Agility kompakt

Werke der „kompakt-Reihe“ zu wichtigen Konzepten und Technologien der IT-Branche:

- ermöglichen einen raschen Einstieg,
- bieten einen fundierten Überblick,
- sind praxisorientiert, aktuell und immer ihren Preis wert.

Bisher erschienen:

- Heide Balzert
  UML kompakt, 2. Auflage
- Andreas Böhm / Elisabeth Felt
  e-commerce kompakt
- Christian Bunse / Antje von Knethen
  Vorgehensmodelle kompakt, 2. Auflage
- Holger Dörnemann / René Meyer
  Anforderungsmanagement kompakt
- Christof Ebert
  Outsourcing kompakt
- Christof Ebert
  Risikomanagement kompakt
- Karl Eilebrecht / Gernot Starke
  Patterns kompakt, 2. Auflage
- Andreas Essigkrug / Thomas Mey
  Rational Unified Process kompakt, 2. Auflage
- Peter Hruschka / Chris Rupp / Gernot Starke
  Agility kompakt, 2. Auflage
- Arne Koschel / Stefan Fischer / Gerhard Wagner
  J2EE/Java EE kompakt, 2. Auflage
- Michael Kuschke / Ludger Wölfel
  Web Services kompakt
- Torsten Langner
  C# kompakt
- Pascal Mangold
  IT-Projektmanagement kompakt, 3. Auflage
- Michael Richter / Markus Flückiger
  Usability Engineering kompakt
- Thilo Rottach / Sascha Groß
  XML kompakt: die wichtigsten Standards
- SOPHIST GROUP / Chris Rupp
  Systemanalyse kompakt, 2. Auflage
- Ernst Tiemeyer
  IT-Controlling kompakt
- Ernst Tiemeyer
  IT-Servicemanagement kompakt
- Ralf Westphal
  .NET kompakt
- Ralf Westphal / Christian Weyer
  .NET 3.0 kompakt

Peter Hruschka / Chris Rupp / Gernot Starke

Agility kompakt

Tipps für erfolgreiche Systementwicklung

2. Auflage

Unter agiler Mitwirkung von Jutta Eckstein, Christiane Gernert, Thorsten Janning, Markus Reinhold und Beatrice Sablov

Spektrum
AKADEMISCHER VERLAG

Autoren:
Peter Hruschka
E-Mail: hruschka@b-agile.de

Chris Rupp
E-Mail: Chris-Rupp@sophist.de

Dr. Gernot Starke
E-Mail: gs@gernotstarke.de

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

Springer ist ein Unternehmen von Springer Science+Business Media
springer.de

2. Auflage 2009

Spektrum Akademischer Verlag ist ein Imprint von Springer

09 10 11 12 13 5 4 3 2 1

Planung und Lektorat: Dr. Andreas Rüdinger, Barbara Lühker
Umschlaggestaltung: SpieszDesign, Neu-Ulm
Herstellung und Satz: Crest Premedia Solutions (P) Ltd., Pune, Maharashtra, India

ISBN 978-3-8274-2092-3

Inhalt

Nicht die schnellsten
oder stärksten überleben,
sondern diejenigen,
die sich am schnellsten auf veränderte
Lebensbedingungen einstellen können.
Charles Darwin

(Statt einer) Einführung

Flink und behände…

Kennen Sie die Geschichte der Ameisenvölker? Die ersten Ameisenvölker lebten vor Millionen Jahren im feuchtwarmen Klima längst vergangener Zeiten. Sie krabbelten unter den Füßen urzeitlicher Tiere, an den giftigen Stängeln riesiger Farne und an den Ufern reißender Ströme entlang, ständig auf der Suche nach Futter. Damals schmeckten den Ameisen die Überbleibsel von Mammutknochen besonders gut.

Dann wurde es kalt. Eisig kalt! Viele der großen Tiere gingen an der zermürbenden Kälte der Eiszeit zugrunde. Die Ameisen nicht – flink und behände hatten sie sich andere Lieblingsspeisen gesucht, und auch neue Wege. Jahrtausende später taute das Eis – und die Ameisen lernten erneut, sich an der Hitze zu erfreuen. Wieder und wieder wandelte sich die Umwelt – und die Ameisen wandelten sich mit. Zwar änderten sie ihre Form und ihr Aussehen kaum, ihr Verhalten und ihre Vorlieben passten Sie jedoch kontinuierlich den Gegebenheiten ihrer Umwelt an.

