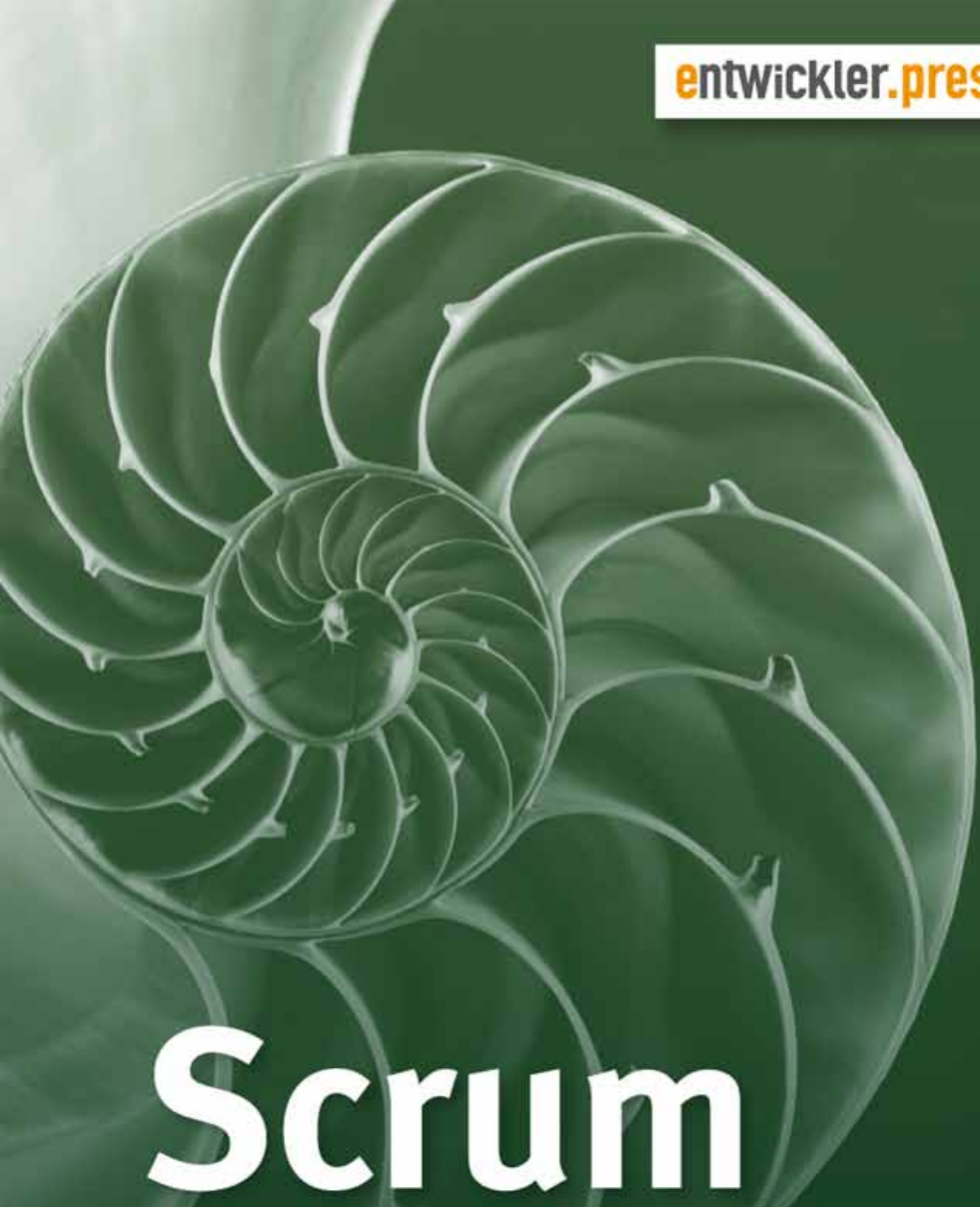
entwickler.press



Scrum

Schnelleinstieg

Andreas Wintersteiger

3. AUFLAGE

Neues Kapitel: „Scrum skalieren“

Andreas Wintersteiger

Scrum

Schnelleinstieg

entwickler.press

Andreas Wintersteiger
Scrum. Schnelleinstieg
(3. aktualisierte und erweiterte Auflage)

