Getting Ready for Success

SCRUM

The Essential Guide

How to Successfully Apply Agile Project Management and Scrum

- Decreased Time to Market
- Faster Return of Investment
- More Customer Flexibility
- More Satisfying Work
- Increased Collaboration and Ownership
- Reduced Project Risk

ROLAND WANNER

SCRUM

How to Successfully Apply Agile Project Management and Scrum

ROLAND WANNER

Author Contact:
Roland Wanner
E-Mail: info@rolandwanner.com
Internet: www.rolandwanner.com

Proconis Publishing

Disclaimer:
This book contains information about Scrum and Agile Project Management. It was written for information and training purposes. Therefore, this text should only be used as a general guide and not as the sole source of information about Scrum and Agile Project Management. For professional use, the support of a competent specialist is recommended.
Despite great care to make this book as complete and correct as possible, it cannot be excluded that it contains typographical or content errors.

The author, publisher and the quoted sources are not liable for any losses that may result from the direct or indirect implementation of the descriptions and formulas used in this book.
Subjects include: Scrum, Agile Project Management, IT and Software Projects, Sprint, Timeboxing, Product Owner, Scrum Master, Daily Scrum and others.

For questions or suggestions please contact:
info@rolandwanner.com

Please note that software and hardware names used in the book as well as brand names and product names, e.g. The Scrum Guide™ or Scrum@Scale™ of the respective companies are generally subject to trademark, brand or patent protection.

First Edition
ISBN: 978-1983653995
V1.0 March 2019

Table of Contents

Preface 9

1 **Agile Project Management 13**

Agile Project Management at a Glance 14

The Difference Between Traditional and Agile Projects 17

Agile Project Management for all Projects? 19

What Makes Teams Successful in Product Development? 21

The Agile Manifesto 23

The Agile Principles 26

Agile Methods 28

The Structure of Agile Project Management 31

2 **The Scrum Framework 33**

How Scrum Evolved 33

Scrum as a Management Framework 36

The Three Pillars of Scrum 37

Scrum at a Glance 39

Scrum and Agile Projects in Corporate Governance 44

3 **The Roles in Scrum 45**

The Product Owner 47

The Development Team 51

The Scrum Master 52

The Tasks of the Scrum Master 52

Personal Attributes and Further Functions of the Scrum Master 55

Further Roles 57

The Scrum Team and its Environment 58

4 **Scrum Values 61**

The Scrum Values as the Foundation 62

Commitment 63

Courage 65

Focus 65
Openness 66
Respect 67

5 Scrum Events – An Overview 69
The Sprint 70
Sprint Planning 71
The Daily Scrum 72
The Sprint Review 74
Sprint Retrospective 75

6 Scrum Artifacts – An Overview 77
The Product Backlog 79
The Sprint Backlog 80
The Product Increment 81
Transparency of Artifacts 82
The "Definition of Done" (DoD) 84
The Scrum Workflow 84

7 Requirements Management 87
From the Product Vision to the Product Backlog 87
The Product Vision 88
The Product Vision Board 90
Requirements Management in Scrum 94
The Requirements Workshop 96
The Characteristics of Good Requirements 99
User Stories 100
Determine Non-Functional Requirements 103
The Difference Between Epic, Theme, User Story and Task 104
The Business Value 105

8 The Product Backlog 107
Overview of the Product Backlog 108
Detailing the Product Backlog 111
Estimating the Effort 111
Managing the Risks 112
Prioritizing the Product Backlog 114

Prioritization According to MoSCoW....115
Prioritizing is a Recurring Optimization Task....116

9 **Planning in Scrum....117**
When Does the Project Start?....118
Before the Project Starts....120
Release Planning....122
DevOps – Development and Operations....126
Estimating the Effort....129
Effort Estimation on Two Levels....129
The Estimation Meeting....130
How to Estimate with Points....134
Planning Poker....135
Magic Estimation....136
User Stories Should be "Small"....138
The Optimal Sprint Duration....139
Determining the Development Velocity....140

10 **Sprint Planning....143**
The Sprint Process....143
The "Definition of Ready"....147
The "Definition of Done" (DoD)....148
Preparation for the Sprint Planning....150
Sprint Planning 1....151
Sprint Planning 2....155

11 **Sprint Execution....159**
Artifacts and Events in the Sprint Process....159
Executing the Work....161
The Sprint Backlog....163
The Scrum Taskboard....164

12 **The Daily Scrum....167**
Preparing and Conducting the Daily Scrum....167
Customizing the Sprint Scope....170
Canceling a Sprint....172
Monitoring Progress....173

