SCRUM

The Essential Guide

How to Successfully Apply Agile Project Management and Scrum

Scrum Guide 2020

incl. eBook

ROLAND WANNER

SCRUM

The Essential Guide

How to Successfully Apply Agile Project Management and Scrum

ROLAND WANNER

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

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.

For information about buying this book in bulk quantities, or for special sales opportunities (which may include electronic versions, custom cover design, and content particular to your business, training goals, marketing focus, or branding interest), please contact info@rolandwanner.com

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.

V1.1 October 2022 (First Edition 2019)
ISBN: 978-1080395002

proconis Publishing

Table of Contents

This book is based on the **Scrum Guide™** from
2020 by Ken Schwaber and Jeff Sutherland
and on the **Scrum@Scale™ Guide** by Jeff Sutherland of
Scrum Inc. from 2019

Preface

Congratulations for wanting to know more about Agile Project Management and Scrum! This is the ideal book for you to learn everything important about agile project management and Scrum or to further deepen existing knowledge. 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 on the rise, and not just in project management. Other areas of management are also slowly trying to take advantage of them, to streamline processes, to focus even more on the customer and customer benefits, to react even faster to market changes and to "experiment" more with self-organizing teams. But agile methods also aim to make work more meaningful, more interesting and more rewarding. If you're a bit older, some of these topics may sound familiar. Self-organizing teams, for example, have been used successfully in various companies since the 1960s—but have never really established themselves until now.

Successful Management Methods Rediscovered

The principles, methods and values on which agile project management is based have been around for decades. Almost all of them originate from the Toyota Production System.

The Toyota Production System (TPS), which emerged after World War II, was a driver for many effective management methods, such as Lean Production, Lean Management, Business Process Reengineering and Six Sigma. Self-organizing teams were also a core element of the TPS and were then used in the 1950s and 1960s in various Japanese and later in American companies.

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.

Project work in industry, construction and other sectors has been carried out in sequential phases for centuries. When information technology and software development emerged in the 1960s, the sequential approach was also adopted in these projects. In the following years, various more or less successful iterative software development methods were developed to make software development more efficient and to be able to react better to changing market requirements.

In 1995, Jeff Sutherland and Ken Schwaber together presented a document describing the Scrum method for software projects. They developed, based on the very successful, but already almost forgotten values, principles and methods, which you briefly got to know earlier, the most successful project management method for software development - since software is 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.

Whether you are already working in the software field, a student, a project owner for software or already working in an agile project - this book provides you with the necessary knowledge to better understand and successfully apply agile project management and especially Scrum. 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 "traditional" project management. Then, you will read what the Agile Manifesto is and what the agile principles mean for Scrum and the other agile methods.

The main part of this book is of course about Scrum. First, I give you a quick **overview of the Scrum Framework** starting on page 33. This will give you a basic knowledge within five minutes and you will understand the context better later on.

The following chapters give you an **overview** of the values of Scrum, the Scrum events and the artifacts. The further chapters then deepen what you have 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 serves to make you become more immersed in Scrum and learn more and more details of the different 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 2020 and the Scrum@Scale Guide from Mai 2019.

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/

How to Get the eBook

If you are reading this book in paperback format, you will not be able to open the hyperlinks. I know that's unfortunate. And maybe you don't like carrying books around and prefer reading on your tablet.

Get the eBook (PDF) for free with active hyperlinks.

To receive the eBook, please send your purchase receipt or a self-made photo showing you and the paperback book at the same time to:

info@rolandwanner.com

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.

The success of agile methods at the enterprise level, however, ultimately only comes about through radical changes within the organizations, and here we are still far from making great progress. At the project level, on the other hand, we are already very far advanced.

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-organizing and cross-functional teams, managers as coaches without a leadership function and the reduction of management levels was already an "interesting topic" for a short time decades ago. I hope that we will have more success with this in the next few years.

Agile methods and Scrum have been successfully applied in software development for more than 20 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 the environment and requirements more dynamic. Customers needed faster usable software, not with all functions, but the most important ones. Software development had to become 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 leaner than is the case with classic process models. Agile software development is characterized by self-organizing teams, as well as an iterative and incremental approach. The aim is to get by with less bureaucracy and fewer rules to 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 (sponsor) 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.