ISBN: 978-3-86802-341-1

Ein Imprint der Software & Support Media GmbH

Bibliografische Information Der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Ihr Kontakt zum Verlag und Lektorat:
Software & Support Media GmbH
entwickler.press
Darmstädter Landstraße 108
60598 Frankfurt am Main
Tel.: +49 (0)69 630089-0
Fax: +49 (0)69 630089-89
lektorat@entwickler-press.de
http://www.entwickler-press.de

Lektorat: Corinna Neu
Korrektorat: Frauke Pesch
Copy-Editor: Nicole Bechtel
Satz: Dominique Kalbassi
Umschlaggestaltung: Maria Rudi
Titelbild: © Maudib | istockphoto.com
Belichtung, Druck und Bindung: Media-Print Informationstechnologie GmbH, Paderborn

Für Valentina

Inhaltsverzeichnis

Vorwort zur 3. Auflage

Vor genau neun Jahren wurde die Objectbay GmbH gegründet mit dem Ziel, Beratungsdienstleistungen und Werkzeuge für agile Softwareentwicklung anzubieten. Unsere Reise führte uns über die Entwicklung eines Werkzeugs für agile Teams zu Entwicklungsleistungen im Auftrag für Kunden, zu letztendlich Beratung und Training für Lean und Agil, insbesondere Scrum. Heute machen wir zwei Dinge: wir entwickeln (natürlich lean und agil) für unsere Kunden und wir helfen anderen dabei.

Vor neun Jahren hielt ich mein erstes öffentliches Scrum-Training und habe zum ersten Mal einer großen Organisation geholfen, Scrum einzuführen. Ich bin stolz, dass wir heute auf eine Reise des stetigen Lernens und Perfektionierens zurückblicken. Nur wenig später musste ich mich erstmals den Herausforderungen in einem größeren, skalierten Umfeld stellen. Es gab wie zu Beginn meiner agilen Reise kurz nach der Jahrtausendwende nur wenig verfügbares Wissen, wie man Scrum nun im Großen einsetzt. Jedoch kursierten schon praktikable Lösungen in der Community.

Heute ist „Skalieren“ von Scrum eine große Sache, und alle Unternehmen fragen danach. Es gibt Frameworks und Best Practices. Eine Blaupause für agile Prozesse funktioniert leider nicht wirklich gut, deshalb ist die Einführung von Scrum im skalierten Umfeld genauso schwer wie die Einführung von Scrum im Pilotbetrieb. Skalieren von Softwareentwicklung heißt Skalieren von Kommunikation.

In diesem Buch liegen viele pragmatische und in der Praxis tatsächlich erprobte und funktionierende Tipps zum Einsatz von Scrum vor. In der dritten Auflage wollen wir dem Trend der Skalierung von Scrum mit einem eigenen Kapitel Rechnung tragen und den Praxisbezug noch weiter vertiefen.

Die wichtigste Erkenntnis aus 15 Jahren mehr oder weniger agilen Arbeitens ist jedoch, stets nach Verbesserung zu streben und den Status quo zu hinterfragen. Liefere Kundenwert früh und häufig und vermeide Verschwendung! Bei Lean habe ich begonnen und zu Lean bin ich zurückgekehrt!

Andreas Wintersteiger
Linz, im September 2015

Vorwort zur 2. Auflage

1983, vor ziemlich genau 30 Jahren, habe ich als Jugendlicher mein erstes Programm auf einem TI-99/4A-Heimcomputer geschrieben und, ohne es zu bemerken, damit die ersten Schritte in Richtung professioneller Softwareentwicklung gemacht. Drei Jahre später habe ich zum ersten Mal eine von mir entwickelte PC-Software verkauft, also mit Softwareentwicklung auch wirklich Geld verdient. Weitere vier Jahre später, zu Beginn meines Studiums, habe ich ein Unternehmen gegründet, das Software individuell für Kunden entwickelt. In den frühen Jahren meiner Karriere habe ich intuitiv und ohne lange darüber nachzudenken mit meinen Kunden eng zusammengearbeitet und häufig und in kurzen Zeiträumen funktionsfähige Software ausgeliefert.

Etwas später wurde meine Karriere „professioneller" und ich begann im Rahmen von Großprojekten und Anstellungen, Projektmanagementmethoden und Softwareprozessmodelle anzuwenden. Die Projekte wurden größer, umfangreicher und zunehmend komplexer. Ich war gezwungen, mehr zu planen, längere Entwicklungsphasen und spätere Lieferungen – bis zu über einem Jahr – waren die Folge. Irgendwann zwischen dem Schreiben meiner Dissertation und der Gründung eines weiteren Unternehmens wurde mir klar, dass diese als „professionell" geltenden Softwaremodelle für mich nicht gut funktionieren. Sie fühlten sich auch niemals gut an und ich war auch nicht wirklich erfolgreich damit. Ich begann, mich für alternative Vorgehensweisen zu begeistern und hatte auch bald die Chance, diese Ideen umzusetzen, noch bevor das Wort „Agil" hierfür verwendet wurde.