The Taskboard 174
The Sprint Burndown Chart 174
The Release Burnup Chart 177
The Story Burndown Chart 178
The Parking Lot Diagram 178
The Velocity Chart 179

13 **The Sprint Review 181**
The Goal of the Sprint Review 181
Preparing and Conducting the Sprint Review 182
The Result of the Sprint Review 185

14 **The Sprint Retrospective 187**
Continuous Improvement and Learning 187
Preparing the Retrospective 189
Conducting the Retrospective 190
Create Safety 190
Visualize Events 191
Gather Insights 192
Define Actions 193
Prioritize Actions 194

15 **Collaboration in the Scrum Team 197**
The New Team Philosophy 198
Great Challenges 199
Dedicated Team 200
Limited Team Size 200
Self-Organizing Teams 201
Working Cross-Functionally 204
Continuous Improvement and Learning 205
Taking Ownership 206
Collaboration within the Team 207
Collaboration with the Product Owner 207
Collaboration with the Scrum Master 208
Collaboration with the Customer 211

16 **Scrum for Large Projects 213**
Large Projects and Distributed Teams 213

The Agile Blueprint for Large Projects 216
The Scrum@Scale™ Guide 217
Scrum@Scale Overview 217
The Scrum Master Cycle 219
The Product Owner Cycle 225
The Organization of Large and Distributed Projects 231
Adapting Practices for Large Projects 232
The Product Backlog for Large Projects 232
Release Planning for Large Projects 233
Determining the Effort 233
Project-Wide Sprint Review and Retrospective 234

17 Glossary 237

18 Appendix 247

Internet Links and Recommended Literature 247
About the Author 249
Bibliography 250
Index 252

This book is based on the **Scrum Guide™** from November 2017 by Ken Schwaber and Jeff Sutherland and on the **Scrum@Scale™ Guide** by Jeff Sutherland of Scrum Inc. from August 2018

Preface

Congratulations on wanting to know more about Agile Project Management and Scrum! This is the ideal book to help you to learn everything important, very quickly, and then work immediately with Scrum or in Scrum projects. This book is a practical guide that will guide you through all stages of a Scrum project.

Scrum and Agile have been one of the most important management topics for several years now, and Agile Project Management will become increasingly important in the future, not only for software development projects. With the knowledge in this book, you will be well-equipped for an interesting future.

At the beginning of this book, you get a short and general introduction to Agile Project Management. This then lays the foundation for the main part of the book, which is Agile Project Management with Scrum for software development projects.

Faster, Cheaper, Better and More Business Value

Agile methods are gaining ground, not only in project management, but also in many other management areas too. There is a sustained effort to derive benefits from streamlining processes, focusing even more on customers and customer benefits, reacting even faster to market changes and "experimenting" more with self-organizing teams. This also with the goal of making work more meaningful, interesting and rewarding. If you are a little older, you may be familiar with these topics, some of which were already in use in the 1960s.

Successful Management Methods Rediscovered

I've been in the business for more than 30 years. The principles, methods and values on which agile project management is based on have existed for decades, they came and most of them disappeared after some time. Almost all of them come from the ideas developed by the Toyota Production System and other Japanese companies in the 1950s

and 1960s. This resulted in the following years in management methods such as Lean Production, Lean Management, Business Process Reengineering and Six Sigma or Knowledge Management. Self-Directed Work Teams (SDWTs) are also old news. These were also successfully used in Japanese and some American companies many decades ago.

All these management methods were very successful in these companies, but surprisingly they have never really established themselves broadly in the economy and have almost been forgotten. Only the automotive industry has learned from and benefited greatly from the Toyota Production System and Lean Production.

The work in projects in industry, construction and other branches has been carried out for centuries sequentially in phases that build on each other. When information technology and software development emerged in the 1960s, sequential procedures were also adopted in these projects. In the following years, various more or less successful iterative software development methods were developed.

In 1995, Jeff Sutherland and Ken Schwaber together presented a document describing the Scrum framework for software projects. They developed, based on the very successful but almost forgotten values, principles and methods that you have been briefly introduced before, the most successful project management method for software development - since software was developed—and this is SCRUM.

Who Should Read This Book?

This is a book for all who are interested in Agile Project Management and would like to know, how the best-known agile method in software development, Scrum, works and how to apply it successfully.

This is not a book with all the details and extensive examples. For this, there are more comprehensive and much more expensive books. Here you will learn the most important things about Agile Project Management and Scrum.

