Projektmanagement-Wissen

SCRUM

Agiles Projektmanagement und Scrum erfolgreich anwenden

Auflage 2022

Abgestimmt mit ***The Scrum Guide™ 2020***

ROLAND WANNER

SCRUM

Agiles Projektmanagement und Scrum erfolgreich anwenden

ROLAND WANNER

Kontakt zum Autor:
Roland Wanner
E-Mail: info@rolandwanner.ch
Internet: www.rolandwanner.ch/

Herstellung:
Amazon Media EU S.à r.l, Luxembourg

Haftungsausschluss
Dieses Buch enthält Informationen über Scrum und agiles Projektmanagement. Es wurde zu Informations- und zu Weiterbildungszwecken geschrieben. Für den professionellen Einsatz empfiehlt sich die Unterstützung durch eine kompetente Fachperson.
Trotz größter Sorgfalt, dieses Buch so vollständig und korrekt wie möglich zu machen, ist nicht auszuschließen, dass es Fehler enthält, typografische oder inhaltliche. Deshalb ist dieser Text nur als genereller Leitfaden und nicht als alleinige Informationsquelle über Scrum und agiles Projektmanagement zu verwenden.
Der Autor, Herausgeber und die zitierten Quellen haften nicht für etwaige Verluste, die aufgrund der direkten oder indirekten Umsetzung der in diesem Buch verwendeten Beschreibungen entstehen könnten.
Themen in diesem Buch sind: Scrum, agiles Projektmanagement, IT und Sofware Projekte, Sprint, Timeboxing, Product Owner, Scrum Master, Daily Scrum und weitere.

Bei Fragen oder Anregungen kontaktieren Sie bitte:
info@rolandwanner.ch

Alle Rechte, einschliesslich derjenigen des auszugsweisen Abdruckes sowie der fotomechanischen und elektronischen Wiedergabe, vorbehalten.

Es wird darauf hingewiesen, dass die im Buch verwendeten Software- und Hardwarebezeichnungen sowie Markennamen und Produktbezeichnungen, wie z. B. The Scrum Guide™, der jeweiligen Firmen im Allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Copyright © 2018 Roland Wanner

ISBN: 978-1983653995
V1.6 September 2022, (1. Auflage Januar 2018)

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.

Inhalt

Vorwort 9

1 **Agiles Projektmanagement** **15**

Agiles Projektmanagement im Überblick 16
Der Unterschied zwischen klassischen und agilen Projekten 19
Agiles Projektmanagement für alle Projekte? 21
Was macht Teams in der Produktentwicklung erfolgreich? 23
Das Agile Manifest 25
Agile Prinzipien 28
Agile Methoden 31
Die Struktur des agilen Projektmanagements 34

2 **Das Scrum Framework** **37**

So entstand Scrum 37
Scrum als Management-Framework 40
Die drei Säulen von Scrum 41
Scrum im Schnellüberblick 44
Scrum und agile Projekte in der Unternehmens-Governance 49

3 **Das Scrum-Team** **51**

Der Product Owner 55
Die Developer 58
Der Scrum Master 59
Die Aufgaben des Scrum Masters 59
Eigenschaften und weitere Funktionen des Scrum Masters 61
Weitere Rollen 64
Das Scrum-Team und sein Umfeld 65

4 **Die Scrum-Werte** **67**

Die Scrum-Werte als das Fundament 69
Verpflichtung (Commitment) 70

Mut (Courage) ... 71
Fokus (Focus) ... 71
Offenheit (Openness) ... 72
Respekt ... 73
Die Scrum Werte leben ... 73

5 Die Scrum Events – Übersicht 75

Der Sprint ... 77
Das ... 78
Sprint Planning ... 78
Die Planungsschritte ... 79
Das Daily Scrum ... 80
Das Sprint Review ... 82
Die Sprint-Retrospektive ... 83

6 Die Scrum-Artefakte – Übersicht 85

Das Product Backlog ... 87
Commitment: Produkt-Ziel ... 88
Das Sprint Backlog ... 90
Commitment: Sprint-Ziel ... 90
Das Increment ... 91
Commitment: Definition of Done ... 91
Der Scrum Workflow ... 94

7 Das Anforderungsmanagement 97

Produktvision, Produktziel, Sprintziel ... 98
Die Produktvision ... 99
Das Product Vision Board ... 100
Anforderungsmanagement in Scrum ... 106
Der Anforderungsworkshop ... 108
Die Merkmale guter Anforderungen ... 111
User Stories ... 112
User Stories sollten „klein“ sein ... 116
Nichtfunktionale Anforderungen erfassen ... 117
Unterschied zwischen Epic, Theme, User Story und Task ... 118

Der Business Value 120

8 **Das Product Backlog** **123**

Übersicht über das Product Backlog 124
Das Product Backlog detaillieren 126
Den Aufwand schätzen 127
Die Risiken der Anforderungen bestimmen 128
Das Product Backlog priorisieren 130
Priorisierung nach MoSCoW 131
Priorisieren ist eine wiederkehrende Optimierungsaufgabe . 132

9 **Planung in Scrum** **133**

Wann startet das Projekt? 134
Bevor das Projekt startet 136
Releaseplanung 138
DevOps – Development and Operations 141
Aufwandschätzung 145
Aufwandschätzung auf zwei Ebenen 145
Das Estimation Meeting 147
Mit Punkten schätzen 151
Planning Poker 152
Magic Estimation 153
Die optimale Sprintdauer 155
Die Entwicklungsgeschwindigkeit bestimmen 156

10 **Das Sprint Planning** **159**

Der Sprint-Ablauf 159
Die Definition of Done (DoD) 165
Das Sprint Planning vorbereiten 167
Thema Eins: Warum ist dieser Sprint wertvoll? 168
Thema Zwei: Was kann im Sprint erledigt werden? 170
Thema Drei: Wie wird die ausgewählte Arbeit erledigt? 173

11 **Den Sprint durchführen** **177**

Artefakte und Events im Sprintablauf 177

Die Arbeit ausführen ... 179
Das Sprint Backlog ... 181
Das Sprint Backlog auf dem Task Board ... 183

12 Das Daily Scrum 187

Das Daily Scrum vorbereiten und durchführen ... 187
Den Sprint-Umfang anpassen ... 191
Einen Sprint abbrechen ... 193
Den Fortschritt überwachen ... 194
Das Taskboard ... 195
Das Sprint Burndown Chart ... 195
Das Release Burnup Chart ... 198
Das Story Burndown Chart ... 199
Der Themenpark ... 200
Das Velocity Chart ... 201
So wird das agile Projekt gesteuert ... 202

13 Das Sprint Review 205

Das Ziel des Sprint Reviews ... 205
Vorbereitung und Ablauf des Sprint- Reviews ... 206
Das Resultat des Sprint Reviews ... 209

14 Die Sprint Retrospektive 211

Kontinuierliche Verbesserung und stetiges Lernen ... 211
Vorbereiten der Retrospektive ... 213
Die Retrospektive durchführen ... 215
Sicherheit schaffen ... 215
Ereignisse visualisieren ... 216
Erkenntnisse sammeln ... 217
Maßnahmen definieren ... 218
Maßnahmen priorisieren ... 219

15 Zusammenarbeit im Scrum-Team 223

Die neue Team-Philosophie ... 224
Große Herausforderungen ... 225

Fest zugeordnetes Team ... 226
Limitierte Teamgröße ... 227
Selbstorganisierende Teams ... 227
Funktionsübergreifend arbeiten ... 231
Kontinuierliche Verbesserung und Lernen ... 232
Ownership übernehmen ... 233
Die Zusammenarbeit im Team ... 235
Zusammenarbeit mit dem Product Owner ... 235
Zusammenarbeit mit dem Scrum Master ... 235
Zusammenarbeit mit dem Auftraggeber ... 239

16 Scrum bei großen Projekten 241

Große Projekte und verteilte Teams ... 241
Der agile Blueprint für große Projekte ... 244
Der Scrum@Scale™ Guide ... 245
Scrum@Scale Übersicht ... 245
Der Scrum Master Cycle ... 247
Der Product Owner Cycle ... 253
Die Organisation großer und verteilter Projekte ... 259
Die Praktiken für große Projekte anpassen ... 260
Das Product Backlog bei großen Projekten ... 260
Releaseplanung bei großen Projekten ... 261
Den Aufwand bestimmen ... 261
Projektweites Sprint Review und Retrospektive ... 262

17 Glossar 265

18 Anhang 277

Internet-Links und empfohlene Literatur ... 277
Über den Autor ... 279
Literaturverzeichnis ... 280
Stichwortverzeichnis ... 282

Dieses Buch basiert auf dem **Scrum Guide™ 2020** von Ken Schwaber und Jeff Sutherland sowie auf dem **Scrum@Scale™ Guide** von Jeff Sutherland von Scrum Inc. von 2019

Vorwort

Ich gratuliere Ihnen, dass Sie sich mit agilem Projektmanagement und Scrum beschäftigen wollen! Dies ist das ideale Buch, mit dem Sie alles Wichtige über agiles Projektmanagement und Scrum lernen oder bestehendes Wissen weiter vertiefen. Dieses Buch ist ein praxisbezogener Leitfaden, welcher Sie durch alle Stationen eines Scrum-Projektes führt.

Scrum und Agile sind seit ein paar Jahren eines der wichtigsten Managementthemen und auch agiles Projektmanagement wird in Zukunft stark an Bedeutung gewinnen – nicht nur für Software-Entwicklungsprojekte. Mit dem Wissen in diesem Buch sind Sie auf dem neusten Stand und für eine interessante Zukunft gut gerüstet.

In diesem Buch erhalten Sie zum Start eine kurze und allgemeine Einführung in das agile Projektmanagement. Diese legt dann die Basis für den größten Teil des Buches, dass agile Projektmanagement mit Scrum für Software-Entwicklungsprojekte.