Gegen Ende der 90er Jahre hat für viele Menschen das Zeitalter der Wissensarbeiter begonnen. Wir arbeiten heute anders, und es sind nicht die definierten Prozesse, welche die Effizienz einer Organisation bestimmen, sondern die Menschen und wie sie zusammenarbeiten. Prozesse treten in den Hintergrund, Selbstorganisation und Autonomie in den Vordergrund. Leider haben auch heute noch viele Unternehmen nicht ausreichend verstanden, dass das Management von Wissensarbeitern deutliche Unterschiede zu den im vorigen Jahrhundert erfundenen Managementmethoden aus dem Zeitalter der Manufaktur und industriellen Produktion aufweist.

Ich bin froh, dass ich mir bereits 2002 die Chance gegeben hatte, aus dieser alten Welt auszubrechen, ein eigenes Unternehmen gegründet und letztendlich heute eine Organisation erschaffen habe, in der man agil arbeiten kann und welche die agilen Werte – eine völlig neue Kultur – tatsächlich lebt. Ich arbeite seit rund 13 Jahren wieder so oder so ähnlich wie in den frühen Jahren meiner Karriere. Kurze Lieferzyklen, enge Zusammenarbeit mit dem Kunden, Transparenz und iteratives Annähern an nicht vollständig bekannte Ziele mit selbstorganisierenden Teams.

Agile Softwareentwicklung ist in der Realität angekommen und funktioniert, und Objectbay, das von mir gegründete Unternehmen, ist neben tausenden anderen ein Beispiel, dass all die agilen Ideen und Prinzipen real funktionieren. Die Tatsache, dass ich seit annährend zwei Jahren unsere Scrum-Trainings auch in Indien halte, wo extrem präskriptive Vorgehensmodelle und jahrelange Projektlaufzeiten mit mehreren hundert Entwicklern üblich waren bzw. sind, untermauert den Erfolg von Scrum & Co. Doch Unternehmen ändern sich nur langsam und es ist sehr schwer, die bestehende Kultur zu trimmen.

Mit diesem Buch möchte ich Ihnen den Einstieg in diese neue Welt erleichtern. Mein Ziel ist es, das „Wie“ zu vermitteln, aber auch das „Warum“ und die Hintergründe zu beleuchten. Als ich 2011 begann, dieses Buch zu schreiben, hatte ich eine Vision: Mit einem kurzen und dennoch informativen Textbuch, das sich „übers Wochenende lesen lässt“ jenen Menschen zu helfen, die Scrum und agile Softwareentwicklung verstehen möchten, ohne sich einem monatelangem Studium der Theorie hin-

geben zu müssen. Diese Vision ist Realität geworden und ich danke den Lesern, die es mit dem Ausverkauf der Erstauflage samt korrigiertem Nachdruck zu diesem Erfolg gemacht haben.

In der zweiten Auflage haben wir an vielen Stellen nur kleine, aber wichtige Änderungen vorgenommen, um noch mehr Klarheit zu schaffen. Unser Ziel war aber auch, den Text nicht unnötig aufzublähen, dennoch wertvolle Hinweise zu einzelnen Themen zu geben.

Ich möchte an der Stelle insbesondere meinem Kollegen Daniel Haslinger danken, der begonnen hat, dieses Buch mit seinen Ideen und Erfahrungen zu ergänzen. Wir werden in Zukunft gemeinsam an dem Buch weiter arbeiten und die Erfahrungen und Erkenntnisse aus unserer täglichen Arbeit mit Scrum und agiler Softwareentwicklung einfließen lassen.