Whether you already work in software development, you are a student, project client for software or you already work on an agile project—this book provides you with the necessary knowledge to better understand

Agile Project Management and Scrum and apply them successfully. This is a book for you!

How This Book is Organized

First, you will learn how Agile Project Management has emerged and how it differs from "normal" project management. Then, you will read what the Agile Manifesto is and what the agile principles mean to Scrum and the other agile methods.

The main part of this book deals with Scrum. First, you will get **a quick overview of the Scrum Framework** starting on Page 33. Within five minutes you will get a basic functional understanding to make connections much better later on.

The following chapters give you an **overview** of the Scrum Values, Scrum Events and Artifact. The other chapters deepen what has been learned and go into detail about the Scrum artifacts and events.

As you read this book, you will probably notice that many topics occur several times in different granularity. This is not a mistake or just to fill the book with pages, but rather it serves to deepen yourself in Scrum and to get to know more and more details on the different related topics.

An Ideal Reference Guide

If you have read this book, it is also an ideal reference guide for you later on. At the end of the book, you will find an extensive glossary and a helpful index with which you can quickly find a specific topic in the book.

Last but not least, this book is excellently suited for training courses and studies in the field of software development.

Der Scrum Guide

The Scrum Guide™ by Ken Schwaber and Jeff Sutherland is the official guideline for Scrum. It is updated periodically. This book is based on the latest version of the Scrum Guide from November 2017 and the Scrum@Scale Guide from August 2018.

The Scrum Guide briefly describes (without illustrations) the minimum requirements for a Scrum project. It can be downloaded here for free:

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

This book gives you a comprehensive insight into Agile Project Management and Scrum with many helpful illustrations and goes far beyond the Scrum-Guide.

The Agile Manifesto

The Agile Manifesto for software development is the lowest common denominator of all agile process models—and thus also for Scrum. It describes the "Twelve Principles of Agile Software Development", which you can read here:

http://agilemanifesto.org/

1

Agile Project Management

Agile methodologies are used in more and more projects—in software product development successful for many years, with other types of projects we are still at the beginning. However, the trend clearly shows that more and more companies are working with agile approaches or are already in the process of introducing them. Agile projects are already being successfully implemented in some traditional industries such as automotive and aircraft construction. Examples are Toyota, BMW and SAAB Technologies.

Recently, agile methodologies have also been used in business processes other than projects. This is an exciting challenge that will strongly influence the entire corporate culture, management systems and collaboration. However, I am not so sure whether many large companies will make this step in the next few years. Small businesses are much more adaptable in this respect.

Learning and understanding agile methodologies like Scrum is relatively easy. However, it is much more difficult to internalize and live

the agile values and basic principles—and many companies still have and will have trouble here.

In the end, however, the success of agile methodologies at company level can only be achieved through radical changes within organizations, and here we are still a long way from making great progress. At the project level, however, we are already well advanced in this area.

Agile methodologies have a radical influence on management and compensation systems in companies. Managers also lose power and influence in self-directed work teams. This will make the change, especially in the corporate culture, not easy.

The principle of self-directed and cross-functional teams, bosses as coaches without management functions and the reduction of management levels, was already an "interesting topic" decades ago. I hope that we will be more successful with this promising topic in the coming years.

With Scrum, agile methodologies having already been successfully applied in software development for several years, an important first step has already been taken!

Agile Project Management at a Glance

Impressive projects were carried out thousands of years ago. Think of the stone structure of Stonehenge (3500 years B.C). and the Egyptian pyramids (built in 2500 B.C.) or in more recent times the medieval castles, fortresses and cathedrals, the steam engine, cars, the atom bomb and the skyscrapers. Some of these were huge and complex projects for their time. However, software and software projects have only been around for a few decades. In the 1950s, software was still part of the hardware and was designated as program code. I also remember the punched cards used to control machine tools in the 1970s. The punch cards were the programs for milling parts with machine tools.

There were no software development methods until the 1960s. The Systems Development Life Cycle (SDLC) was the first to be developed during this period with the aim of developing large, functional business systems. Until the 1990s, all projects were carried out according to the sequential waterfall model (Figure page 17). This means that all requirements were defined at the beginning of the project, then concepts, specifications and plans were created and then the product was built and launched on the market.

Software development in the 1990s was shaped by object-oriented programming, the rise of the internet and the dot.com boom. Time-to-market and company growth were decisive here, i.e., the development cycles for software became shorter and shorter.

The Waterfall Model Was No Longer Suitable

People in software development were less and less satisfied with the rigid waterfall model, especially as projects became more complex, product life cycles shorter and shorter, and the environment and requirements more dynamic. Customers needed faster and faster usable software, not with all functions, but the most important ones. Software development had to be lighter, more flexible and faster and less administration was desired.