Schneller, günstiger, besser und mehr Business Value

Agile Methoden sind auf dem Vormarsch, nicht nur im Projektmanagement. Auch in anderen Managementbereichen versucht man langsam den Nutzen daraus zu ziehen, Prozesse zu verschlanken, sich noch mehr auf den Kunden und den Kundennutzen zu fokussieren, noch schneller auf Marktveränderungen zu reagieren und mehr mit selbstorganisierenden Teams zu „experimentieren“. Agile Methoden haben aber auch das Ziel, die Arbeit sinnvoller, interessanter und lohnenswerter zu gestalten. Wenn Sie schon etwas älter sind, dann kommen Ihnen einige dieser Themen vielleicht bekannt vor. Selbstorganisierende Teams wurden zum Beispiel schon seit den 1960er Jahren in verschiedenen Unternehmen erfolgreich eingesetzt – haben sich aber bis nie richtig etabliert.

Erfolgreiche Managementmethoden wiederentdeckt

Die Prinzipien, Methoden und Werte, auf denen das agile Projektmanagement aufbaut, gibt es schon seit Jahrzehnten. Sie stammen praktisch alle vom Toyota Production System.

Das Toyota Production System (TPS), welches nach dem zweiten Weltkrieg entstand war ein Treiber für viele wirkungsvolle Managementmethoden, wie Lean Production, Lean Management, Business Process Reengineering und Six Sigma. Auch selbstorganisierende Teams waren ein Kernelement des TPS und wurden dann in den 1950er und 1960er in verschiedenen japanischen und später in amerikanischen Unternehmen eingesetzt.

Alle diese Managementmethoden waren in den betreffenden Unternehmen sehr erfolgreich, haben sich aber erstaunlicherweise in der Wirtschaft nie richtig durchgesetzt und gingen fast vergessen. Einzig die Autoindustrie hat vom Toyota Production System und Lean Production gelernt und davon stark profitiert.

Die Arbeit in Projekten in der Industrie, auf dem Bau und in anderen Branchen wird schon seit Jahrhunderten in sequenziellen, aufeinander aufbauen Phasen ausgeführt. Als die Informationstechnologie und Softwareentwicklung in den 1960er Jahren entstand wurde die sequenzielle Vorgehensweise auch in diese Projekte übernommen. In den Folgejahren wurden verschiedene mehr oder weniger erfolgreiche iterative Softwareentwicklungs-Methoden entwickelt, um die Softwareentwicklung effizienter zu machen und um besser auf verändernde Marktanforderungen reagieren zu können

1995 präsentierten Jeff Sutherland und Ken Schwaber zusammen ein Dokument, welches die Scrum Methode für Softwareprojekte beschrieb. Sie entwickelten, basierend aus den sehr erfolgreichen, aber schon fast vergessenen Werten, Prinzipien und Methoden, die Sie vorhin kurz kennengelernt haben, die erfolgreichste Projektmanagement-Methode für die Softwareentwicklung – seit Software entwickelt wird – und dies ist SCRUM.

Für wen wurde dieses Buch geschrieben?

Dies ist ein Buch für alle, die am agilem Projektmanagement interessiert sind und wissen wollen, wie die bekannteste agile Methode in der Softwareentwicklung, Scrum, funktioniert.

Dies ist kein Buch mit allen Details, vielen Beispielen, Geschichten und Ausschweifungen. Für das gibt es umfassendere und viel teure Bücher. Hier lernen Sie konzentriert das Wichtigste über agiles Projektmanagement und Scrum.

Ob Sie bereits im Softwarebereich arbeiten, Student, Projektauftraggeber für Software oder bereits in einem agilen Projekt arbeiten – dieses Buch liefert Ihnen in kompakter Form das notwendige Wissen, agiles Projektmanagement und besonders Scrum besser zu verstehen und erfolgreich anzuwenden.

So ist das Buch aufgebaut

Zuerst lernen Sie, wie das agile Projektmanagement entstanden ist und was der Unterschied ist zum „normalen" Projektmanagement. Dann lesen Sie, was das „agile Manifest" ist und was die agilen Prinzipien für eine Bedeutung für Scrum und die anderen agilen Methoden hat.

Der Hauptteil dieses Buches beschäftigt sich natürlich mit Scrum. Zuerst gebe ich Ihnen **ab Seite 43** einen schnellen **Überblick über das Scrum Framework**. Damit erhalten Sie innerhalb von fünf Minuten ein Grundwissen und verstehen so später die Zusammenhänge besser.

Die folgenden Kapitel geben Ihnen einen **Überblick** über die Werte von Scrum, die Scrum Events und die Artefakte. Die weiteren Kapitel vertiefen dann das gelernte und gehen ins Detail der Scrum Artefakte und Ereignisse.

Sie werden während dem Lesen sicher feststellen, dass viele Themen in diesem Buch mehrfach in verschiedener Granularität vorkommen. Das ist kein Fehler oder nur um das Buch mit Seiten zu füllen, sondern dient dazu, damit Sie sich immer weiter in Scrum vertiefen und immer mehr Details der verschiedenen Themen kennenlernen.

Ein ideales Nachschlagewerk

Wenn Sie dieses Buch gelesen haben, dann ist es für Sie später auch ein ideales Nachschlagewerk. Am Schluss des Buches finden Sie ein umfangreiches **Glossar** und ein hilfreiches **Stichwortverzeichnis**, mit dem Sie schnell ein bestimmtes Thema im Buch finden und sich darüber informieren können.

Nicht zuletzt eignet sich dieses Buch ausgezeichnet **für Schulungen** und **das Studium** im Bereich Softwareentwicklung.

Der Scrum Guide

„The Scrum Guide™" von Ken Schwaber und Jeff Sutherland ist die offizielle Guideline für Scrum. Er definiert die Grundlagen von Scrum und wird periodisch aktualisiert. Dieses Buch basiert auf der neusten Version des Scrum Guides vom November 2020 und auf dem Scrum@Scale Guide vom Mai 2019.

Der Scrum Guide beschreibt in sehr knapper Form (ohne Abbildungen) die Mindestanforderung an ein Scrum-Projekt. Sie können ihn hier kostenlos heruntergeladen:

https://www.scrum.org/resources/scrum-guide.

Dieses Buch gibt Ihnen einen umfassenden Einblick in das agile Projektmanagement und Scrum mit vielen hilfreichen Abbildung und geht weit über den Scrum-Guide hinaus.

Das Agile Manifest

Das Agile Manifest für Softwareentwicklung ist der kleinste gemeinsame Nenner aller agilen Vorgehensmodelle – und damit auch für Scrum. Es beschreibt die „Zwölf Prinzipien Agiler Softwareentwicklung", die Sie hier nachlesen können:

http://agilemanifesto.org/

Terminologie in diesem Buch

Ich verwende in diesem Buch bewusst die bekannten, englischsprachigen Scrum-Begriffe für die *Rollen*, *Artefakte* und *Ereignisse*. So werden Sie nicht mit neuen Wortschöpfungen unnötigerweise verwirrt.

Dieses Buch ist absichtlich in der männlichen Form geschrieben, natürlich nicht um das weibliche Geschlecht auszugrenzen oder zu diskriminieren, sondern damit der Text einfacher zu lesen ist. Ich hoffe die weiblichen Leser haben Verständnis dafür.

1

Agiles Projektmanagement

Agile Methoden werden in immer mehr Projekten eingesetzt – in der Softwareproduktentwicklung schon seit vielen Jahren erfolgreich, bei anderen Projektarten stehen wir hier noch am Anfang. Der Trend zeigt aber eindeutig, immer mehr Unternehmen beschäftigen sich mit agilen Methoden oder sind bereits an deren Einführung. Auch in traditionellen Industrien, wie zum Beispiel im Automobil- oder Flugzeugbau, führt man damit Projekte bereits erfolgreich durch. Beispiele dafür sind Toyota, BMW oder SAAB Technologies.

Seit kurzem wird versucht agile Methoden auch in anderen Geschäftsprozessen einzusetzen als bei Projekten. Dies ist eine spannende Herausforderung, welche die gesamte Unternehmenskultur, Führungssysteme und die Zusammenarbeit stark beeinflussen wird. Ich bin mir aber nicht so sicher, ob viele Großunternehmen diesen Schritt in den nächsten Jahren schaffen werden. Kleinunternehmen sind diesbezüglich viel anpassungsfähiger.

Agile Methoden wie Scrum zu lernen und zu verstehen ist relativ einfach. Die agilen Werte und Grundprinzipien zu verinnerlichen

und zu leben ist hingegen einiges schwieriger – und hier haben viele Unternehmen noch Mühe.

Der Erfolg agiler Methoden auf Unternehmensebene kommt schlussendlich aber nur zustande durch radikale Veränderungen innerhalb der Organisationen und hier sind wir noch weit davon entfernt große Fortschritte zu machen. Auf Projektebene sind wir hier hingegen schon sehr weit fortgeschritten.

Agile Methoden haben einen radikalen Einfluss auf die Führungs- und Kompensationssysteme in Unternehmen. Manager verlieren auch Macht und Einfluss bei selbstorganisierenden Teams. Dies wird die Veränderung, speziell in der Unternehmenskultur nicht einfach machen.

Das Prinzip von selbstorganisierenden und funktionsübergreifenden Teams, Chefs als Coaches ohne Führungsfunktion und der Abbau von Managementebenen, war schon vor Jahrzehnten kurz ein „interessantes Thema". Ich hoffe in den nächsten Jahren haben wir damit mehr Erfolg.

In der Softwareentwicklung werden agile Methoden und Scrum schon seit mehr als 20 Jahren erfolgreich angewendet. Ein erster, wichtiger Schritt ist also schon gemacht!