Heute gibt es weder starke Mammuts noch riesige Giftfarne. Aber Ameisen in rauen Mengen. Und wenn wir sie fragen könnten, würden sie bestimmt sagen, dass sie sich wohl fühlten und Spaß hätten bei ihrer ewigen Krabbelei.

… oder erstarrt in Tradition

Vor langen Jahren begann der Hunger der Menschheit nach Kohle von Jahr zu Jahr zu wachsen. Das führte in vielen Gegenden Europas zur Gründung von Bergwerken – der Abbau in der Tiefe war eine gefährliche Unternehmung, die mit abenteuerlichen Methoden Mutter Erde das dunkle und wertvolle Gut entlockte. Viele Arbeiter ließen ihr Leben dabei, unter Strapazen den Grundstoff der industriellen Revolution aus dem Bauch der Erde zu stemmen.

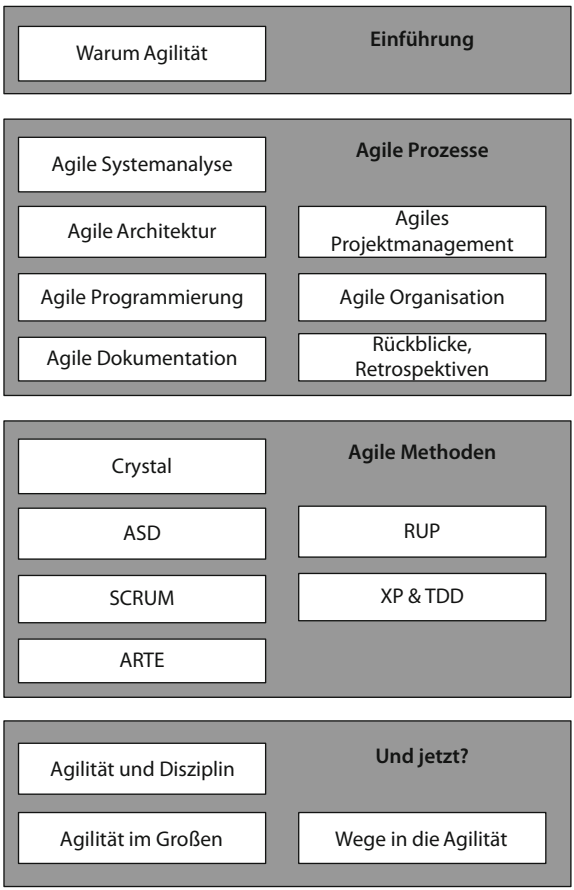
Die Bergwerke wuchsen zu gigantischen Unternehmen heran, ihre Besitzer häuften fantastische Reichtümer an. Jahr um Jahr fuhren die Bergleute in die dunklen Stollen, Jahr um Jahr blieb alles beim Alten. Die Herren und Arbeiter der Bergwerke glaubten, so würde es endlos weitergehen.

Doch vor einigen Jahren blieb die Nachfrage nach Kohle hinter der geförderten Menge zurück, die Preise sanken weltweit. Die Bergwerke warfen nicht mehr genug Rendite ab. Die Realität holte die langen Träume vom ewigen Reichtum und gesicherten Arbeitsplätzen abrupt ein.

Heute wird immer noch Kohle gefördert. Und einige der Beteiligten glauben immer noch, Bergwerke seien zukunftsträchtige Unternehmen.

Flexibilität

Wir können nicht besonders weit in die Zukunft schauen. Trotzdem versuchen wir immer wieder, IT-Projekte über Monate, teilweise Jahre im Voraus minutiös zu planen und diese Pläne einzuhalten, statt uns kontinuierlich der Realität anzupassen.



Warum Agilität?

Der Wechsel allein
ist das Beständige
Arthur Schopenhauer.