Andreas Wintersteiger
Bangalore, im Juli 2013

Vorwort zur 1. Auflage

Agile Softwareentwicklung verspricht viele Vorteile. So z. B. die Transparenz über den Projektfortschritt, große Flexibilität im Umgang mit Veränderungen und kürzere Time-to-Market. Aber auch die Motivation der Teammitglieder steigt und damit auch Ihre Effizienz. Scrum ist der mit Abstand prominenteste Vertreter agiler Vorgehensweisen. Das liegt mit Sicherheit an seiner Einfachheit, den wenigen und klaren Rollen, der übersichtlichen Anzahl an Meetings und einer geringen Zahl vorgeschriebener Artefakte. Trotz dieser Einfachheit gibt es nach meiner Erfahrung zu viele, die auf Halbwissen basierend mit Scrum beginnen, die wenigen Regeln nicht befolgen und sich wundern, dass Scrum nicht alle ihre Probleme löst. Die wenigen Spielregeln von Scrum sollte man schon kennen und ernst nehmen, wenn man von den Vorteilen, die Scrum bietet, profitieren will.

Andreas Wintersteiger hat mit diesem Buch eine sehr kompakte und doch sehr runde und inhaltsreiche Darstellung von Scrum vorgelegt, die jedem hilft, der sich einen fundierten Überblick und Einblick verschaffen will. Halbwissen zu Scrum existiert schon ausreichend, dieses Buch aber vermittelt Vollwissen.

Mir gefällt es sehr gut, dass Andreas seine Darstellung von Scrum an vielen Stellen um seine Meinung und seine Erfahrung aus der Coaching-Praxis ergänzt hat. So kann der Leser unterscheiden zwischen dem, was Scrum ausmacht, und dem, was der erfahrene Coach empfiehlt. Ich glaube, dass wir in unseren Erfahrungen und Empfehlungen einen sehr ähnlichen Blick auf Scrum haben, zumindest haben wir bei den zahlreichen gemeinsam gegebenen Scrum-Trainings immer hervorragend harmoniert. Danke für diese Trainings, Andreas, Danke für dieses Buch.

Henning Wolf, CEO it-agile GmbH
Hamburg, im Januar 2012

Danksagung

An erster Stelle bedanke ich mich bei meiner Frau Petra, die bereits die vergangenen Jahre hindurch mein massiv eingeschränktes Zeitbudget für Privates ertragen musste. Als ich ihr eröffnete, ein zweites Buch zu schreiben, unterstützte sie mich, ohne ein Wort darüber zu verlieren, welche Auswirkungen das auf unser Eheleben hat. Danke für deine Unterstützung und dein Verständnis.

Ein großes Dankeschön auch an meine Verleger und Lektoren bei entwickler.press, Eure Flexibilität und Geduld mit mir war und ist vorbildlich. An meine Mitarbeiter bei Objectbay möchte ich ein Dankeschön für die Unterstützung durch Lesen und Hinweisen auf Unverständlichkeiten richten.

Bei meinen Kollegen und Freunden in der Agile- und Scrum-Community möchte ich mich bedanken für die gute Zusammenarbeit und großartigen Diskussionen, die wir in den letzten Jahren führten. Viele Er-

kenntnisse und Erfahrungen beruhen darauf. Besonderer Dank gilt den Kollegen, mit welchen ich in den letzten Jahren gemeinsame Trainings und Coachings halten durfte. Danke an Peter Beck, Sven Blesin, Malte und Timo Foegen, Boris Gloger, Alessandro Grimaldi, Daniel Haslinger, Andreas Havenstein, Johannes Link, Christoph Mathis, Simon Roberts, Stefan Roock, Josef Scherer und Henning Wolf, sowie meinen Freunden und Kollegen in Indien.

Ganz besonderen Dank richte ich an unsere zahlreichen, internationalen Kunden, die uns ihr Vertrauen geschenkt haben und sich mit uns als ihre Begleiter auf die Reise begeben haben, Scrum im Unternehmen einzuführen. Viele praktische Erfahrungen stammen aus unseren Engagements bei ihnen – herzlichen Dank dafür. Ich hoffe, mit diesem Buch ein wenig davon zurückgeben zu können.

1 Einleitung

Agile Softwareentwicklung hat es in den Mainstream geschafft. Nach beinahe fünfzehn Jahren nach dem Agilen Manifest sind agile Vorgehensweisen keine exotischen Modelle mehr von Eigenbrötlern, die sich ihre eigene Welt schaffen, sondern eine anerkannte und bewiesenermaßen erfolgreiche Art und Weise, Software zu entwickeln[1]. Agile Entwicklung ist heute nicht nur erfolgreich, sondern deutlich erfolgreicher als klassische, wasserfallorientierte Vorgehensmodelle – dafür haben wir heute Evidenz.

