Robert C. Martin (besser bekannt als „Uncle Bob") und Ward Cunningham, um nur ein paar zu nennen. Natürlich waren die meisten von ihnen auch schon vorher aktiv und bekannt, aber erst Kent Beck und sein XP sorgten – zumindest in Deutschland – für die agile Initialzündung. Etwa zwei Jahre nach Kent Becks Keynote auf der OOP war es dann soweit: Die genannten Personen sowie eine Reihe weiterer agiler Protagonisten kamen im Februar 2001 für drei Tage in einem Ski-Resort in Snowbird (Utah) zusammen und formulierten das „Agile Manifest" (vgl. [Agi]), das in diesem Jahr seinen zehnten Geburtstag feiert.

Interessanterweise hat das Manifest zumindest in meinem Umfeld lange nicht die explosive Wirkung gezeigt wie Kent Becks XP-Feldzug. Das Manifest schlich sich eher durch die Hintertür in meine Projekte ein. Man kannte es, man schätzte es, keiner widersprach dem darin Geschriebenen offen, aber es brauchte doch relativ lange, bis aus diversen Lippenbekenntnissen ein Umdenken wurde. Es mag auch daran liegen, dass ich zu der Zeit primär mit Unternehmen zusammen arbeitete, die zumindest damals eher auf der „rechten Seite" des Manifests anzusiedeln waren, d. h. die eher einem traditionellen Wertesystem verhaftet waren. Man konnte in dem Umfeld zwar ein paar ausgewählte agile Praktiken verankern, aber ein Etablieren der agilen Ideen und Werte auf breiter Ebene war (noch) nicht möglich.

In unseren Projekten etablierten wir sukzessive immer mehr agile Praktiken, da wir wie alle anderen auch unter den gleichen Problemen litten, die auch die agilen Protagonisten zum Entwickeln ihrer leichtgewichtigen Methoden und zum Agilen Manifest geführt hatten. Wir entfernten so viel Overhead wie möglich (was aufgrund der umgebenden Management- und Kundenvorgaben nicht immer einfach war). Wir förderten direkte Kommunikation und arbeiteten mit häufigen Feedback-Zyklen, um sicherzustellen, dass wir noch auf dem richtigen Weg sind. Wir versuchten, die Anforderungen des Kunden in Abstimmung mit ihm zu priorisieren. Auch Ideen wie das tägliche Standup-Meeting aus Scrum und diverse XP-Techniken fanden ihren Einzug in unsere Projekte.

Aber es blieb immer ein Kompromiss. Das enge Prozesskorsett der beteiligten Unternehmen machte es praktisch unmöglich, durchgängig agil zu arbeiten. Rückblickend betrachtet würde ich sagen, wir waren eher „pragmatisch agil" als wirklich agil. Dennoch, wir taten, was wir konnten und was möglich war, und waren damit ziemlich erfolgreich.

Nachhaltig geändert hat sich die Situation für mich vor circa zweieinhalb Jahren. Mein damals neuer Arbeitgeber setzte nicht nur in der Projektdurchführung, sondern in allen Bereichen auf Agilität. Das agile Wertesystem wurde so auch durchgängig in der Interaktion mit den Kunden angewandt: Die Verträge waren kurz, die Wichtigkeit der Kundenzufriedenheit nicht nur eine Management-Worthülse und eine vertrauensvolle Zusammenarbeit mit dem Kunden war wichtiger als die kurzfristige Maximierung von Quartalszahlen. Gerade das Vertrauensthema war für mich eine sehr interessante Erfahrung. Ich kannte aus meiner Vergangenheit „agile" Projekte, eingebettet in perfekt ausformulierte Verträge, die von Rechtsabteilungen nach allen Regeln der Kunst auf Wasserdichte bezüglich eines möglichen Prozesses mit dem Kunden getrimmt worden waren. Wie soll ein Kunde einem glauben, dass man vertrauensvoll und partnerschaftlich mit ihm zusammenarbeiten will, wenn er so einen juristisch bis ins letzte Detail ausgefeilten Vertrag in die Hand gedrückt bekommt?