Agiles Projektmanagement im Überblick

Imposante Projekte wurden schon vor tausenden von Jahren durchgeführt. Denken Sie z.B. an die Steinstruktur von Stonehenge, die 3500 Jahre v. Chr. und die ägyptischen Pyramiden, die 2500 v. Chr. erbaut wurden oder in der neueren Zeit an die mittelalterlichen Burgen, Festung, Schlösser und Kathedralen, die Dampfmaschine, Autos, die Atombombe oder die Wolkenkratzer. Dies waren teilweise riesige und komplexe Projekte für Ihre Zeit.

Software und Softwareprojekte gibt es jedoch erst seit ein paar Jahrzenten. In den 1950er Jahren war Software noch Teil der Hardware

und wurde als Programmcode bezeichnet. Ich kann mich auch noch gut an die Lochkarten erinnern, mit denen in den 1970er Jahren Werkzeugmaschinen gesteuert wurden. Die Lochkarten waren die Programme, die notwendig waren, damit Werkzeugmaschinen z. B. Teile fräsen konnten.

Softwareentwicklungsmethoden gab es bis in die 1960er Jahre noch keine. Der Systems Development Life Cycle (SDLC) war der erste, der in dieser Zeit entstand, mit dem Ziel, große, funktionale Business Systeme zu entwickeln. Alle Projekte wurden bis in die 90er Jahre nach dem sequenziellen Wasserfallmodell abgewickelt (siehe Abbildung Seite 19). Das heißt, alle Anforderungen wurden zu Projektbeginn festgelegt, dann wurden Konzepte, Spezifikationen und Pläne erstellt und dann das Produkt gebaut und im Markt eingeführt.

Die Softwareentwicklung in den 1990er Jahren wurde durch die objektorientierte Programmierung, durch den Aufstieg des Internets und den Dot.com-Boom geprägt. Hier war Time-to-Market und Firmenwachstum entscheidend, d.h. die Entwicklungszyklen für Software wurden immer kürzer.

Das Wasserfall-Modell war nicht mehr geeignet

Mit dem starren Wasserfall-Modell war man in der Software-Entwicklung deshalb immer weniger zufrieden, besonders weil die Projekte immer komplexer wurden, Produktlebenszyklen immer kürzer und das Umfeld und die Anforderungen dynamischer. Man benötige immer schneller brauchbare Software, nicht mit allen Funktionen, aber mit den Wichtigsten. Leichtgewichtiger, flexibler und schneller sollte die Softwareentwicklung werden und weniger Administration waren gewünscht.

Neue Methoden sollten den Softwareentwicklungsprozess flexibler und schlanker gestalten – als Gegenbewegung zu den eher schwergewichtigen und bürokratischen, traditionellen Methoden, wie eben z.B. dem Wasserfall-Modell. Diese Forderungen stießen eine aktive Entwicklung von verschiedenen Methoden in der Softwareentwicklung an:

- 1986 – Barry Boehm entwickelte erste Ansätze eines iterativen Software-Entwicklungsprozesses mit dem risikoorientierten Spiralmodel.
- 1991 – Rapid Application Development (RAD) wurde vorgestellt
- 1995 – Jeff Sutherland und Ken Schwaber präsentierten zusammen ein Dokument, welches die Scrum Methode erstmals beschrieb
- 1998 – Rational Unified Process (RUP) wurde vorgestellt
- 1999 – Extreme Programming (XP) wurde vorgestellt, das auf großes Interesse bei den Softwareentwicklern stieß.

Wie Sie sehen sind erste Ansätze zu agiler Softwareentwicklung bereits Anfang der 1990er Jahre zu finden. Popularität erreichte die agile Softwareentwicklung erstmals 1999 als Kent Beck, der Extreme Programming entwickelte, das Buch *Extreme Programming Explained* veröffentlichte.

Das Interesse an Extreme Programming ebnete den Weg auch für andere agile Prozesse und Methoden. Die Bezeichnung „agil" für diese Art der Softwareentwicklung wurde ausgewählt von 17 Vertretern von verschiedenen Softwareentwicklungsmethoden im Februar 2001, bei einem Treffen in Utah (USA). Dies als Ersatz für das bis dahin gebräuchliche leichtgewichtig (engl. lightweight). Bei diesem Treffen wurde auch das Agile Manifest (siehe Seite 25) formuliert. Daraus hat sich dann im Laufe der Jahre die Bezeichnung „agiles Projektmanagement" entwickelt, denn nicht nur Softwareprojekte lassen sich agil planen, umsetzen und steuern, sondern auch andere Projektarten.

Das Ziel agiler Softwareentwicklung ist es, den Entwicklungsprozess flexibler und schlanker zu machen, als dies bei den klassischen Vorgehensmodellen der Fall ist. Agile Softwareentwicklung zeichnet sich durch selbstorganisierende Teams, sowie eine iterative und inkrementelle Vorgehensweise aus. Es wird versucht mit geringem bürokratischem Aufwand und weniger Regeln auszukommen, damit man sich schnell an Veränderungen anpassen kann, ohne dabei das Risiko für Fehler zu erhöhen.

Der Unterschied zwischen klassischen und agilen Projekten

Bei **klassischen Projekten** werden vom internen oder externen Kunden (Auftraggeber) zu Projektbeginn klare Ziele und Anforderungen definiert, die sich während der Projektdurchführung möglichst nicht mehr ändern. Am Ende des Projektes erhält der Kunde das, was er am Anfang definiert hat.

Das Projekt wird strikt in nacheinander folgenden Phasen durchgeführt. Eine vorhergehende Phase muss beendet sein, bevor mit der nächsten gestartet werden kann. Das Projektresultat entsteht im Ablauf der Phasen, bis es dann am Ende der letzten Phase vollständig fertiggestellt ist. Dieser Ablauf wird Wasserfall-Modell genannt.

Je weiter das Projekt fortschreitet, desto weniger kann der Kunde hier Einfluss auf das Endergebnis nehmen. Eine große Einschränkung beim Wasserfallmodell ist, dass jede Änderung oder neue Anforderungen, die der Kunde in einer späteren Projektphase noch umgesetzt haben will, ein Mehrfaches kostet, als wenn er diese am Anfang definiert hätte.

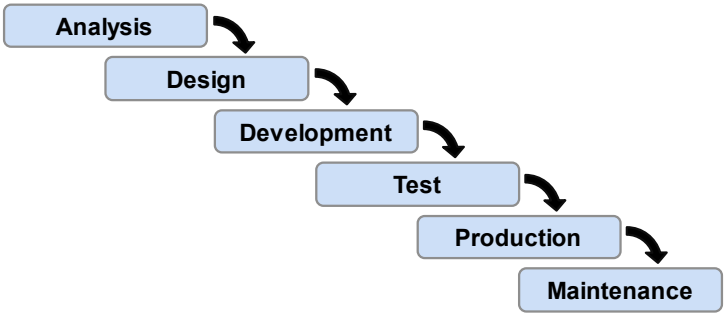


Abbildung 1: Das Wasserfall-Modell

Sie können sich vermutlich gut vorstellen, wie man ein Haus baut. Wenn bei diesem Projekt die Bauarbeiter bereits die Wände mauern

und Sie wünschen zu diesem Zeitpunkt noch ein zusätzliches Zimmer, dann wird dies sehr teuer oder es ist fast unmöglich.

Agile Methoden, z. B. Scrum werden in Entwicklungsprojekten (besonders von Software) verwendet, um sich der hohen Dynamik der Ziele, Anforderungen, Umfeld und Erwartungen anzupassen. Man setzt dabei z. B.:

- auf enge Zusammenarbeit zwischen dem Kunden, dem Product Owner und dem sich weitgehend selbst organisierenden Team
- auf kurze Entwicklungszyklen, nach denen Änderungen und neue Anforderungen in die Planung zusätzlich aufgenommen werden.

Damit macht es das agile Projektmanagement dem Auftraggeber oder den Stakeholdern möglich, neue Anforderungen während dem gesamten Projektablauf einzubringen oder bestehende Anforderungen zu ändern. So kann man auf kurzfristige Marktbedürfnisse flexibel einzugehen – und dies, ohne ein exponentielles Kostenwachstum zu verursachen. Dabei muss natürlich das Gesamtbudget trotzdem im Auge behalten werden.

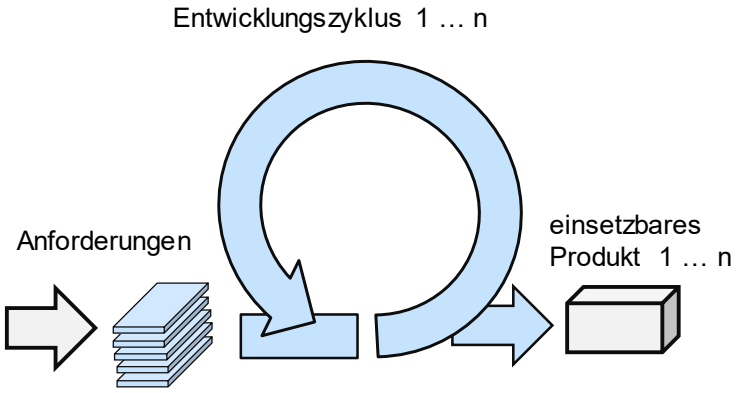


Abbildung 2: Das agile Vorgehen

Agiles Projektmanagement für alle Projekte?