Bei der NATO-Konferenz 1968 in Garmisch-Partenkirchen wurde der Begriff „Software Engineering“ geprägt, um die anwesenden Wissenschaftler und Praktiker dazu anzuregen, als Reaktion auf die Softwarekrise eine ingenieurmäßige Vorgehensweise bei der Entwicklung von Software und Systemen vorzugeben. Viele Methoden und Verfahren wurden seither erfunden; Die Vorschläge, wie Softwareentwickler vorgehen sollen, wurden immer aufwendiger, die Handlungsanweisungen immer umfangreicher und dicker. Aber seien Sie einmal ehrlich: Sind Sie glücklich mit dem erreichten Stand? Bringen die Vorgehensmodelle und Methoden die erhoffte Wirkung? Erhalten Auftraggeber dadurch wirklich das, was sie haben wollen, rechtzeitig und zu akzeptablen Kosten? Und haben Entwickler Spaß daran, so ihre Software zu entwickeln? Werden diese Vorgaben auf breiter Front akzeptiert als die beste Art, Software und Systeme zu entwickeln?

Zu Beginn des neuen Jahrtausends haben Ward Cunningham und Kent Beck [Beck00] mit provozierenden Aussagen zu eXtreme Programming (XP) ein anderes Leitbild für Softwareentwicklung vorgegeben. Dieses neue Leitbild ist scheinbar in vielen Punkten die Abkehr von der Forschung der letzten 30 Jahre. Eine Gruppe von 17 praktizierenden Softwareentwicklern und Methodenforschern hat sich im Februar 2001 in Utah getroffen und das neue Idealbild für Softwareentwicklung in Worte gefasst: das agile Manifest war geboren. Darauf basieren heute eine ganze Reihe neuer Entwicklungsansätze, von denen eXtreme Programming nur einer, wenn auch der bisher bekannteste, aber leider auch der radikalste ist.

Lassen Sie sich durch dieses Buch zum Nachdenken anregen, ob Ihre Art Software zu erstellen diesem neuen Leitbild schon nahe kommt oder ob Sie noch nach Perfektionierung eines Vorgehensmodells streben. Wir haben für Sie in diesem Buch neben den Grundideen agiler Softwareentwicklung auch zusammengefasst, wie sich Agilität auf die unterschiedlichen Rollen in der Softwareentwicklung auswirkt, was es

für Projektleiter, Analytiker, Designer und Programmierer bedeutet, ja vielleicht sogar, wie eine ganze Organisation diesen Kulturwandel vollziehen kann.

Außerdem erläutern wir Ihnen die Kerngedanken einiger populärer agiler Methoden – die Aspekte, die sie von konventionellen Methoden unterscheiden.

Das agile Manifest – Ein neues Wertesystem

Worauf konnten sich die 17 Experten in Utah nach heftigen Diskussionen einigen? Lesen Sie im Folgenden eine Übersetzung der auf den Punkt gebrachten Aussagen des agilen Manifests (Die englische Fassung finden Sie unter www.agilemanifesto.org).

Wir entdecken bessere Wege zur Entwicklung von Software, indem wir Software entwickeln und anderen bei der Entwicklung helfen. Durch diese Tätigkeiten haben wir gelernt, dass uns

■ *Menschen und Zusammenarbeit*	*mehr als Prozesse und Werkzeuge bedeuten,*
■ *lauffähige Software*	*mehr als umfangreiche Dokumentation,*
■ *Zusammenarbeiten mit Auftraggebern*	*mehr als Vertragsverhandlungen und*
■ *Reagieren auf Änderungen*	*mehr als das sture Befolgen eines Plans.*

Natürlich sind auch die Dinge rechts wichtig, aber im Zweifelsfall schätzen wir die linken höher ein.

Überlegt man sich, was diese vier Wertaussagen für die vielen Tätigkeiten bei der Softwareentwicklung bedeuten und was sie für Konsequenzen auf das Verhalten der am Projekt beteiligten Personen haben, dann ergibt sich daraus ein ganz anderes Weltbild, als es viele Auftraggeber, Projektleiter und die Mitglieder im Projektteam heute gewohnt sind.