Meine Kernerkenntnis aus der neuen Vorgehensweise war, dass man zuerst auch als Unternehmen an die agilen Werte glauben muss, wenn man agile Projekte erfolgreich durchführen will. Im ersten Moment mag einen zwar der gefühlte Kontrollverlust abschrecken, aber mal ganz ehrlich: Was hat man letztlich von der zusätzlichen Absicherung, wenn man ein Projekt erfolgreich gegen die Wand gefahren hat? Wenn man sich erst einmal mit einem Kunden vor Gericht streitet, ist der Imageschaden perfekt, egal ob man gemäß Vertrag „recht" hat oder nicht. Da konzentriert man sich doch lieber darauf, dass der Kunde zufrieden ist und (schnell) das bekommt, was er braucht, und reduziert die ganzen Absicherungen auf ein Minimum.

Neben dem eingesparten Aufwand für letztlich unnütze Dinge hat das durchgän-

gig agile Vorgehen für mich noch einen ganz anderen Nutzen: Es ist unfassbar, wie viel mehr Spaß es macht, mit einem Kunden zusammenzuarbeiten, der einem vertraut und der davon überzeugt ist, dass man gemeinsam daran arbeitet, dessen Ziel zu erreichen. Dieses Vertrauen zwischen Kunden und Dienstleister ist aber nur möglich, wenn man auch als Dienstleister die agilen Werte ganzheitlich adaptiert und nicht ein „bisschen agil" versucht, aber ansonsten weiterhin dem nicht agilen *CYA (Cover Your Ass)* frönt.

Apropos Kunden: Sind diese denn nach zehn Jahren Agilem Manifest so richtig agil? Durch meine Dienstleister-Brille betrachtet würde ich sagen: Nicht immer, aber es wird besser und besser. Mein Eindruck ist, dass die agilen Erfolgsmeldungen die Kunden immer aufgeschlossener für Agilität machen. Trotzdem ist ein agiles Zusammenarbeiten im Kunden-Dienstleister-Verhältnis noch lange nicht selbstverständlich. Ja, man möchte Agilität, da geht doch alles viel schneller und billiger und flexibler ist es dazu auch noch. Aber trotzdem möchte man sich häufig noch nach guter Väter Sitte (und aktueller Prozessvorgabe) in alle Richtungen absichern und jedes Detail geregelt wissen. Der Wandel fällt nicht immer leicht.

Das führt dann auch immer mal wieder zu Rosinenpicken und so skurrilen Ideen wie: „Wir möchten jederzeit alles ändern können und zusätzlich am Ende alles so haben, wie es im Angebot spezifiziert wurde" oder „Agil, wie wir vorgehen wollen, spezifizieren wir die Anforderungen erst während des Projekts, wollen aber trotzdem einen Werkvertrag mit Festpreis, weil das unser Einkauf so will". Da heißt es dann auf die agile Tugend der direkten Kommunikation mit dem Kunden zurückzugreifen. In Summe lässt sich aber festhalten, dass die meisten Kunden auch in der Zusammenarbeit mit Dienstleistern immer stärker auf agile Prinzipien setzen und auch immer stärker die agilen Werte dahinter verstehen und verinnerlichen.

Agilität heute …

Wo stehen wir heute, zehn Jahre nach dem Agilen Manifest? Ich denke, man kann sagen, dass Agilität mittlerweile Mainstream ist. Scrum ist bis in die obersten Führungsebenen allgegenwärtig, XP ebenso in den entwicklungsnäheren Kreisen. Kanban wiederum schickt sich an, sich an den Stellen zum agilen Standard zu mausern, an denen die Veränderung (*Change*), die Scrum einfordert, zu radikal ist oder an denen Scrum aus anderen Gründen nicht die probate Methode ist.