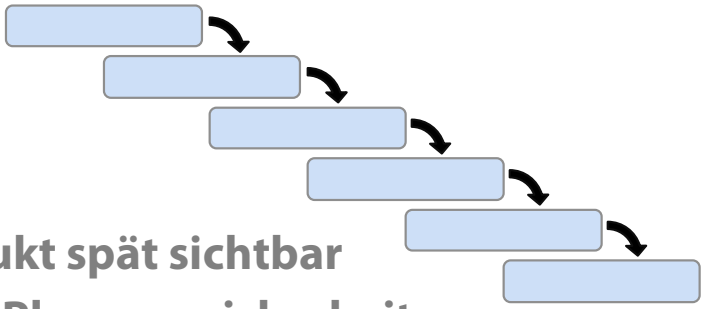
Agiles Projektmanagement setzt sich immer mehr durch, auch außerhalb der traditionellen Softwareentwicklung. Viele sagen dem Wasserfallmodell schon den Tod voraus. So weit wird es aber nicht kommen. Versuchen Sie ein Haus, einen Flugzeugträger, eine Werkzeugmaschine oder eine Fabrikationsanlage mit agilem Projektmanagement zu entwickeln und zu fertigen. Unmöglich! Dort müssen praktisch alle Anforderungen zu Projektbeginn bekannt sein, und der Spielraum diese während dem Projektablauf zu ändern oder neue Anforderungen hinzuzufügen ist sehr gering.

Bei der Entwicklung von „physischen Produkten“ wird das Wasserfall-Modell der bevorzugte Projektmanagement-Prozess bleiben, bei „immateriellen Produkten“, wie z.B. Software wird sich agiles Projektmanagement aber immer mehr durchsetzen.

Das heißt jedoch nicht, dass agile Methoden nicht für bestimmte Lieferobjekte von „Wasserfallprojekten“ eingesetzt werden können. Überall, wo zum Beispiel Software entwickelt werden muss, können wahrscheinlich agile Entwicklungsmethoden eingesetzt werden, zum Beispiel für die Software einer Werkzeugmaschine oder eines Auto-Unterhaltungs-Systems.

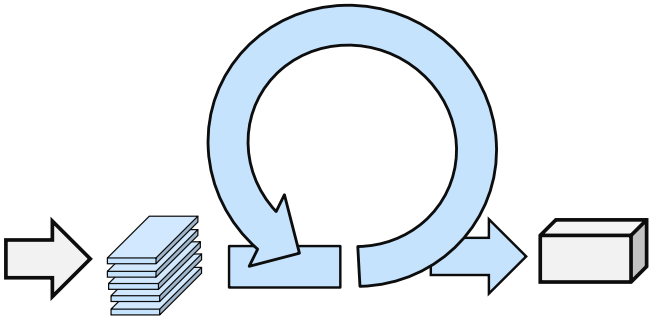
Bestimmte Prinzipien, Praktiken und Werte des agilen Projektmanagements werden sicher auch das traditionelle Projektmanagement in Zukunft stark beeinflussen. Ich denke nur schon an die Zusammenarbeit im Team und an die agilen Werte und Prinzipien. Aber überall, wo physische Produkte entwickelt werden ist die Flexibilität im Entwicklungsprozess eindeutig geringer als in der Softwareentwicklung.

Wasserfall-Modell



- starr
- Produkt spät sichtbar
- hohe Planungssicherheit
- konstante Anforderungen notwendig

Agiles Vorgehen



- wenig Struktur, iterativ
- früh einsetzbare Resultate
- hohe Flexibilität, marktnah
- ändernde Anforderungen willkommen

Was macht Teams in der Produktentwicklung erfolgreich?

Ein Kernelement im agilen Projektmanagement ist die Zusammenarbeit in Teams und das Anwenden von Wissen. Um mehr darüber zu erfahren machen wir kurz einen Abstecher in die Vergangenheit.

Hirotaka Takeuchi und Ikujiro Nonaka beschrieben 1986 im Harvard Business Review Artikel *„The New New Product Development Game"* die Eigenschaften erfolgreicher Produktentwicklungsteams. Der Inhalt des Artikels ist stark beeinflusst vom „Toyota Production System" und japanischen Produktentwicklungsmethoden z.B. bei Canon, Honda oder NEC aus den 1970er Jahren. In diesem Artikel werden wesentliche Eigenschaften von erfolgreichen Product-Development-Teams beschrieben:

1. **Built-in instability:** Das Management stößt den Entwicklungsprozess an, setzt herausfordernde Ziele und Anforderungen und gibt dem Team größtmögliche Freiheit in der Entwicklung.
2. **Self-organizing project teams:** Das Team arbeitet wie ein Start-up Unternehmen, es ist initiativ, geht Risiken ein und hat eine unabhängige Agenda: Die Teams charakterisieren sich durch folgende drei Eigenschaften:
   - *Autonomy*: Die Teams organisieren sich selber.
   - *Self-transcendence:* Stetiges lernen. Das Team strebt nach vorne und will immer besser werden.
   - *Cross-fertilization:* Die Teams arbeiten funktionsübergreifend. Diese Diversität fördert neue Ideen und Konzepte, was zu erfolgreicheren Produkten führt.
3. **Overlapping development phases:** Mit stark überlappenden Phasen wird der Entwicklungsprozess schneller und flexibler.
4. **Multilearning:** Durch Lernen auf verschiedenen Unternehmensebenen und in Gruppen, durch stetigen Erfahrungsaustauch und durch den Kontakt mit der Umwelt kann man schneller auf verändernde Marktbedingungen reagieren.

5. **Subtle control:** Das Management setzt genügend Checkpunkte, obwohl sich das Projektteam selbst organisiert. Damit sollen Instabilität, Unklarheiten und Spannungen vermieden werden.
6. **Organizational transfer of learning:** Gelerntes Wissen soll in die Organisation einfließen, um sich stetig zu verbessern.

Außerhalb Japans haben nur wenige Industrien Elemente dieses Konzepts in ihre Geschäftstätigkeit einließen lassen, und wenn, dann oft eher halbherzig. Abgeleitet aus diesem Wissen entstand in den 1990er Jahren das Lean Management und das Knowledge Management.

Die geringe Erfolgsquote bei Softwareprojekten in den 1990er Jahren lag nicht in der Ausbildung der Mitarbeiter oder an der fehlenden Reife, der noch jungen Informatik. Die Hauptgründe dafür waren schwergewichtige, sequentielle Implementierungsmethoden und unzureichende Zusammenarbeit in Projekten.. Ken Schwaber und Jeff Sutherland haben dies erkannt und dann aus dem Wissen von *„The New New Product Development Game“* und eigenen Erfahrungen ein erfolgreiches Softwareentwicklungs-Framework entwickelt – SCRUM.

Was hat Scrum mit Rugby zu tun?

Sie haben sich vielleicht gewundert, was Scrum mit Rugby zu tun hat. Das Titelbild dieses Buches zeigt ein „Scrum“ beim Rugby. Dies ist das Gedränge der Spieler beim Einwurf des Balles nach kleineren Regelverstößen.

Jeff Sutherland und Ken Schwaber haben sich vom Business Review Artikel „*The New New Product Development Game*“ inspirieren lassen, indem „Rugby“ und der Zusammenhang zu erfolgreichen, selbstorganisierenden Teams mehrmals erwähnt wurde. Darum benutzten Jeff und Ken für ihr neues Softwareentwicklungs-Framework einen Begriff aus dem Rugby – und dieser ist SCRUM.

Das Agile Manifest

Im Frühjahr 2001 verabschiedeten 17 projekterfahrene Software-Entwickler in Utah (USA) das sogenannte „Manifesto for Agile Software Development“, heute vor allem unter der Kurzbezeichnung „Agile Manifesto“ bekannt. Diese Erstunterzeichner, darunter auch die beiden Scrum-Gründer Ken Schwaber und Jeff Sutherland, formulierten mit dem Agile Manifesto erstmals die zentralen Werte agiler Softwareentwicklung – ein Meilenstein und zugleich das Fundament des agilen Projektmanagements.

Die Werte des Agilen Manifests (www.agilemanifesto.org) sind paarweise beschrieben, wobei die Werte auf der linken Seite jeweils höher eingeschätzt werden, als die Werte auf der rechten Seite. Dies heißt aber nicht, dass diese bedeutungslos sind.

Das Agile Manifest lautet folgendermaßen:

„Wir erschließen bessere Wege Software zu entwickeln, indem wir es selbst tun und anderen dabei helfen. Durch diese Tätigkeit haben wir diese Werte schätzen gelernt:

- **Individuen und Interaktionen** stehen über Prozessen und Werkzeugen
- **Funktionierende Software** steht über einer umfassenden Dokumentation
- **Zusammenarbeit mit dem Kunden** steht über der Vertragsverhandlung
- **Reagieren auf Veränderung** steht über dem Befolgen eines Plans

Das heißt, obwohl wir die Werte auf der rechten Seite wichtig finden, schätzen wir die Werte auf der linken Seite höher ein.“

Ich würde die agilen Werte des Agilen Manifests eher als „Grundsätze der agilen Softwareentwicklung bezeichnen“, damit diese nicht mit den Scrum-Werten verwechselt werden.

17 Glossar

Nachfolgend finden Sie eine Übersicht über die wichtigsten Begriffe von Scrum und dem agilen Projektmanagement, die ich in diesem Buch verwendet habe. Dies ist die erste Anlaufstelle, wenn Ihnen ein Begriff im agilen Projektmanagement unklar ist.

Accountable – Responsible (ausführungsverantwortlich) ist, ist die Person, von der erwartet wird, oder die verpflichtet ist, eine Sache zu tun. Accountable (ergebinsverantwortlich oder rechenschaftspflichtig) ist, der sicherstellt, dass die Arbeit erledigt und korrekt ausgeführt wird.

Agile Manifesto – Das im Jahr 2001 verabschiedete Manifest ist die Grundlage aller agilen Vorgehensweisen und beschreibt die zentralen agilen Werte.