Mit dem Einleitungssatz positioniert sich die Gruppe von Experten: „Wir sind keine Methodenentwickler aus dem Elfenbeinturm. Wir stehen alle mitten in Softwareprojekten. Wir praktizieren diese Tätigkeiten aktiv, geben unser Wissen aber gerne auch an Kollegen und Kunden weiter. Und wir denken auch darüber nach, was wir hier

tun und ob es hilft oder eher schadet. Die Ergebnisse dieses Nachdenkens und Abwägens konnten wir in vier Grundsätzen zusammenfassen.“ Der Einleitungssatz fasst damit eine Grunderkenntnis agilen Handelns zusammen: Hören Sie auf diejenigen, die sich aus eigener Erfahrung mit Methoden auskennen. Vertrauen Sie erprobten, erfolgreichen Praktiken mehr als abstrakten Ideen von Theoretikern, die selbst schon lange keine Softwaresysteme mehr entwickelt haben. Der Nachsatz des Manifests schlägt die Brücke zu den Fortschritten in den letzen 30 Jahren. Die waren nicht alle falsch: der Versuch, den Entwicklungsprozess immer besser zu verstehen, zu definieren und Werkzeuge dafür zur Verfügung zu stellen; der Versuch, die „Zwischenergebnisse“ auf dem Weg zu auslieferbaren Systemen sauber zu dokumentieren; das Absichern von Projektzielen durch Verträge und das Vorausdenken, um in Form von Plänen dem Team Handlungsanweisungen mitzugeben. Aber vielleicht haben wir in all diesen Punkten übertrieben und damit die wirklich wichtigen Aspekte der Softwareentwicklung vernachlässigt.

eXtreme Programming war eine der ersten Methoden, die die abstrakten Ideen des agilen Manifests durch konkrete Vorschläge präzisiert hat. XP ist aber – wie der Name schon andeutet – sicherlich auch die extremste Interpretation dieses neuen Wertesystems. Etwas überspitzt formuliert wird darin u.a. Folgendes gefordert: Das einzige Ergebnis, das zählt, ist lauffähige Software. Wir entwickeln in so kurzen Zyklen (wenige Stunden bis maximal einige Tage), dass wir außer lauffähiger Software keine Dokumente brauchen. Auch keine Spezifikationen über die Problemstellung, denn der Kunde ist immer griffbereit im gleichen Raum, sodass Unklarheiten sofort beseitigt werden können (Mehr zu XP finden Sie im Abschnitt „eXtreme Programming (XP)“, Seite 84).

Angemessenheit

XP ist vielleicht unter gewissen Randbedingungen ein ideales Vorgehen – wenn es sich um relativ kleine Projekte handelt, mit einem einzelnen, hoch verfügbaren Kunden, der weiß, was er will, und einem Entwicklungsteam, das im selben Raum mit dem Kunden direkt Lösungen finden kann. Diese Randbedingungen sind aber nicht in jedem Projekt gegeben. Deshalb ist XP definitiv nicht immer und für jeden die richtige Antwort.

Andere agile Methoden setzen die Grundsätze des agilen Manifests in weniger radikale Handlungsanweisungen um. Statt „nur der Source Code reicht“ hören Sie z. B.: Natürlich ist unser Hauptziel einsetzbare, lauffähige Software, aber wir müssen uns auch richtig aufstellen für den nächsten Schritt. Und richtig aufgestellt sein bedeutet, dass den Entwicklern von Folgeversionen vielleicht neben dem Source Code auch ein ausgewogenes Maß an Zusatzdokumenten zur Verfügung stehen sollte (z. B.: Architekturdokumente, Regressionstestdaten, …), sodass sie nicht alles wieder durch Reengineering herausfinden müssen.

Aber wie viel Zusatzaufwand und Zusatzdokumentation soll ein Projekt erzeugen? Wie viel ist ausreichend und wo beginnt die bürokratische Übertreibung? Die Antwort muss jeder für jedes Projekt selbst geben. Der Schlüssel dazu liegt in einer Risikoabwägung: Was gewinnen wir, wenn wir die Schritte durchführen und die Ergebnisse erzeugen, bzw., was verlieren wir, wenn wir es nicht tun? Jeder Projektbeteiligte beurteilt daher für seinen Zuständigkeitsbereich ständig die Risiken und entscheidet dann über die geeigneten Maßnahmen. Agil heißt beweglich und flexibel sein, mitdenken statt „Dienst nach Vorschrift“ und Dogma.