Mussten wir vor zehn Jahren immer wieder den Beweis antreten, so sind wir heute einen bedeutenden Schritt weiter: Es ist nicht nur die Akzeptanz im breiten Feld vorhanden, ein großer Teil der Unternehmen strebt agile Entwicklungspraktiken und -methoden initiativ an. Darunter ganz besonders Scrum, das aus heutiger Sicht sicherlich das erfolgreichste Framework[2] unter den agilen Vorgehensweisen ist. Scrum erfreut sich zunehmender Beliebtheit und Verbreitung. Immer mehr Unternehmen und Teams setzen Scrum und agile Entwicklungspraktiken ein oder behaupten das zumindest.

Als Scrum Coach komme ich zu vielen Unternehmen, und als Trainer führe ich viele Gespräche mit Teammitgliedern oder Vertretern von Organisationen, die Scrum – oft schon seit Jahren – verwenden. Dabei stelle ich manchmal fest, dass es viele Missverständnisse gibt, was Agil und Scrum ausmacht und was nicht.

---

1 In Trainings habe ich schon öfter zu hören bekommen, dass Scrum und agile Methoden sich eine völlig eigene, künstliche Welt schaffen und die Teams, in dieser Blase lebend, so tun, als würde es die Außenwelt nicht geben.

2 Wir werden in Kapitel 4 genau betrachten, warum wir Scrum nicht als Methode, sondern als „Framework" bezeichnen.

1.1 Agile Softwareentwicklung

Im Gegensatz zu den klassischen Vorgehensweisen stützen sich agile Prozesse auf frühes und häufiges Feedback aufgrund von fertiggestellten Teillieferungen. Diese Teile müssen jedoch voll funktionsfähig sein, um den Anspruch an wertvolles Feedback zu erfüllen. Nur wirklich funktionierende Software erzeugt realistisches Feedback. Pläne, Diagramme oder Folienpräsentationen im Kontrast dazu verlangen danach, dass sich Menschen immer etwas vorstellen müssen, denn das Ergebnis ist (noch) nicht real. Je nach Komplexität und Vorstellungsvermögen der Beteiligten entsteht damit auch realistisches oder eben unrealistisches Feedback. Im klassischen Vorgehen entsteht reales Feedback oft erst nach Durchlauf vieler Phasen, was oft mehrere Monate, manchmal sogar mehr als ein Jahr dauert.

Iteratives Vorgehen

Im Kern einer jeden agilen Vorgehensweise stehen so genannte Iterationen, das sind Zyklen fixer Dauer, die zum Ziel haben, Feedback zu liefern. In iterativen Vorgehensweisen werden sämtliche Aktivitäten, die wir aus den klassischen Vorgehensmodellen kennen, wie Analyse, Design, Implementierung, Testen, Integrieren, Systemtest etc., durchgeführt, jedoch mehrmals. Wir durchlaufen diese Phasen in Zyklen.

Inkrementelles Vorgehen

Bei der inkrementellen Entwicklung wird das iterative Vorgehen verwendet, jedoch auf andere (neue) Teile des Systems angewendet (manchmal wird hierfür auch der Terminus „iterativ-inkrementell“ verwendet). Es handelt sich also um eine Erweiterung des Produkts, und die Iteration betrachtet damit eine neue Entwicklung[3] Bei der inkrementellen Entwicklung wird auch der Prozess verbessert. Das Ergebnis einer Iteration ist ein Inkrement des Produkts, es wird damit ausgehend von einem kleinen, funktionierenden Durchstich ständig erweitert und funktional gehalten.

3 In der iterativen Entwicklung werden diese Aktivitäten erneut durchlaufen, jedoch wird das bestehende Produkt betrachtet, bereinigt, erneuert oder Teile davon werden neu aufgebaut. Iteratives Vorgehen hilft dabei, das Produkt zu verbessern [14].

Inkremente sind beispielsweise Features oder Teile der Produktfunktionalität. Es ist wesentlich, festzustellen, dass bei der inkrementellen Entwicklung ein vertikal durch die Systemarchitektur liegender Querschnitt geliefert wird. In klassischen Vorgehensweisen geschieht oft das exakte Gegenteil, es werden dabei horizontale Schichten sequenziell und oft gänzlich geliefert. Ein funktionaler und damit feedbackfähiger Querschnitt ist erst nach Lieferung der letzten Schicht möglich. Aus diesem Grund erfolgt bei klassischer Entwicklung das Testen der Funktionalität sehr spät.