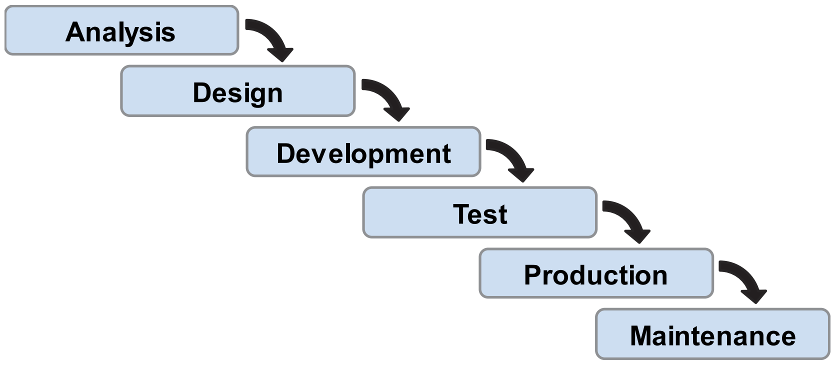


Figure 1: The Waterfall Model

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-organizing development team
- Short development cycles, whereby changes and new requirements are additionally incorporated into the planning process for next cycles

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

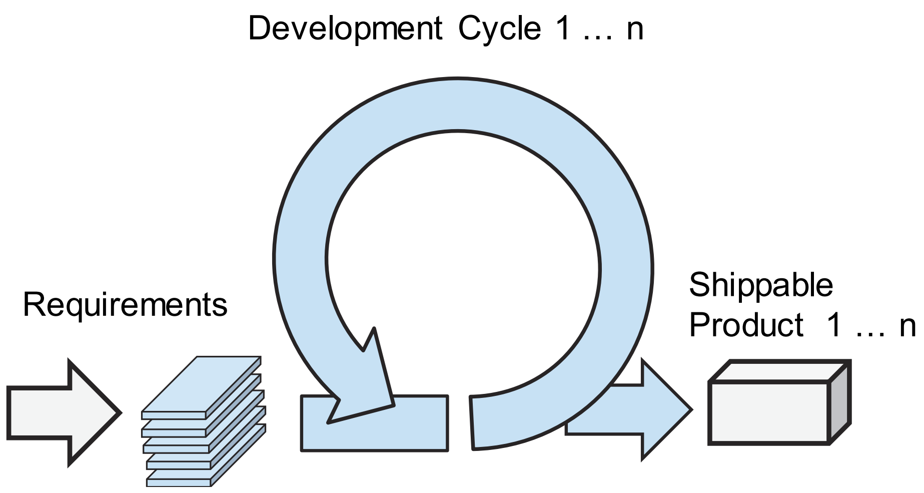


Figure 2: The Agile Process

Agile Project Management for all Projects?

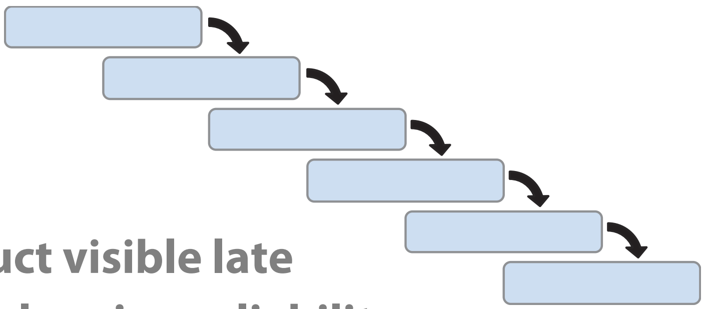
Agile Project Management is gaining more and more acceptance, even outside traditional software development. Many predict the death of the waterfall model. It won't come to that. Try to develop and manufacture a house, aircraft carrier, machine tool, or production plant with Agile Project Management. 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.