Auch wenn mittlerweile jeder agil sein will, sei es, weil man sich davon einen Wettbewerbsvorteil verspricht oder weil man gehört hat, dass Scrum alle Probleme lösen würde (was natürlich nicht stimmt), so ist doch die große Lektion der vergangenen Jahre, dass Agilität eine Menge Veränderung bedeutet. Eine Methode wie Scrum oder Kanban ist schnell gelernt, nicht umsonst passt Scrum auf einen Bierdeckel. Aber agil zu denken, das agile Wertesystem zu verinnerlichen und danach zu handeln, bedeutet für alle Beteiligte eine große Veränderung, oftmals auch persönlich. Und das fällt nicht jedem leicht.

Entsprechend hört man heute auf agilen Konferenzen immer weniger über Methoden und immer mehr über *Change Management* und die menschliche Psyche im Allgemeinen. Mittlerweile beschleicht mich gelegentlich der Eindruck, dass sich die agile Community vor lauter „Veränderung" und „Motivation" teilweise in ziemlich esoterischen Gefilden verläuft, aber letztlich sind genau diese Themen die größte Herausforderung für eine richtige Einführung von Agilität. Die Methoden funktionieren fast von allein, wenn man erst einmal den geistigen Schritt hin zu den agilen Werten getan hat.

Ein weiteres großes Thema ist die Einbindung agiler Methoden wie Scrum oder Kanban in die Unternehmenssteuerung. Durch die neuen Methoden haben viele Unternehmen das Gefühl, die von ihnen benötigten Controlling- und Steuerungsmöglichkeiten gingen ihnen verloren. Sie erhalten nicht mehr die notwendigen Kennzahlen – zumindest nicht, wenn sie die Methode isoliert betrachten. Deshalb ist es wichtig, agile Methoden sinnvoll in ein Unternehmens-Controlling und eine Unternehmens-Governance einzubinden. Dabei müssen die (grundsätzlich sinnvollen) Ziele des Controllings und der Governance sichergestellt werden, die agilen Werte dürfen aber trotzdem nicht aus den Augen verloren werden. Daher denke ich, dass der weitere Erfolg der Agilität auch stark von guten Konzepten zur Einbindung in Frameworks, wie z. B. „CobiT" (vgl. [CobiT]), oder ein SOX-konformes Controlling (vgl. [SOX]) abhängen wird.

… und morgen

Zum Abschluss möchte ich noch einen Blick in die „Kristallkugel" wagen: Wie wird es mit der Agilität weitergehen? So ganz genau lässt sich das natürlich nicht voraussehen, aber ich bin überzeugt, dass Agilität nicht wie eine vorübergehende Modewelle verschwinden wird. Natürlich gab und gibt es jede Menge Hype um den Begriff und die Dienstleister und Produkthersteller verwen-

Abb. 2: Merkmale einfacher, komplizierter, komplexer und chaotischer Systeme.

den den Begriff „agil" geradezu inflationär. Natürlich gab und gibt es jede Menge Evangelisten bzw. mittlerweile eher Fundamentalisten, die dogmatisch auf die penible Einhaltung des geschriebenen Wortes pochen, egal ob es in der jeweiligen Situation sinnvoll ist oder nicht. Und natürlich sind Scrum, XP und Kanban nicht die Lösung für alle aktuellen Probleme in der IT. Tritt man aber einmal einen Schritt zurück, wird meiner Ansicht nach recht schnell klar, dass wir ohne Agilität dauerhaft gar nicht mehr überlebensfähig sind.

Warum so dramatisch? Wagt man einen Seitenblick in die Systemtheorie, dann werden dort häufig einfache, komplizierte, komplexe und chaotische Systeme unterschieden (siehe auch Abbildung 2 und vgl. [Fri10]). Spannend ist hierbei die Unterscheidung zwischen „kompliziert" und „komplex". Die Schwierigkeit komplexer Systeme steckt in der Dynamik der Beziehungen. Während es bei komplizierten Systemen hinreichend ist, die Einzelteile und ihre Beziehungen untereinander einmal zu verstehen, reicht das bei komplexen Systemen nicht. Hier muss man immer wieder das Gesamtsystem (nicht die Einzelteile) überprüfen und regelmäßige Anpassungen am Vorgehen vornehmen, um sich auf die sich über die Zeit verändernden Vorgaben einzustellen.