New methods should make the software development process more flexible and streamlined—as a countermove to the rather heavyweight and bureaucratic, traditional methods, such as the waterfall model. These requirements triggered an active development of different methods in software development:

- 1986 – Barry Boehm developed the first approaches of an iterative software development process with the risk-oriented spiral model.
- 1991 – Rapid Application Development (RAD) was introduced
- 1995 – Jeff Sutherland and Ken Schwaber presented a document describing the Scrum method for the first time
- 1998 – Rational Unified Process (RUP) was introduced
- 1999 – Extreme Programming (XP) was introduced, which generated great interest among software developers

As you can see, the first approaches to agile software development can already be found in the early 1990s. Agile software development first became popular in 1999, when Kent Beck published the first book on Extreme Programming (XP).

The interest in extreme programming also paved the way for other agile processes and methods. The term "agile" for this type of software development was selected by 17 representatives of various software development methods in February 2001, at a meeting in Utah (USA). This as a replacement for “lightweight” used up to then. The Agile Manifesto (see page 23) was also formulated at this meeting. Over the years, the term "agile project management" has developed from this, because not only software projects can be planned, executed and controlled with agile methods, but also other types of projects.

The goal of agile software development is to make the development process more flexible and streamlined than conventional process models. Agile software development is characterized by self-directed work teams, as well as an iterative and incremental approach. The aim is to get by with less bureaucracy and fewer rules, and in this way, adapt quickly to changes without increasing the risk of errors.

The Difference Between Traditional and Agile Projects

In **Traditional Projects**, clear goals and requirements are defined by the internal or external customer (client) at the beginning of the project, which do not change during the project implementation if possible. At the end of the project, the customer receives the project result.

The project will be carried out strictly in successive phases. A previous phase must be completed before the next phase can be started. The project result is created in the course of the phases until it is completed at the end of the last phase. This process is called the waterfall model.

The further the project progresses, the less the customer can influence the final result. A major limitation of the waterfall model is that any change or new requirements that the customer wants to have implemented in a later project phase costs several times as much as if it had been defined at the beginning.

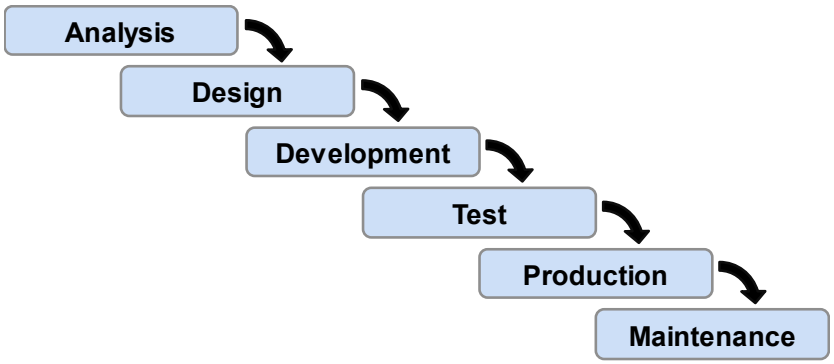


Figure 1: The Waterfall Model

You can probably imagine how to build a house. If in this project the construction workers are already building the walls and you want one more room at this time, then this will be very expensive, or it will be almost impossible.

Agile Methods, e.g., Scrum, are used in development projects (especially software) to adapt to the high dynamics of goals, requirements, environment and expectations. Main characteristics are:

- Close collaboration between customers, the Product Owner and the self-directed work team
- Short development cycles, after which changes and new requirements can be added to the planning (iterations with a defined length; typically 14 to a maximum of 30 days)

Agile project management makes it possible for the customer or the stakeholders to introduce new requirements during the entire project life-cycle or to change existing ones and thus to respond flexibly to short-term market needs—and this without causing exponential cost growth. Of course, the overall budget must still be kept in mind.