Agile Prinzipien – Das Agile Manifesto beruht auf 12 Prinzipien, zu deren Einhaltung sich jeder Anwender agiler Methoden verpflichtet.

Agile Werte – Sind die wesentlichen Grundwerte für ein erfolgreiches, agiles Projektmanagement. Zusammengefasst: Mehr Flexibilität, weniger Strukturen, produktivere Zusammenarbeit. (siehe auch Agiles Manifesto)

Akzeptanzkriterien – Jedes Backlog Item (User Story) hat Akzeptanzkriterien. Diese sind Bestandteil der Fertigstellungskriterien (Definition of Done) und werden durch den Product Owner beim Review der fertigen User Story verwendet.

Anforderungen – Anforderungen werden in Scrum üblicherweise in einer User Story definiert (verpackt). Deshalb spricht man meistens von User Stories und nicht von Anforderungen.

Anforderungsliste – siehe Product Backlog

Artefakt – Artefakte sind die Ergebnisse von Aktivitäten im Softwareentwicklungsprozess. Zentrale Scrum Artefakte sind: Product Backlog, Sprint Backlog und das (auslieferbare) Product Increment.

Benutzergeschichte – siehe User Story

Burndown-/Burnup-Chart: Eine Grafik mit der die Developer die noch verbleibende Arbeit im Sprint in Relation zu der noch verbleibenden Zeit im Sprint visualisiert. Sie zeigt den Sprint-Fortschritt. Sie kann auf der Projekt-, Release- und Sprint-Ebene eingesetzt werden.

Business Value – Er ist der (meist finanzielle) Wert eines Backlog Items für ein Unternehmen oder den Kunden. Das Product Backlog wird meistens so sortiert, dass Items mit dem höchsten Geschäftswert als Erstes erledigt werden.

Commitment – (Verpflichtung) Das Developer verpflichten sich im Sprint Planning eine „Prognose“ zu liefern, welche Backlog Items im nächsten Sprint umgesetzt werden. Aber Commitment umfasst auch sich verpflichten zum Team, zur Qualität, zur Zusammenarbeit und zum Lernen. Sich verpflichten das Beste zu tun was man kann – jeden Tag.

Chief Product Owner (CPO) – Ist in Scrum@Scale die skalierte Rolle des Product Owners. Er ist verantwortlich für das Gesamtprodukt, die Produktvision und das übergreifende Product Backlog des Projektes. Der CPO koordiniert die Prioritäten der Product Owner, die mit einzelnen Teams zusammenarbeiten im MetaScrum.

Daily Scrum – Ist ein Meeting von 15 Minuten, innerhalb dem die Developer ihre Aktivitäten synchronisieren und an der Planung für die

nächsten 24 Stunden arbeiteten Es wird oft als Stand-up Meeting durchgeführt.

DEEP – Ein gute Product Backlog ist gemäß Mike Cohen DEEP. Dies bedeutet: Detailed, Estimated, Emergent (entwickelt sich), Prioritized.

Definition of Done (DoD) – Ist die Einigung eines agilen Teams darauf, was getan werden muss, damit ein Feature als fertig angesehen wird. Die DoD ist eine Liste von Fertigstellungskriterien.

Definition of Ready (DoR) – Ist eine Liste von Kriterien, die an die Product Backlog Items gestellt werden, damit diese bereit (Ready) sind, um im Sprint daran zu arbeiten. Im Gegensatz zur DoD ist der Product Owner für die Einhaltung der DoR verantwortlich.

DevOps – Durch die enge Zusammenarbeit von *Development and Operations* im agilen Umfeld und durch Automatisierung im ganzen Lebenszyklus können Releases schneller, in höherer Qualität, zu niedrigen Kosten und Risiken geliefert werden.

Dienende Führung – (Leader who serves) Der Scrum Master ist Führer der dem Scrum-Team dient. Das heißt, er verfügt zwar über Einfluss, hat aber keine Autorität bezüglich der Arbeitsorganisation im Team. Er agiert als Coach und als Change Agent.

Dokumentation – Diese ist auch in agilen Projekten notwendig und hilfreich. Es gelten jedoch vorrangig die Grundsätze des Agilen Manifests (Working software over comprehensive documentation).

Done – In Scrum wird strikt unterscheiden zwischen »nicht fertig« und „fertig". Ein Backlog Item gilt als „Done", wenn alle Tasks erledigt sind und die Kriterien der Definition of Done erfüllt sind.

Developer – Developer führen alle Arbeiten aus, um die Anforderungen in ein nutzbares Produktinkrement umzusetzen. Sie organisieren ihre Arbeit selbst.

Entwicklungsgeschwindigkeit (Velocity) – Ist die Summe aller Aufwände (Story Points), die das Scrum-Team in einem Sprint umgesetzt hat. Über die Zeit pendelt sich die Entwicklungsgeschwindigkeit auf einen relativ stabilen Wert ein. Die Entwicklungsgeschwindigkeit zu kennen ist für den Product Owner wichtig, damit er den

Releaseplan erstellen und im diesen im Verlauf des Projektes justieren kann.

Epic – Ist eine grobgranulare Anforderung (große User Story), die zu groß oder zu komplex ist, um Sie in einem Sprint umzusetzen. Sie wird später, wenn die Kundenanforderungen klarer und mehr Details bekannt sind, im Product Backlog aufgeteilt in User Stories.

Estimation – Damit ist meistens das Schätzen des Aufwandes von Product Backlog Items, bzw. User Stories gemeint. Schätzmethoden sind z.B. Planning Poker und Magic Estimation.

Estimation Meeting – Das Estimation Meeting (auch Backlog Refinement genannt) ist das Planungsmeeting mit dem Product Owner, um Klarheit für den nächsten Sprint zu erhalten. Ziel dabei ist: Details zu klären, Aufwände zu schätzen und die Priorisierung zu überprüfen.

Event (Ereignis) – Siehe Scrum Event.

Extreme Programming – Ist eine agile Softwareentwicklungsmethode, die mehrheitlich von Kent Beck entwickelt wurde. Sie folgt den fünf zentralen agilen Werten: Kommunikation, Einfachheit, Respekt, Feedback und Mut.

Feature —Ist eine funktionale Eigenschaft des zu entwickelnden Produktes.

Geschäftswert – siehe Business Value

Hindernis (Impediment) – Sind Probleme jeglicher Art, welche die Developer bei der effektiven Ausführung ihrer Arbeit behindern.

Impediment – siehe Hindernis

Impediment Backlog – Ist eine öffentliche im Teamraum aufgehängte Arbeitsliste des Product Owners mit allen Hindernissen.

Increment – Ein Increment ist ein Teil eines Ganzen. Unter einem Product Increment im Scrum versteht man das Ergebnis eines Sprints, als fertiges Teilstück des Gesamtproduktes.

Inspect & Adapt – Ein Prinzip der permanenten Verbesserung. Die eigene Arbeit wird in regelmäßigen Abständen überprüft und die

Arbeitsprozesse, wenn notwendig, in der nächsten Iteration angepasst.

INVEST – Gute User Stories entsprechen INVEST-Kriterien. Sie sind: (I) Independend *unabhängig*, (N) Negotiable *verhandelbar*, (V) Valuable *nützlich*, (E) Estimable *schätzbar*, (S) Small *klein*, (T) Testable *testbar*

Iteration – Ein Zeitabschnitt, in dem ein Inkrement des Produktes entwickelt wird. In Scrum wird eine Iteration „Sprint" genannt und dauert immer gleich lange.

Iterativ-inkrementell – In Scrum wird diese Technik verwendet, um durch kurze Feedbackzyklen Software zu erstellen. Iterativ (in einzelnen Sprints), werden Inkremente (fertige Produktteile) geliefert.

Kunde – Ist derjenige, der das Produktergebnis beauftragt (der Auftraggeber) und bestimmt, was dieses leisten soll und auch dafür bezahlt.

Lean Management – Ist eine Management-Philosophie, die Denkprinzipien, Methoden und Verfahrensweisen zur effizienten Gestaltung der gesamten Wertschöpfungskette von Produkten umfasst.

Magic Estimation – Ähnliche Aufwandschätz-Methode für User Stories wie Planning Poker®, jedoch einiges effizienter.

Manager – (Linien-) Manager sind wichtige Stakeholder auch bei Scrum-Projekten. Sie können das Projekt unterstützen und Hindernisse beseitigen, aber auch ein Umfeld schaffen, indem agile Prinzipien und Projekte sich entwickeln können.

MetaScrum – In Scrum@Scale ist dies die skalierte Version des Backlog Refinement Meeting der Product Owner. Die PO's koordinieren einen Backlog selbst, welches dann ein Scrum of Scrums (SoS) speist. Zu jedem SoS gibt es ein zugehöriges MetaScrum.

Minimal Marketable Feature (MMF) – Ist die kleinstmögliche Menge an Funktionen, die für sich genommen vermarktbar ist. Software, die nur eines dieser Merkmale aufweist, hat einen Nutzen für den User, für den dieser bezahlen würde.

Minimal Viable Product (MVP) – Das Minimal Viable Product ist die minimale Menge von Features, die notwendig sind, um herauszufinden, was ein Kunde möchte. Es ist eine Strategie, um schnelles und breites Feedback vom Kunden zu erzielen, ohne das Produkt bis zu seiner Marktreife auszuarbeiten.

MuSCoW – Die MuSCoW-Methode ist eine Technik zur Priorisierung, um Backlog Items grob in vier Kategorien einzuordnen: Must have, Should have, Could have, Won't have.

PBI – Product Backlog Item. Ein Element des Product Backlogs

PDCA-Zyklus – Ist ein vierstufiger Managementprozess, der zur ständigen Verbesserung bei der Softwareentwicklung verwendet wird: Plan (P), Do (D), Check (C) und Act (A).