In der inkrementellen Entwicklung konzentriert man sich auf kleine (vertikale) Teile, die auch sehr früh getestet werden können. Alle anderen Aktivitäten können damit auch beinahe zeitgleich in einer Iteration erledigt werden (Details dazu werden in Kapitel 6 vertieft).

Das Entwicklungsvorhaben wird nicht mehr für den gesamten Umfang und damit auch nicht für die gesamte Zeit im Detail geplant. Eine Planung mit hohem Detailgrad beschränkt sich hier auf die kommende Iteration. Planung und Entwicklung und damit auch die Validierung der Planung wechseln sich in raschen Rhythmen ab.

Mit einem deutlich geringerem Detailgrad beschäftigt sich der nächsthöhere Planungshorizont, der der unmittelbar folgenden Iterationen: Hier werden nur mehr gröbere Ziele auf Basis der Erkenntnisse aus den bisherigen Iterationen geplant.

Mehr Kundenwert in kürzerer Zeit

Agile Prozesse liefern schneller bessere Ergebnisse und sind fokussiert auf die frühe Bereitstellung von Kundenwert. Es wird bei der Priorisierung auf die Auslieferung von Features mit hohem Kundenwert geachtet. Mit jedem Inkrement steht so nach wenigen Wochen ein lieferbarer Teil des Produkts zur Verfügung, das im Idealfall vom Kunden bereits genutzt werden kann.

In einem typischen Scrum-Projekt wird z. B. alle zwei Wochen ein voll funktionsfähiger Teil der Software geliefert und steht für Feedback durch Kunden, Anwender oder Management zur Verfügung. Durch den Fokus auf die frühe Auslieferung von Features mit hohem Kundenwert

entsteht deutlich früher ein bereits einsetzbares Produkt, das auf die Problemstellung fokussiert ist.

1.2 Agile Werte und Prinzipien

Die Bewegung in Form einer Ablehnung von klassischen, schwerfälligen Methoden und Vorgehensmodellen begann in den frühen 90er Jahren damit, dass einzelne Produktentwicklungsteams die bisher bekannten Prozesse der Softwareentwicklung wie iterativ und inkrementelles Vorgehen weitergetrieben haben. Darunter fallen auch Ken Schwaber und Jeff Sutherland, die Scrum entwickelt haben. Neben ihnen gab es eine Reihe weiterer, heute bekannter Persönlichkeiten, die ähnliche Praktiken einsetzen und damals noch „leichtgewichtig" genannte Methoden entwickelten. Viele von ihnen haben dazu auch auf wissenschaftlichen Konferenzen publiziert. Die zweite Hälfte der 90er Jahre war geprägt von vielen alternativen teilweise sehr revolutionären Ansätzen, die allesamt gegen die schwergewichtigen Methoden ankämpften.

Das Agile Manifest

Im Februar 2001 trafen sich diese Personen und versuchten zusammenzutragen, was ihre Vorgehensweisen gemeinsam haben, und kamen – obwohl sie es zuvor nicht glaubten – am letzten Tag zu einer Menge an Werten, die sie gleichermaßen hochhielten, und einem Dutzend Prinzipien sowie zu einem Konsens über deren Formulierung. Sie erschufen ein Manifest aus Werten, basierend auf Vertrauen und Respekt für einander, die die Organisationsmodelle fördern, die auf Individuen und Zusammenarbeit fußen. Übersetzt klingt das Manifest in etwa so:

Wir entdecken dadurch bessere Wege, Software zu entwickeln, indem wir es selber tun und andere dabei unterstützen, es zu tun. Durch diese Arbeit sind wir dazu gekommen,

- *Individuen und deren Interaktionen über Prozesse und Werkzeuge,*
- *funktionierende Software über umfassende Dokumentation,*

- *die Zusammenarbeit mit dem Kunden über Vertragsverhandlungen und*
- *das Eingehen auf Veränderung über das Befolgen eines Plans*

zu stellen. Während die Dinge auf der rechten Seite durchaus einen Wert darstellen, schätzen wir die Dinge auf der linken Seite mehr.

Sehr oft wurde dieses Manifest missverstanden und fehlerhaft interpretiert. Keineswegs war damit gemeint, dass es von nun an gar keine Dokumentation mehr geben sollte, sondern lediglich hinterfragt werden muss, wie viel Dokumentation nun wirklich notwendig ist und dass es besser ist, eine funktionierende Software zu haben, als schöne Dokumente, aber keine Software dazu.