Somit wird für agile Entwicklung die folgende Maxime zum Ausgangspunkt für alle Entscheidungen über Aktivitäten, Ergebnisse und Rollen in der Softwareentwicklung:

„eher Angemessenheit als Extremismus!“.

In den folgenden Abschnitten diskutieren wir, was diese risikogetriebene Angemessenheit für drei wichtige Faktoren in der Systementwicklung konkret bedeutet:

- Für das, was wir erreichen wollen: → Ergebnisse
- Für das, was wir (nicht) haben: → Zeit
- Für die, die an der Entwicklung beteiligt oder davon betroffen sind: → Menschen

Ergebnisorientierung

Überlegen Sie einmal Folgendes: Können wir ein erfolgreiches Softwareentwicklungsprojekt haben, wenn wir das Vorgehensmodell der Firma perfekt eingehalten haben, aber keine brauchbare Software geliefert haben? Wenn wir alle Dokumentationen abgegeben haben, aber keine laufende Software?

Ziel eines Softwareentwicklungsprojekts ist es, laufende Software zum Nutzen des Unternehmens in Betrieb nehmen zu können. Deshalb geht an dem Ergebnis Source Code in IT-Projekten kein Weg vorbei. Nutzen bringende Software sollte unser Hauptmaß für den Projektfortschritt sein. Ab einer gewissen Größe des Source Codes, wenn die Software langfristig für eine ganze Produktfamilie eingesetzt wird oder wenn an der Beauftragung und an der Entwicklung viele Personen an vielen Standorten beteiligt sind und sich abstimmen müssen, dann werden wir auch andere Projektergebnisse benötigen, z. B. Zielvereinbarungen, Architekturdokumente, Geschäftsprozessmodelle, Qualitätsrichtlinien und Spielregeln für die Projektorganisation und die Kommunikation im Projekt.

Die konkreten Gegebenheiten in Projekten, die Ziele und die Randbedingungen bestimmen, wie viel wir davon jeweils brauchen und in welcher Form. Je kleiner und lokaler das Projekt ist, desto weniger Vorschriften sind notwendig und desto weniger Formalismus muss dafür aufgewandt werden.

Sie sehen, dass „Agilität“ keineswegs die Erlaubnis zur Anarchie einschließt. Agile Vorgehensmodelle haben Ergebnisse, nur unterscheiden sich diese in unterschiedlichen Branchen und Projekten in Anzahl, Tiefgang und Formalität.

Der Unterschied zu „schweren“ Vorgehensmodellen besteht also hauptsächlich darin, dass es keine allgemeingültigen Vorschriften oder „Zwischenergebnisse“ auf dem Weg zum Projektziel (= lauffähiges, Nutzen bringendes Softwaresystem) gibt, sondern dass konkret für jedes Projekt Kosten und Nutzen von Zwischenergebnissen abgewogen werden. Was auch immer das Ergebnis ist, die Devise dafür muss sein: so wenig wie möglich, so viel wie nötig. Welchem Risiko setzen wir uns aus, wenn wir das Ergebnis nicht (oder nicht formal genug oder nicht umfangreich genug) erzeugen? Sie sollen in einem agilen Projekt keine Ergebnisse erzeugen, die mehr kosten als sie Nutzen bringen! Das Ergebnis muss jemandem etwas Wert sein. Sie sollten diejenigen kennen, denen das Ergebnis so viel Wert ist, wie Sie für die Erstellung oder Pflege aufwenden müssen. Suchen Sie diese „Sponsoren“ für jedes Ergebnis. Wenn Sie keinen Sponsor dafür finden, sollten Sie auch den Mut haben, das Ergebnis nicht zu produzieren. Auf keinen Fall sollten Sie sich hinter einem anonymen Vorgehensmodell verstecken, das die Erstellung bestimmter Ergebnisse verlangt. Als Maxime formuliert: Agile Projekte arbeiten