Genau dafür benötigen wir Agilität. Traditionelle Vorgehensweisen, die von den Ideen der klassischen Softwaretechnik aus den frühen 70er Jahren geprägt sind, gehen von einem komplizierten Umfeld aus, das man mit einer „Teile-und-Herrsche"-Strategie beherrschen kann. Heutige Softwareentwicklungsprojekte sind aber meistens hochgradig komplex: Die Zielvorgaben sind zu Beginn eines Projekts eher unscharf, Meinungen und Anforderungen ändern sich dynamisch über den Projektverlauf, Projektergebnisse führen zu neuen, veränderten Anforderungen und der Wettbewerb liefert laufend neue Treiber für Veränderungen.

Für solche Umfelder liefern die traditionellen Verfahren keine Antworten, denn sie können nur mit einfachen und komplizierten Sachverhalten umgehen (die es in heutigen Projekten natürlich ebenfalls noch gibt). Doch auf komplexe Umgebungen muss man anders reagieren. In der Managementlehre ist das schon lange bekannt (wenn auch in der Praxis leider häufig nicht umgesetzt) und es gibt auch

Abb. 3: Empfehlungen zum Management einfacher, komplizierter, komplexer und chaotischer Systeme.

sehr viel gute Literatur dazu (vgl. beispielsweise [Mal08], [Hon08], [Sno07]). Dort gibt es auch klare, wissenschaftlich belegte Empfehlungen für den Umgang mit unterschiedlichen Situationen (vereinfacht sind diese in Abbildung 3 zusammengefasst). Schaut man sich die Empfehlungen zum Umgang mit komplexen Situationen an, dann findet man dort:

- systemische Analyse
- evolutionäre Techniken
- indirekte Steuerung
- kontinuierliches Lernen
- erhöhte Interaktion

Vergleicht man diese Empfehlungen mit den Prinzipien agiler Methoden, dann stellt man fest, dass Agilität diese Empfehlungen bis auf die systemische Analyse erfüllt. Anders ausgedrückt, ist Agilität eine Möglichkeit, komplexe Systeme erfolgreich zu beherrschen. Und da die meisten heutigen IT-Projekte komplexe Systeme sind, bin ich davon überzeugt, dass Agilität bleiben wird – einfach weil wir sie benötigen, um auch zukünftig in der immer komplexer und schneller werdenden Geschäfts- und IT-Welt überhaupt überlebensfähig zu sein.

Zum erfolgreichen Umgang mit Komplexität fehlt – ergänzend zur Agilität – noch das systemische, das ganzheitliche Denken, aber auch auf dem Gebiet kann man mittlerweile eine Bewegung feststellen. Von der Seite sind meines Erachtens weitere spannende Impulse für die Weiterentwicklung der Agilität zu erwarten. Zusammenfassend denke ich, dass wir zehn Jahre nach dem Agilen Manifest schon viel erreicht haben, um die heutigen komplexen Projekte besser und erfolgreicher beherrschen zu können. Andererseits sind wir aber noch lange nicht am Ziel der agilen Reise angekommen und haben noch eine Menge Arbeit und Lernen vor uns. Ich freue mich deshalb auf die nächsten zehn Jahre Agilität. ■

Literatur & Links

[Agi] Manifesto for Agile Software Development, siehe http://www.agilemanifesto.org/

[CobiT] CobiT (Control Objectives for Information and Related Technology), siehe http://www.isaca.org/cobit

[Fri10] U. Friedrichsen, Agil oder ingenieursmäßig?, in: Business Technology Magazin 2/2010

[Hon08] J. Honegger, Vernetztes Denken und Handeln in der Praxis, Versus Verlag AG 2008

[Mal08] F. Malik, Strategie des Managements komplexer Systeme (10. Aufl.), Haupt Verlag 2008

[Sno07] D.J. Snowden, M.E. Boone, A Leaders's Framework for Decision making, in: Harvard Business Review November 2007

[SOX] Sarbanes-Oxley Act, siehe http://de.wikipedia.org/wiki/Sarbanes-Oxley_Act