Diese Aussagen wurden durch zwölf Prinzipien gestützt, die den Kern der agilen Softwareentwicklung definieren. Diese Prinzipien greifen ineinander und wirken als Ganzes:

1. Unsere höchste Priorität ist es, den Kunden durch frühe und kontinuierliche Lieferung werthaltiger Software zufriedenzustellen.
2. Wir begrüßen veränderte Anforderungen auch in einer späten Entwicklungsphase. Agile Prozesse nutzen Veränderungen für den Wettbewerbsvorteil des Kunden.
3. Liefere funktionierende Software häufig, von wenigen Wochen bis zu wenigen Monaten, mit einer Präferenz für kürzere Intervalle.
4. Anwender und Entwickler müssen täglich zusammenarbeiten.
5. Entwickle Projekte um motivierte Personen herum. Schaffe die Umgebung und Unterstützung, die sie benötigen, und vertraue ihnen, dass sie ihre Arbeit machen.
6. Die effizienteste und effektivste Methode, Informationen in ein und innerhalb eines Entwicklungsteams zu übermitteln, ist von Angesicht zu Angesicht.
7. Funktionierende Software ist das primäre Maß für Projektfortschritt.
8. Agile Prozesse fördern nachhaltige Entwicklung. Die Sponsoren, Entwickler und Anwender sollen unendlich lang ein konstantes Entwicklungstempo beibehalten können.
9. Kontinuierliche Aufmerksamkeit für technische Exzellenz und gutes Design unterstützt Agilität.

10. Einfachheit – die Kunst, unnötige Arbeit zu vermeiden – ist essenziell.
11. Die besten Architekturen, Anforderungen und Designs entstehen in selbstorganisierenden Teams.
12. Das Team reflektiert in regelmäßigen Abständen über die Verbesserung seiner Arbeitsweise, verbessert sie und passt sie an.

1.3 Agile Vorgehensweisen

Zum Zeitpunkt der Publikation des Agilen Manifests wurde auch die Agile Alliance durch die Teilnehmer und originalen Unterzeichner gegründet. Diese waren gleichzeitig auch Vertreter der damals breiten agilen Methodenlandschaft. Von den ehemaligen rund 20 Vertretern dieser agilen Methoden sind heute nur noch einige wenige relevante übrig geblieben, die in der Praxis tatsächlich Anwendung finden. Ich möchte kurz auf drei davon eingehen, die mir heute wichtig erscheinen: XP und Scrum sowie eine aus einer anderen Bewegung entstandene Methodologie: Lean bzw. darin die Methode Software Kanban.

1.3.1 Extreme Programming

Extreme Programming (XP) war zumindest in Europa früher deutlich bekannter als Scrum. Kent Beck publizierte 1998 die Methode erstmals mit vier Werten und zwölf Praktiken. In einer zweiten Ausgabe wurde sie leicht erweitert und hat nun fünf Werte, 13 Praktiken, außerdem sind 14 Prinzipien beschrieben.

Extreme Programming hat, wie der Name bereits vermuten lässt, seinen Schwerpunkt in den Praktiken der Entwicklung (Programmierung), definiert aber auch einen Ablauf (Prozess), siehe Abbildung 1.1.

Zu den Praktiken gehören unter anderem testgetriebene Entwicklung, kontinuierliche Integration, inkrementeller Entwurf, räumliche Nähe, Pair Programming, kollektive Verantwortung für den Code etc. Diese Praktiken gehören heute zum guten Ton bei der agilen Entwicklung und

haben damit für das Überleben von XP gesorgt. XP hatte immer einen stärkeren Fokus auf die Praktiken und weniger auf den Prozess.