Planning – siehe Sprint Planning

Planning Poker® – ist eine Technik für die Aufwandsschätzung von Backlog Items im Estimation Meeting. Dabei wird besonders Wert darauf gelegt, dass die Grösse eines Features geschätzt wird und nicht der Aufwand.

Product Backlog – Das Product Backlog eine Liste von Anforderungen (User Stories), die im laufenden Projekt umgesetzt werden sollen. Es wird vom Product Owner priorisiert, bereitgestellt, verantwortet und gemeinsam mit den Developern gepflegt.

Product Increment – Ist lauffähige, getestete und dokumentierte Software, die Anforderungen aus dem Product Backlog umsetzt. Das Product Increment ist das Ergebnis eines Sprints.

Product Owner – Der Product Owner ist für den wirtschaftlichen Erfolg und das „Was" im Scrum-Projekt verantwortlich. Er plant und steuert die Entwicklung in Scrum. Er ist Owner des Product Backlogs, priorisiert es und legt damit fest, welche Features wann realisiert werden.

Product Owner Team – Eine Gruppe von Product Owner, die ein gemeinsames Backlog koordinieren, dass ein Netzwerk von Teams speist. Für jeden Scrum of Scrums (SoS) gibt es ein assoziiertes Product Owner Team, das die Prioritäten der Teams auf einem gemeinsamen Weg abgleicht.

Product Backlog Item (PBI) – Dies ist eine Anforderung, üblicherweise beschrieben in einer User Story. Es ist eine Arbeitseinheit, die klein genug ist, damit das Scrum-Team diese in einem Sprint abschließen kann.

Product Backlog Refinement – So wird die laufende Pflege des Product Backlog bezeichnet. (siehe auch Estimation Meeting)

Produktvision – Die Produktvision wird vom Product Owner vor dem ersten Sprint erstellt. Sie leitet das Scrum in die richtige Richtung und ist ein übergeordnetes Ziel, dass jeder teilen muss, das Scrum-Team, der Kunde und die Stakeholder. Die Vision beschreibt, warum das Projekt unternommen wird und was der erwünschte Endzustand ist.

Prognose – Das Developer liefern im Sprint Planning eine Prognose, welche Backlog Items sie im nächsten Sprint umsetzen.

Release – Ist eine Softwareversion, die vom Anwender produktiv eingesetzt werden kann. Ein oder mehrere Release sind das Ergebnis des Scrum-Projektes.

Release-Burndown-Chart – Dies ist ein Burndown-Chart, das für jeden Zeitpunkt eines Releases zeigt, wie viele Story Points bis zum Releaseende noch umzusetzen sind.

Releaseplanung – Ist die Mittelfristplanung des Projektes und gibt dem Kunden eine Übersicht, wann er welche Funktionalitäten geliefert bekommt.

Scaled Daily Scrum (SDS) – Ist ein Events von Scrum@Scale. Bei großen Projekten spiegelt der SDS-Event das Daily Scrum wider, indem es die Zusammenarbeit und Leistung des Netzwerks von Teams optimiert.

Scrum Ereignisse (Events) – In Scrum gibt es fünf Event. Vier sind Meetings des Scrum-Teams: Sprint Planning, Daily Scrum, Sprint Review, Sprint-Retrospektive und das fünfte ist der Sprint selbst.

Scrum Guide – Wird herausgegeben und regelmässig updated von Ken Schwaber und Jeff Sutherland und beschreibt die offiziellen Scrum Grundlagen. Die neuste Version ist vom November 2017.

Scrum@ScaleTM Guide – Wird herausgegeben von Jeff Sutherland und Scrum Inc. und beschreibt wie Scrum einfach skaliert werden kann. Die erste Version ist im Februar 2018 erschienen.

Scrum Master – Der Scrum Master ist eine Scrum-Rolle. Er hilft, Scrum richtig anzuwenden, unterstützt das Team und stellt die optimale Zusammenarbeit zwischen Product Owner und Developern sicher. Der Scrum Master beseitigt Hindernisse und hilft dem Team, seine Produktivität und Reife kontinuierlich zu steigern.

Scrum of Scrums (SoS) – Ist ein Team von Scrum Mastern und weiterer Fachfunktionen der beteiligten Scrum-Teilprojekte. Ein SoS agiert als Release-Team und muss in der Lage sein, dem Kunden direkt Geschäftswert (Value) zu liefern.

Scrum of Scrums Master (SoSM) – Ist eine Rolle des Scrum@Scale. Es ist ein Team von Scrum Masters welches am Ende jedes Sprints verantwortlich ist für ein vollständiges Paket von potenziell lieferbaren Produktinkrementen von allen beteiligten Teams.

Scrum-Rollen – Sind: Product Owner, Scrum Master, Developer.

Scrum-Team – Besteht aus den drei Scrum Rollen: Product Owner, Scrum Master, Developer.

Scrum-Werte – Scrum basiert auf fünf Werten: Verpflichtung, Mut, Fokus, Offenheit und Respekt. Wenn diese Werte durch das Scrum-Team verkörpert und gelebt werden, werden die Scrum-Säulen *Transparenz, Überprüfung* und *Anpassung* lebendig und bauen bei allen Beteiligten Vertrauen zueinander auf.

Selbstorganisation – Ist ein wesentliches Merkmal von Scrum-Teams. Das Team organisiert seine Arbeit selber. Entscheidungen werden im Team getroffen und das Team einigt sich auf teaminterne Regeln und Arbeitsweisen. Einzig die Erreichung des Sprint-Ziels ist ein Maßstab für das Team.

Selected Product Backlog – siehe Sprint Backlog.

Servant Leadership – Der Scrum Master verkörpert die Prinzipien der „dienenden Führung“. Servant Leadership beschreibt die grenzenlose Selbstverpflichtung der Führungskraft gegenüber dem Team bzw. der Organisation.

Sprint – In Scrum werden Iterationen als Sprint bezeichnet. Ein Sprint hat eine feste Dauer, in welcher die Developer die Anforderungen des Kunden in funktionsfähige Software umsetzen.

Sprint Backlog – Im Sprint Planning entscheiden die Developer, wie viele der am höchsten priorisierten Backlog Items sie im folgenden Sprint implementieren wollen. Die gewählten Items sind dann das Sprint Backlog für den folgenden Sprint. Das Sprint Backlog entspricht der *Forecast* der Developer, was es im startenden Sprint gemeinsam erreichen will.

Sprint Burndown Chart – siehe Burndown Chart

Sprint Planning Meeting – Das Sprint Planning Meeting findet zu Beginn eines Sprints statt und besteht aus zwei Teilen. Beide Meetings dauern jeweils max. vier Stunden für einen 4-Wochen Sprint.
Im ersten Teil definiert das Scrum-Team das Sprint Backlog, also WAS es im nächsten Sprint umsetzen will. Im zweiten Teil werden die Implementierungsdetails, das WIE geklärt. Am Ende des Sprint Plannings sind alle User Stories (WAS) und die dazugehörigen Tasks auf dem Taskboard dargestellt.

Sprint-Retrospektive – Wird nach jedem Sprint-Review durchgeführt. Der Sprint und die Geschehnisse werden analysiert und es werden Maßnahmen für die nächsten Sprints definiert, mit dem Ziel stetig zu lernen und besser zu werden.

Sprint-Review – Findet am Ende jedes Sprints statt, mit dem Ziel die entstandenen Arbeitsergebnisse dem Product Owner, dem Kunden und den Stakeholdern vorzustellen und Feedback zu erhalten. Der Product Owner überprüft dabei, ob die Developer alle User Stories, gemäß Prognose, umgesetzt haben.

Sprint-Abbruch – Der Product Owner kann einen Sprint abbrechen, wenn er sieht, dass das Sprint-Ziel entweder nicht erreicht werden kann oder inzwischen nicht mehr sinnvoll ist. In diesem Fall wird ein Sprint-Review durchgeführt und alle abgeschlossenen „Done“ Product Backlog-Einträge begutachtet und wenn brauchbar vom Product Owner abgenommen.

Sprint-Ziel – Das Sprintziel wird vom Product Owner definiert und beschreibt das auszuliefernde Produktinkrement – möglichst in einen Satz. Es ist das, was der Product Owner am Ende des Sprints an den Kunden ausliefern will.

Stakeholder – (Interessensvertreter) werden Personen bezeichnet, die vom Projektergebnis betroffen oder daran interessiert sind, aber nicht am Projekt aktiv mitwirken, z.B. Anwender, Kunde, Vertrieb, Marketing, Manager.

Sprint 0 – So wird oft ein initialer Sprint genannt. Dieser ist meist kürzer als ein normaler Sprint und produziert keinen Business Value. Hier wird z.B. das Product Backlog vorbereitet, die Infrastruktur aufgesetzt, Architekturfragen geklärt und Regeln definiert, wie das Scrum-Team zusammenarbeitet.

Story – siehe User Story

Story Card – Beschreibt ein Backlog Item (User Story) auf einer Karteikarte mit max. zwei Sätzen. Auf der Vorderseite steht zusätzlich der Business Value und der Aufwand in Story Points. Auf der Rückseite stehen z.B. noch Abnahmekriterien und die Teststrategie.

Story Point – Story Points sind eine abstrakte Einheit, mit welcher der Aufwand (Grösse) eines Backlog Items bewertet wird, immer relativ zu einer anderen Story. Der Aufwand kann nur bestimmte vordefinierte Werte annehmen, die meistens der Fibonacci-Sequenz entsprechen.

Task – Im Sprint Planning definieren die Developer die Tasks (Aktivitäten, Aufgaben), mit denen die User Stories umgesetzt werden. Diese werden dann auf dem Taskboard neben den User Stories befestigt.