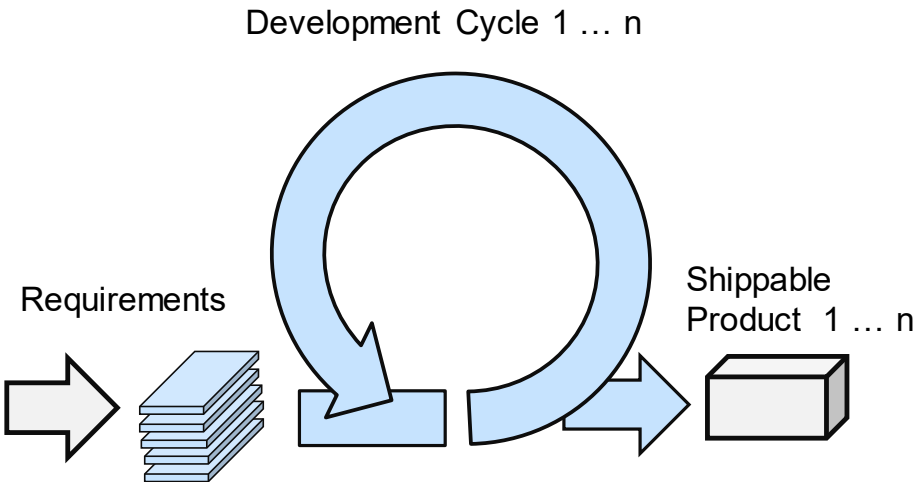


Figure 2: The Agile Process

Agile Project Management for all Projects?

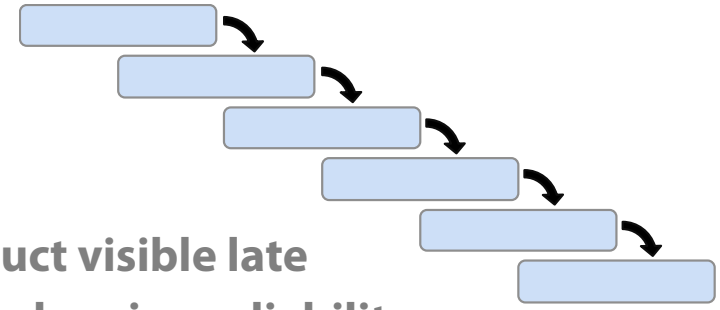
Agile Project Management is gaining more and more acceptance, even outside traditional software development. Many predict the death of the waterfall model. It won't come to that. Try to develop and manufacture a house, aircraft carrier, machine tool or production plant with Agile Project Management. Impossible! In such cases, practically all requirements must be known at the beginning of the project, and the flexibility to change or add new requirements during the course of the project is very small.

In the development of *physical products*, the waterfall model will remain the preferred project management process; however, for "intangible products" such as software, agile project management will gain more and more acceptance.

Nevertheless, this does not mean that agile methods cannot be used for certain deliverables supplied by “waterfall projects”. Wherever software needs to be developed, agile development methods can probably be used, for example for the software of a machine tool or a car entertainment system.

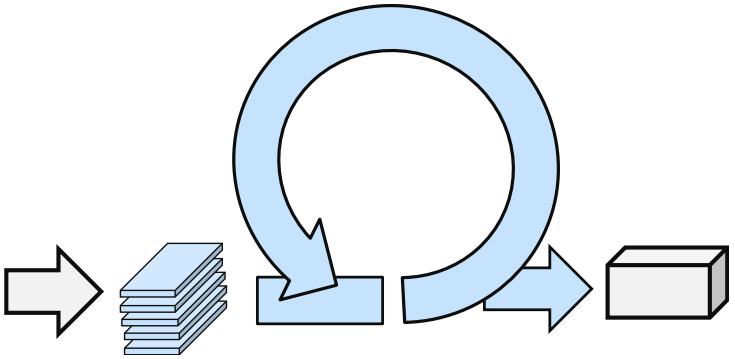
Certain principles, practices and values of agile project management will certainly have a strong influence on traditional project management in the future. I'm just thinking about the teamwork and the agile values and principles. But wherever physical products are developed, the flexibility in the development process is clearly smaller than in software development.

Waterfall Model



- rigid
- product visible late
- high planning reliability
- consistent requirements necessary

Agile Process



- little structure, iterative
- results can be used early on
- high flexibility, close to the market
- changing requirements welcome

What Makes Teams Successful in Product Development?

A key aspect of agile project management is teamwork and the application of knowledge. To learn more about this, let's make a short excursion into the past.