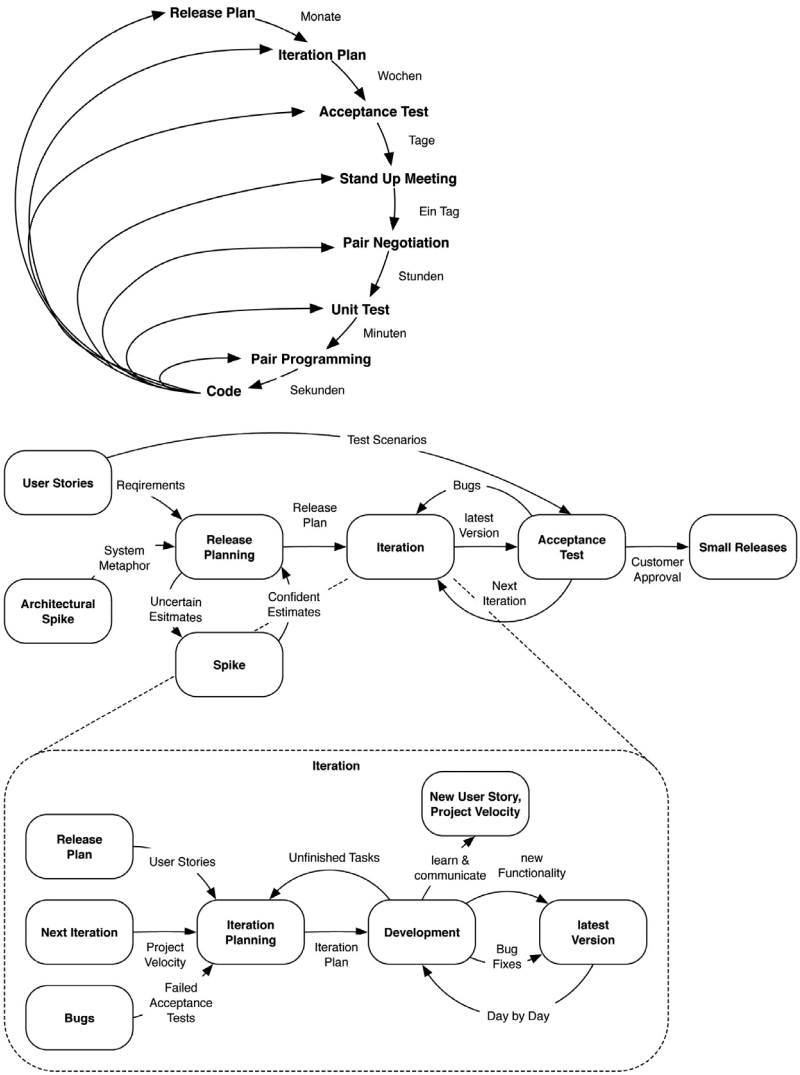


Abbildung 1.1: Feedbackzyklen und Prozesse in XP (vgl. [17] und [18])

XP definiert dennoch Rollen, Iterationen und auch Meetings, z. B. das tägliche Treffen (Daily Standup) oder die Retrospektive. Kent Beck hat viele Ideen aus Scrum übernommen und damit eine mächtigere Methode entwickelt. In XP liegt jedoch der Schwerpunkt auf den Feedbackzyklen, wovon es sieben definiert (Abb. 1.1). Diese greifen ineinander, sodass ein konzentrisches Zyklensystem entsteht, das vom Minutentakt bis zum Rhythmus von mehreren Monaten dauert.

Ein wesentliches Merkmal von XP ist auch die Unterteilung der Planung in unterschiedliche Horizonte und damit unterschiedliche Detailgrade.

1.3.2 Scrum

Bei Scrum hingegen findet man gar keinen Schwerpunkt auf technischen Praktiken. Scrum legt deutlich mehr Wert auf die geregelte Zusammenarbeit in selbstorganisierten Teams. Es definiert dazu drei Rollen, Iterationen mit vier Meetings und drei Artefakte. Eines der Artefakte ist die Forderung nach einem auslieferbaren Produktinkrement am Ende einer Iteration (in Scrum „Sprint“ genannt).

Betrachtet man die Mächtigkeit der Methoden, so ist XP deutlich mehr „verordnend“ als Scrum, hat also mehr Regulative. Um jedoch Scrum erfolgreich in der Praxis umzusetzen, z.B. das auslieferbare Inkrement am Ende einer Iteration, benötigt man gute Entwicklungstechniken, z.B. die aus XP bekannten technischen Praktiken.

Scrum wurde von Jeff Sutherland zum ersten Mal bei der Smalltalk-Firma Easel in der heute bekannten Form eingesetzt und hat seine Wurzeln in der empirischen Prozesssteuerung. Durch zeitnahe Feedbackschleifen wird in Scrum zunächst auf Prozessverbesserung fokussiert[4]. Der Name Scrum kommt aus dem Rugby-Sport, wo dieser Begriff übersetzt etwa „Gedränge“ bedeutet. Das Scrum in Rugby ist ein komplexer Vorgang, bei dem sich zwei Teams gegeneinander drängen, um auf einer Seite des

4 Wie wir im Rest des Buchs noch sehen werden, hat Scrum zwei Feedbackschleifen, um sowohl den Prozess zu verbessern als auch das Produkt inkrementell zu erweitern.