Taskboard – Das Taskboard visualisiert das Sprint Backlog. Darauf lässt sich jederzeit erkennen, welche Product Backlog Einträge für den Sprint ausgewählt wurden, welche Aufgaben (Tasks) dazu zu bearbeiten sind, und in welchem Bearbeitungszustand diese Aufgaben sind.

Team – siehe Scrum-Team

Timebox – Alle Scrum Events haben eine zeitliche Beschränkung (Timebox). Der Event wird nach einer bestimmten Zeit beendet, auch wenn diese noch nicht inhaltlich abgeschlossen ist. Die Timebox hat zum Ziel, Verschwendung zu vermeiden, um sich auf die wirklich wichtigen Dinge im Sprint zu fokussieren.

User – Als User bzw. (End-) Nutzer bezeichnet man die Personen, die das Projektergebnis nutzen. Scrum stellt das Liefern von Funktionalität für einen User in den Vordergrund, nicht die Erfüllung von Aufgaben. Es ist eine bewährte Vorgehensweise, die Backlog Items aus Sicht des Nutzers in einer User Story zu beschreiben.

User Story – Auch Product Backlog Items (PBI), bzw. Anforderungen oder Features genannt, werden meistens in User Stories beschrieben. User Stories beschreiben Anforderungen aus der Sicht des Anwenders. Verfasst werden User Stories in einer speziellen Schreibweise: Als <Rolle> möchte ich <Ziel/Wunsch>, um <Nutzen/Grund> erreichen.

Velocity – siehe Entwicklungsgeschwindigkeit

Vision – siehe Produktvision

Werte – siehe Scrum-Werte

18 Anhang

Internet-Links und empfohlene Literatur

Auf der Internetseite http://www.rolandwanner.ch finden Sie eine Liste mit Links und Artikel zum agilen Projektmanagement, Projektcontrolling, Earned Value Management und Risikomanagement in Projekten.

Auf meinem Blog http://www.rolandwanner.ch/blog finden Sie interessante Artikel zum agilen Projektmanagement, Projektcontrolling, Earned Value Management und Risikomanagement in Projekten.

Folgende ergänzenden Scrum-Bücher empfehle ich:

Scrum in der Praxis: Erfahrungen, Problemfelder und Erfolgsfaktoren, Sven Röpstroff, Robert Wiechmann, dpunkt.verlag, 2022

Scrum: Produkte zuverlässig und schnell entwickeln, Boris Gloger, Hanser Verlag, 2016

Agiles Produktmanagement mit Scrum: Erfolgreich als Product Owner arbeiten, Roman Pichler, dpunkt.verlag, 2013

Folgende Blogs empfehle ich zu verfolgen:

Mike Cohn's Blog, https://www.mountaingoatsoftware.com/blog

Scrum Alliance, Resources : https://www.scrumalliance.org/agile-resources

Roman Pichler's Blog: http://www.romanpichler.com/blog/

Scrum.org Blog: https://www.scrum.org/resources/blog

scrum inc. Blog: https://www.scruminc.com/scrum-blog/

Innolution Blog (Kenneth S. Rubin): https://innolution.com/blog

LeadingAgile Blog: https://www.leadingagile.com/blog/

DZone Agile Zone Articles: https://dzone.com/agile-methodology-training-tools-news

AgileConnection Blog : https://www.agileconnection.com/

Über den Autor

Roland Wanner ist schon seit über 30 Jahren im Projektgeschäft tätig und hat schon viele Projekte miterlebt – erfolgreiche und gescheiterte. Nach seiner Ausbildung als Maschineningenieur und Wirtschaftsingenieur war er zuerst 5 Jahre als Projektleiter und dann mehrere Jahre als Projektcontroller und Projektportfolio-Manager im Maschinen- und Anlagenbau tätig. Seit mehr als 10 Jahren arbeitet er als Projektmanagement-Spezialist, Projektportfolio-Manager und Project Office Manager im Banken- und Versicherungsbereich. Dort unterstützte er verschiedene Softwareprojekte, die Scrum und agiles Projektmanagement angewendet haben.

Auf seinem Blog http://www.rolandwanner.ch/blog finden Sie interessante Artikel über agiles Projektmanagement, Projektcontrolling, Earned Value Management und Risikomanagement in Projekten.

Ihre Meinung ist uns wichtig!

Herzlichen Dank, dass Sie dieses Buch gekauft haben. Wir haben unser Bestes gegeben, beim Inhalt wie auch bei der Aufmachung. Es wurde viel Aufwand geleistet, um dieses Buch so vollständig und korrekt wie möglich zu machen. Es ist jedoch nicht ganz auszuschließen, dass uns an der einen oder anderen Stelle des Buches ein Missgeschick unterlaufen ist, ob inhaltlich oder in der Rechtschreibung. Vielleicht vermissen Sie auch bestimmte Informationen oder sind der Meinung, gewisse Themen sollten vertieft werden, oder sind bei gewissen Themen anderer Meinung. Wir sind auf Ihre Meinung angewiesen!

Für Ihre Ideen, Gedanken und Korrekturvorschläge bedanken wir uns ganz herzlich. Senden Sie diese bitte an: info@rolandwanner.ch

Literaturverzeichnis

Amabile, T. (2011). *The Progress Principle: Using Small Wins to Ignite Joy, Engagement, and Creativity at Work.* Harvard Business Review Press.

Cohn, M. (2012). *Release Planning: Retiring the Term but not the Technique*. Von https://www.mountaingoatsoftware.com/blog/release-planning-retiring-term-not-technique abgerufen

Cohn, M. (2014). *Agile Estimating and Planning.* Prentice Hall.

Cohn, M. (2016). *Don't Estimate the Sprint Backlog Using Task Points*. Von https://www.mountaingoatsoftware.com/blog/dont-estimate-the-sprint-backlog-using-task-points abgerufen

Cohn, M. (2017). *Planning Poker Cards: Effective Agile Planning and Estimation*. Von https://www.mountaingoatsoftware.com/tools/planning-poker abgerufen

Cohn, M. (2017). *When Should We Estimate the Product Backlog*. Von https://www.mountaingoatsoftware.com/blog/when-should-we-estimate-the-product-backlog abgerufen

Dräther, R. (2013). *Scrum kurz & gut.* O'Reilly.

Gloger, B. (2016). *Scrum - Produkte zuverlässig und schnell entwickeln.* Hanser.

Gloger, B. (2017). *Was ist Scrum*. Von https://borisgloger.com/scrum/ abgerufen

Greaves, K. (2012). *Release Planning with Scrum*. Von https://www.growingagile.co.za/2012/10/release-planning-with-scrum/ abgerufen

Joas, H. (1999). *Die Entstehung der Werte.* Suhrkamp.

Kerth, N. (2017). *The Retrospective Prime Directive*. Von http://retrospectives.com/pages/retroPrimeDirective.html abgerufen

Larsen, D. (2004). *Team Agility: Exploring Self-Organizing Software Development Teams.* Von http://www.futureworksconsulting.com/resources/TeamAgilityAgileTimesFeb04.pdf abgerufen

McChrystal, G. S. (2015). In S. McChrystal, *Team of Teams: New Rules of Engagement for a Complex World.* Penguin.

Pichler, R. (2011). *The Product Vision Board*. Von https://www.romanpichler.com/blog/the-product-vision-board/ abgerufen

Pichler, R. (2013). *Scrum - Agiles Projektmanagement erfolgreich einsetzen.* dpunkt.Verlag.

Schwaber, K. (2004). *Agile Project Management with Scrum.* Microsoft Press.

Scrum.de. (2015). *Scrum.de*. Von https://www.scrum.de/wir-brauchen-devops/ abgerufen

Wintersteiger, A. (2012). *Scrum Schnelleinstieg.* entwickler.press.

Stichwortverzeichnis

3

3C Kriterien 115

A

Accountabilities 54
Adaptive Software Development 31
Agil oder Lean 33
Agile Frameworks 244
Agile Manifest 12, **25**
Agile Manifests, Bedeutung 27
Agile Methoden 20, **31**
Agile Prinzipien **28**
Agile Prinzipien und Praktiken 34
Agilen Methoden, Unterschiede 32
Agiles Projektmanagement 15
Agiles Projektmanagement, Strukur **34**
Agilität im Projektmanagement 27
Agilität und Menschen 29
Agilität und Wissen 29
Agilität, was ist es 32
Aktivitäten definieren 174
Anforderung rangieren 109
Anforderungen 19, 45, 106
Anforderungen erfassen 108
Anforderungen verstehen 109
Anforderungen, detaillierte 170
Anforderungen, gute 111
Anforderungen, High-Level 103
Anforderungen, nicht funktionale und technische **117**
Anforderungen, nicht funktionale, funktionale 103
Anforderungsbeschreibung 56
Anforderungsmanagement **106**
Anforderungsworkshop 108, 110, 123, 136
Anpassung 42
Anwender (User) **64**
Anwendererzählung 112
Arbeit ausführen 179
Arbeit priorisieren 180
Arbeit, zu wenig/zu viel 192
Arbeitszyklen 45
Architektur und Design 173
Artefakte, weitere 47
Auftraggeber **64**
Aufwand schätzen 126, 174
Aufwandschätzung **145**
ausführungsverantwortlich 54
Autoindustrie 10
Automatisierung 143

B

Backlog Refinement 89, 147
Barry Boehm 18
Burndown Chart 194, **195**
Burnup Charts 198
Business Case 49, 134
Business Value **120**

C

Chief Product Owner **254**
Coach 61
CPO *Siehe* Chief Product Owner

D

Daily Scrum 80, **187**
Daily Scrum durchführen 188
Daily Scrum vorbereiten 187
Daily Scrum, Fragen 188
Daily Scrum, verteilte Teams 190
DEEP 125
Definition of Done **165**
Definition of Ready **163**
Definitionsphase, trad. Projekt 106