Hirotaka Takeuchi and Ikujiro Nonaka described the characteristics of successful product Development Teams in the 1986 *Harvard Business Review* article "The New New Product Development Game". The content of the article is strongly influenced by the "Toyota Production System" and Japanese product development methods, e.g., at Canon, Honda or NEC in the 1970s. This article[1] describes the key features of successful product development teams:

1. **Built-in instability:** The management initiates the development process, sets challenging goals and requirements and gives the team the greatest possible freedom in development.
2. **Self-organizing project teams**: The team works like a start-up company, it is proactive, takes risks and has an independent agenda: The teams are characterized by the following three characteristics:
   - *Autonomy*: The teams are self-directed.
   - *Self-transcendence:* Continuous learning. The team strives forwards and always wants to improve.
   - *Cross-fertilization:* The teams work across functions. This diversity promotes new ideas and concepts, leading to more successful products.
3. **Overlapping development phases**: With strongly overlapping phases, the development process becomes faster and more flexible.
4. **Multilearning:** Through learning at different company levels and in groups, through constant exchange of experience and

[1] https://hbr.org/1986/01/the-new-new-product-development-game

through contact with the environment, one can react more quickly to changing market conditions.

5. **Subtle control:** The management sets enough checkpoints, although the project team organizes itself. The aim is to avoid instability, ambiguities and tensions.
6. **Organizational transfer of learning:** Learned knowledge should flow into the organization in order to improve constantly.

Outside Japan, few industries have incorporated elements of this concept into their operations, and if so, often rather half-heartedly. Derived from this knowledge, Lean Management and Knowledge Management emerged in the 1990s.

The low success rate of software projects in the 1990s was not due to the training of employees or the lack of maturity, the still young computer science and its tools, but to the implementation and collaboration in projects. Ken Schwaber and Jeff Sutherland recognized this and then developed a successful software development framework from the knowledge of "The New New Product Development Game" and their own experiences—SCRUM.

What Does Scrum Have to Do with Rugby?

You may have wondered what Scrum has to do with rugby. The cover of this book shows a scrum at a rugby match, which is the crowd of players when the ball is thrown in for minor rule violations.

Jeff Sutherland and Ken Schwaber were inspired by the business review article "The New New Product Development Game", where rugby and the connection to successful, self-organizing teams were mentioned several times. That's why Jeff and Ken used a rugby term for their new software development framework - and that's SCRUM.

The Agile Manifesto

In spring 2001, 17 experienced software developers in Utah (USA) passed the so-called "Manifesto for Agile Software Development", today mainly known as "Agile Manifesto". These first signatories, among them the two Scrum founders Ken Schwaber and Jeff Sutherland, formed the central values of agile software development with the Agile Manifesto—a milestone and at the same time the foundation of agile project management.

The values of the Agile Manifesto (www.agilemanifesto.org) are described in pairs, whereby the values on the left side are considered higher than the values on the right side. This does not mean, however, that these are meaningless.

The Agile Manifesto reads as follows:

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

"That is, while there is value in the items on the right, we value the items on the left more".

I would rather call the agile values of the Agile Manifesto "principles of agile software development", so that they are not confused with the Scrum values.

Individuals and Interactions

Working Software

Customer Collaboration

Responding to Change

The Importance of the Agile Manifesto

The Agile Manifesto has lost none of its importance for agile project management. Over the years, thousands more supporters and signatories have been added. At that time, the authors succeeded in bringing the core ideas of modern software development to a common denominator once and for all, despite sometimes very different views and approaches.

The great progress was and is that with the agile manifesto a system of values has finally been set in stone that outlines a concrete approach. At the same time, the comparatively fuzzy concept of "lightweight software development", which had been common up to then, could be replaced. In this respect, the Agile Manifesto can best be understood as the "state of mind of agility", and this state of mind still lives on today in agile methods such as Scrum or Kanban or their rules and roles.

The Agile Manifesto also meant that from then on, employees were not simply regarded as resources, but were the focus of attention.

Agility in project management demands and promotes the individual skills of employees by giving them more responsibility and creative opportunities. This paves the way for more effective and successful projects.

17 Appendix

Internet Links and Recommended Literature

On the website https://www.rolandwanner.com, you will find a list with links and articles on agile project management, project controlling, earned value management and risk management within projects.

On my blog https://www.rolandwanner.com/blog you will find interesting articles on agile project management, project controlling, earned value management and risk management in projects.

I Recommend the Following Additional Scrum Books:

Agile Estimating and Planning, Mike Cohn, Prentice Hall, 2005

Agile Product Management with Scrum: Creating Products that Customers Love, Roman Pichler, Addison-Wesley Signature Series, 2010

Scrum: The Art of Doing Twice the Work in Half the Time, Jeff Sutherland, rh Business Books, 2015

I Recommend Following These Blogs:

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

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

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

About the Author

Roland Wanner has been in the project business for over 30 years and has participated in many projects, both successful and failed. After his education as a mechanical and industrial engineer, he first worked for 5 years as a project manager and then for several years as a project controller and project portfolio manager in mechanical and plant engineering. For more than 10 years he has worked as a project management specialist, project portfolio manager and project office manager in the banking and insurance sector. There he supported various software projects which had applied Scrum and Agile Project Management practices

On his blog https://www.rolandwanner.com/blog you will find interesting articles about agile project management, project control, earned value management and risk management within projects.