Development and Operations 141
Development-Team **58**
DevOps 141, 142
Die Rollen in Scrum 46
Distributed Teams 242
Done 165

E

Empirie 41
empirischer Prozesssteuerung 41
Entwicklungsgeschwindigkeit 139, 201
Entwicklungsgeschwindigkeit bestimmen 156
Epics **118**
ergebinsverantwortlich 54
Estimation Meeting 124, 126, **147**
Estimation Meeting, Teilnehmer 149
Events **75**
Executive Action Team (EAT) 251
Executive MetaScrum 257
Explorationssprints 136
externe Projekte 134
Extreme Programming 31
Extreme Programming (XP) 18

F

Feature Driven Development 31
Feedback 208
Feiern 210
Fokus (Focus) **71**
Fortschritt überwachen **194**
Fortschrittsüberwachung 182, 184
Führungssysteme 15
funktionsübergreifend 54
Funktionsübergreifend arbeiten 231
funktionsübergreifende Teams 227

G

große Projekte 241
Großunternehmen 15
Grundursachen, finden 218

H

Herausforderungen, große 225
Hindernisse 190

I

Impediment Backlog 220
Impediments 190
Initialisierungsphase 136
Inspect & Adapt 46, 80, 212
Interdisziplinäre Teams 54
INVEST, Kriterien 111
Iterationen 45

J

japanischen Produktentwicklungsmethoden 23
Jeff Sutherland 10, 25, 244

K

Karteikarten 114
Ken Schwaber 10, 25, 244
klassische Projekte **19**
Kleinunternehmen 15
Knowledge Management 24
Kompensationssysteme 16, 224, 238
Konflikte im Team 237
kontinuierliche Verbesserung 211
Kontinuierliche Verbesserung 232
kontinuierlichen Verbesserungsprozess 212
Kunde **64**
Kundenanforderungen verstehen 109
Kundenbedürfnisse 103

L

Large Scale Scrum 244
Lean Development 33
Lean Innovation 33
Lean Management 24

leichtgewichtig 18
Leistungsbewertung 238
LeSS 244

M

Macht 16, 224
Magic Estimation 153
Manager **65**
Marktbedürfnisse 20
Märkte 103
Marktsegment 103
Mentor 61
Mentor und Coach 236
Minimum Marketable Feature Set 109, **120**
Minimum Viable Product **121**
Mittelfristplanung 138, 140
MoSCoW 131
Mut (Courage) **71**

N

Net Present Value 120
New New Product Development Game 37

O

objektorientierte Programmierung 17
Offenheit (Openness) **72**
Ownership übernehmen 233

P

Parking Lot Diagram 200
Pflichten- und Lastenheft 134
Pflichtlieferobjekte 49
Planning Poker 152
Planung 138
Planung in Scrum **133**
Pre-Sprint Story Review 167
Priorisierung nach MoSCoW 131
Product Backlog 88, **123**
Product Backlog bei großen Projekten 260
Product Backlog detaillieren 89
Product Backlog Management 56
Product Backlog priorisieren 130
Product Backlog, detaillieren 126
Product Backlog, Übersicht 124
Product Increment 91
Product Owner **55**
Product Owner Cycle 253
Product Owner, Zusammenbarbeit 235
Product Vision Board 100
Product Vision, Team Identifikation 105
Product-Development-Teams 23
Produktattribute 103
Produktentwicklungsteams 23
Produktlebenszyklen 17
Produktvision 55, **99**
Produktvision teilen, kommunizieren 104
Projektbudget 134
Projekt-Governance 49
Projektorganisation, große Projekte 259
Projektplanung 133
Projektstart 134
Projekt-Startaktivitäten 134
Proof of Concept 137

R

Rapid Application Development (RAD) 18
Rational Unified Process (RUP) 18
Ready Stories 170
Release-Management 140
Releaseplanung **138**
Releaseplanung bei großen Projekten 261
Remote Teams 242
Respekt **73**
Responsible vs. Accountable 54
Retrospektive durchführen 215
Retrospektive, Ablauf 214

Retrospektive, Maßnahmen 218
Retrospektive, Maßnahmen priorisieren 219
Retrospektive, vorbereiten 213
Return of Investment 120
Return on Investment (ROI) 66
Risiken 155
Risiko der Anforderungen 128
Risikoanalyse 134
Rugby 24
Rugby-Ansatz 37

S

SAFe® 227, 244
ScALed 227, 244
Scaled Agile Framework 244
Scaled Agile Lean Development 244
Scaled Daily Scrum 247, 248
Schätzen mit Punkten 151
Scope Änderungen 191
Scrum als Management Framework 40
Scrum Artefakte 47
Scrum Artefakte – Übersicht 85
Scrum bei großen Projekten 241
Scrum But 47
Scrum Ereignisse 46
Scrum Framework **37**
Scrum Guide 12, **38**
Scrum im Schnellüberblick 44
Scrum Master **59**
Scrum Master Cycle 247
Scrum Master Cycle, Ergebnisse 251
Scrum Master, Aufgaben 59
Scrum Master, Eigenschaften 61
Scrum Master, Entscheidungen 62
Scrum Master, Zusammenarbeit 235
Scrum of Scrums 247
Scrum of Scrums Master (SoSM) 249
Scrum Team, Umfeld 65
Scrum Workflow **94**
Scrum, die drei Säulen 41
Scrum, Entstehung 37
Scrum@Scale 245
Scrum@Scale Guide 245
Scrums Master (SoSM) 248
Scrum-Werte **67**, **69**
SDLC 17
selbstmanagt 54
selbstorganisierende Teams 16
Selbstorganisierende Teams **227**
Selbstorganisierende und interdisziplinäre Teams 54
Set-Based Concurrent Engineering 33
Skalierung 250
Softwareentwicklungsmethoden 17
SoS skalieren 250
Spiralmodel 18
Sprint 77
Sprint 0 136
Sprint abbrechen 193
Sprint Burndown Chart 195
Sprint Planning 78, **159**
Sprint Planning, Vorbereitung 167
Sprint Retrospektive **211**
Sprint Review **205**
Sprint Review Teilnehmer 206
Sprint Review, Ablauf 207
Sprint Review, Maßnahmen daraus 209
Sprint Review, Resultat 209
Sprint-Ablauf 159
Sprintdauer, optimale 155
Sprint-Durchführung **177**
Sprint-Review, Vorbereitung 207
Sprint-Reviews, Projektweite 262
Sprintstart, Wochentag 179
Sprint-Umfang anpassen 191
Sprintziel 169, 205
Stakeholder 53, 208
Startaktivitäten 134
stetige Lernen 211
Story Burndown Chart 199
Systems Development Life Cycle 17

T

Taskboard **183**, 185, 194, 195
Taskboard, Informationszentrum 184
Task-Planung 180
Tasks **119**
Team, fest zugeordnetes 226
Teambildungsprozess, Tuckman 239
Teamgröße 156
Teamgröße, optimale 227
Teamkapazität überprüfen 175
Teamleistung und Belohnung 238
Team-Philosophie 224
Teams, mehrere 62
The New New Product Development Game 23, 24, 225
Theme **118**
Themenpark 200
Timeboxing **76**
Timeline-Analyse 216
Time-to-Market 17
Toyota Production System 10, 23
traditionelle Projektmanagement 21
traditionellen Softwareentwicklung 21
Transparenz 42, 71, 72

U

Überprüfung 42
Unified Process 31
Unternehmens-Governance 49
Unternehmenskultur 15
User Stories, kleine 116

V

Velocity 156
Velocity Chart 201
Verpflichtung (Commitment) **70**
verteilte Teams 241
Vorprojekt 136

W

Wasserfallmodell 17, 21
Wasserfall-Modell 19
Wertesystem 27, 68

Z

Zusammenarbeit im Scrum-Team **223**
Zusammenarbeit in Teams 23
Zusammenarbeit mit Auftraggeber 239

Kurze Entwicklungszyklen, flexibler auf Marktveränderungen reagieren, schneller einsetzbare Produkte entwickeln

Wollen Sie mit innovativen Produkten schneller auf Marktveränderungen reagieren, Ihre Time-to-Market radikal verkürzen, einen schnelleren Return on Investment und ihren Kunden mehr Flexibilität bieten? Agiles Projekt Management und Scrum bieten nicht nur das, sondern auch interessantere und zufriedenstellendere Arbeit für Ihr Projektteam und eine viel produktivere Zusammenarbeit. Teams organisieren ihre Arbeit selbst und übernehmen mehr Verantwortung. Mit Scrum steigern Sie nachhaltig die Kundenzufriedenheit, die Wertschöpfung aber auch die Zufriedenheit Ihrer Projektmitarbeiter

In diesem Buch lernen Sie das Wichtigste zum agilen Projektmanagement und Scrum, um dieses erfolgreich anzuwenden. Es beschreibt Scrum umfassend, systematisch und leicht verständlich. Sie lernen folgende Themen:

- Agiles Projektmanagement
- Das Scrum Framework
- Die Rollen in Scrum
- Agile- und Scrum-Werte
- Scrum-Ereignisse und Artefakte
- Anforderungsmanagement in Scrum
- Zusammenarbeiten im selbstorganisierenden Scrum-Team
- Scrum bei großen und verteilten Projekten

Mit dem umfassenden Glossar mit Definitionen aller Schlüsselbegriffe eignet sich dieses Buch gleichermaßen als umfassende Einführung und als Nachschlagewerk für die tägliche Arbeit.

Dies ist ein Buch für Einsteiger in Scrum und agiles Projektmanagement, aber auch für fortgeschrittene Leser, die sich z.B. auf eine Scrum-Zertifizierung vorbereiten.

Dieses Buch basiert auf dem aktuellen Scrum Guide vom November 2020 von Ken Schwaber und Jeff Shuterland