Your Opinion is Important to Me!

Many thanks for buying this book. We have done our best, both in content and presentation. Much effort has been put into making this book as complete and correct as possible. However, it cannot be completely ruled out that we made a mistake at one point or another in the book, whether in terms of content or spelling. Perhaps we also missed certain information or certain topics could be explained in more detail, or you even disagree with our opinions on certain topics. We depend on your opinion!

We thank you very much for your ideas, thoughts and correction suggestions. Please send them to: info@rolandwanner.com

Bibliography

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

McChrystal, G. S. (2015). *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.

Index

3

3C criteria 100

A

activities, define 154
Adaptation 38
Adaptive Software Development 28
agil or lean 30
Agile Manifesto 12, 23, 34
Agile Manifesto, Importance 25
agile methodologies, differences 29
Agile Methods 18, **28**
agile principles **26**
agile principles and practices 31
Agile Project Management **13**
agile project management, structure **31**
agility and knowledge 27
agility and people 27
agility in project management 25
agility, what is it 29
Architecture and Design 153
Artefact Transparency 76
Artifact Transparency 42
automation 125
automotive industry 10

B

Backlog Refinement Meeting 128
Barry Boehm 15
burnup charts 175
business case 44, 116
business process re-engineering 10
Business Value **103**

C

celebrate 185
challenges 198
Chief Product Owner **224**
coach 55
Collaboration in the Scrum Team 196
Collaboration with Customer 210
Commitment **63**
compensation systems 14, 197, 209
Conflicts in a Team 208
continuous improvement 204
continuous improvement process **187**
corporate culture 13, 14
Corporate Governance 44
Courage **64**
CPO *Siehe* Chief Product Owner
cross-functional working **203**
customer **57**
customer needs 88

D

Daily Scrum 70, **165**
Daily Scrum, distributed teams 168
Daily Scrum, performing 166
Daily Scrum, preparation 165
Daily Scrum, questions 166
debriefings **186**
DEEP, criteria 107
Definition of Done (DoD) **146**
Definition of Done (DOD) 82
Definition of Ready 145
Development and Operations 124
Development Team, capacity 155
Development-Team **51**
DevOps **124**
distributed teams 212
Distributed teams **213**
Done 146

E

effort estimation 154

Effort Estimation 109, **127**
empirical process control 37
empiricism 37
Epic, Theme, User Story, Task **102**
Epics **102**
estimating with points 132
Estimation meeting 109
Estimation Meeting 106, 128
Estimation Meeting, participants 130
Executive Action Team (EAT) 221
Executive MetaScrum. 226
Exploration Sprints 118
Extreme Programming 28

F

Feature Driven Development 28
Focus **64**

I

initialization phase 118
inspect and adapt 41, **68**
Inspect and Adapt **187**
Inspection 38
INVEST criteria 97
iterations 40

J

Japanese product development methods 21
Jeff Sutherland 10, 23, 215

K

Ken Schwaber 10, 23, 215
knowledge management 10
Knowledge Management 22

L

large companies 13
lean development 30
lean innovation 30
Lean Management 22
lean production 10
LeSS 215
lessons learned **186**
lightweight 16

M

Magic Estimation 134
management systems 13
Manager **57**
mandatory deliverables 44
market needs 18
market segments 88
markets 88
medium-term planning 120
mentor 55
Mentor and Coach 207
Minimum Marketable Feature Set **103**
Minimum Viable Product **104**
MoSCoW 113

N

Net Present Value (NPV) 103
New New Product Development Game 33
Non-functional requirements 88, **101**

O

object-oriented programming 15
open-error culture **65**
Openness **65**
ownership **205**

P

parking lot diagram 176
performance evaluation 209
Planning in Scrum **115**
Planning Poker® 133
power 14, 197
preliminary project 118
Pre-Sprint Story Review 148
Prioritizing according to MoSCoW 113
Product attributes 88
Product Backlog 77, **105**
Product Backlog for Large Projects 229
Product Backlog Management 49

Product Backlog, detailing 109
Product Backlog, overview 106
Product Backlog, prioritizing 112
Product Backlog, ranking 95
Product Backlog, refinement 77
product development teams 21
Product Increment 79
product life cycles 15
Product Owner **47**
Product Owner Cycle **223**
Product Owner, collaboration 206
product vision 48
Product Vision **86**
Product Vision Board 88
Product Vision, sharing 90
Product Vision, team identification 91
progress monitoring 161, 171
project budget 116
Project governance 44
project organization, large projects 228
project planning 115
project start 116
project start activities 116
proof of concept 118

R

Rapid Application Development (RAD) 15
Rational Unified Process (RUP) 15
Ready Stories 150
release burnup chart 175
Release Management 49
Release Planning **120**
Release Planning for Large Projects 230
Release-Management 122
Remote teams **213**
requirements 17, 40, **92**
requirements identification 94
requirements management **85**
Requirements Management in Scrum **92**
requirements specification 48
requirements workshop 95, 105
Requirements Workshop 94, 118
requirements, definition phase 92
requirements, detailled 150
requirements, index cards 98
Requirements, Risk 110
requirements, understanding 95
Respekt **66**
Retrospective process 189
Retrospective, actions 192
Retrospective, in five steps 189
Retrospective, prioritizing actions 193
Retrospektive, execution 189
return on investment 58
Return on Investment **103**
risk analysis 116
Risk management **110**
risks 137
Roles in Scrum 41, 45
Ron Jeffries 215
root cause 192
ross-functional teams 200
rugby approach 33

S

SAFe® 215
ScALed 215
Scaled Backlog Refinement Meeting 223
Scaled Daily Scrum **219**
scope changes 169
Scrum Artifacts 42
Scrum Artifacts – An Overview 75
Scrum as a Management Framework 36
Scrum at a Glance 39
Scrum But 43
Scrum Events 41, **67**
Scrum for Large Projects 212
Scrum Guide 12, **34**
Scrum Master **52**
Scrum Master Cycle 218
Scrum Master Cycle, outcomes 222
Scrum Master, Attributes 55
Scrum Master, Decisions 55
Scrum Master, Several Teams 55
Scrum Master, Tasks 52

Scrum of Scrum of Scrums (SoSoS) 220
Scrum of Scrums 218
Scrum of Scrums Master (SoSM) 218
Scrum Taskboard 162
Scrum Team, Environment 58
Scrum Values **61**
Scrum Workflow **82**
Scrum, origin 33
Scrum, three pillars 37
Scrum@Scale 200
Scrum@Scale Guide 216
Scrums Scrum Master **220**
SDLC 15
self-directed work teams 14
Self-Organizing and Interdisciplinary Teams 47
self-organizing teams 200
servant leader 207
Servant Leader 52
Set-Based Concurrent Engineering 30
six sigma 10
software development methods 15
SoS scaling 220
Spiralmodel 15
sponsor **57**
Sprint 68
Sprint 0 118
Sprint Backlog **78**, **161**
Sprint Burndown Chart 172
Sprint canceling 170
sprint duration, optimal 137
sprint duration, risks 137
Sprint execution **157**
sprint goal 181
Sprint goal 149
Sprint Planning 69, **141**, 143
Sprint Planning 1 **149**
Sprint Planning 2 **153**
Sprint Planning, preparation 148
Sprint Process 141
Sprint Retrospective 73, **186**
Sprint Retrospective, preparation **188**
Sprint Review 72, **180**
Sprint Review, actions daraus 184
Sprint Review, execution 182
Sprint Review, feedback 183
Sprint Review, goal 180
Sprint Review, participants 181
Sprint Review, preparation 182
Sprint Review, results 184
Sprint Reviews, project wide 231
Sprint scope, customizing 168
sprint start, weekday 144
stakeholder 86
Stakeholder 45
start activities 116
story burndown chart 176
system of values 25
Systems Development Life Cycle 15

T

task planning 159
Taskboard 171, 172
taskboard, information center 163
Tasks 102
Team Performance and Compensation 209
Team Philosophy 197
team size 138
Team size, optimal 199
Team, dedicated 199
team-building process, Tuckman 210
teamwork 21
The New New Product Development Game 21, 198
Theme **102**
Timeboxing **68**
timeline analysis 190
Toyota Production System 9, 21
traditional project management 19
Traditional Projects **17**
traditional software development 19
transparency 64, **65**, 80
Transparency 37
Transparenz der Artefakte 80

U

Unified Process 28
User **57**
user stories 86
User Stories **98**

User Stories, should be small 136
User stories, specific format 99
User Story 102

V

value system 61
velocity 121, 177
velocity chart 177
velocity, determine 138

W

waterfall model 15, 17, 19
work allocation 159
work cycles 40
work execution 159
work, prioritizing 159
work, too little/too much 169