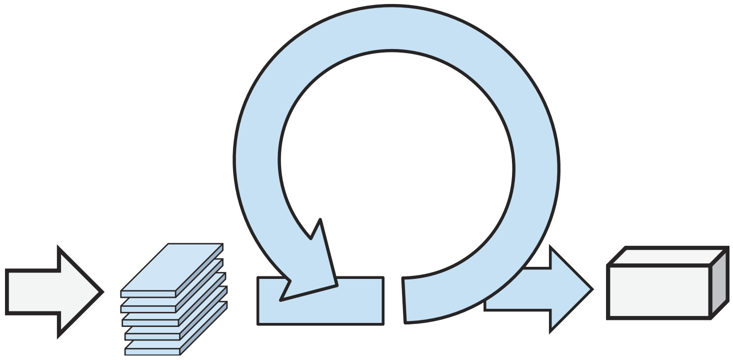
Certain principles, practices and values of agile project management will certainly have a strong influence on traditional project management in the future. I'm just thinking about the teamwork and the agile values and principles. But wherever physical products are developed, the flexibility in the development 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. The main reasons were heavy-weight, sequential implementation methods and insufficient 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.

16

Glossary

Below you will find an overview of the most important terms of Scrum and agile project management, which I used in this book. This is the place to go if you are unclear about a term in agile project management.

Accountable – Responsible is the person who is expected or required to do a work. Accountable is the person who ensures that the work is done and done correctly.

Agile Manifesto – The Manifesto defined in 2001 is the basis of all agile approaches and describes the core agile values.

Agile Principles – The Agile Manifesto is based on 12 principles, which every user of agile methods is committed to adhere to.

Agile Values – These are the essential basic values for successful, agile project management. In summary: More flexibility, fewer structures, more productive cooperation. (see also Agile Manifesto).

Acceptance Criteria – Each backlog item (user story) has acceptance criteria. These are part of the completion criteria (Definition of Done) and are used by the Product Owner in the review of the finished user story.

Artifact – Artifacts are the results of activities in the software development process. Central Scrum artifacts are: Product Backlog, Sprint Backlog and the Product Increment.

Burndown-/Burnup-Chart – A chart with which the Developers visualize the remaining work in the Sprint in relation to the remaining time in the Sprint. It shows the Sprint progress. It can be used at project, release and Sprint level.

Business Value – It is the (usually financial) value of a backlog item for a company or the customer. The Product Backlog is usually sorted so that items with the highest business value are developed first.

Commitment – In Sprint Planning, the Developers are committed providing a "forecast" of which backlog items will be implemented in the next Sprint. But commitment also includes commitment to the team, to quality, to cooperation and to learning. Commit to doing the best I can—every day.

Chief Product Owner (CPO) – Is the scaled role of the Product Owner in Scrum@Scale. He is responsible for the overall product, the product vision and the overall Product Backlog of the project. The CPO coordinates priorities among Product Owners who work with individual teams in the Meta Scrum.

Daily Scrum – Is a 15-minute meeting where the Developers synchronizes their activities and work on planning for the next 24 hours. It is often held as a stand-up meeting.

DEEP – According to Mike Cohen, a good Product Backlog is DEEP. This means: Detailed, Estimated, Emergent, Prioritized.

Definition of Done (DoD) – Is the agreement of an agile team on what needs to be done in order for a feature to be considered completed. The DoD is a list of completion criteria.

Definition of Ready (DoR) – A list of criteria (generally based on the INVEST matrix) that a user story must meet prior to being accepted into the upcoming iteration. In contrast to the DoD, the Product Owner is responsible for compliance with the DoR.

Developers – The Developers carry out all work to convert the requirements into a usable Product Increment. They work self-managed.

DevOps – Close collaboration between development and operations in an agile environment and automation throughout the lifecycle enables releases to be delivered faster, of higher quality, at lower cost and at lower risk.

Documentation – This is also necessary and helpful in agile projects. However, the principles of the Agile Manifesto take precedence (Working software over comprehensive documentation).

Done – In Scrum it is strictly distinguished between "not finished" and "finished". A backlog item is considered "Done" when all tasks are completed and the criteria of the Definition of Done are fulfilled.

Epic – Is a coarse granular requirement (large user story) that is too large or too complex to be implemented in a Sprint. Later, when the customer requirements are clearer and more details are known, it is divided into user stories in the Product Backlog.

Estimation – This usually means estimating the effort of Product Backlog items or user stories. Estimation methods are usually Planning Poker and Magic Estimation.

Estimation Meeting – The Estimation Meeting (also called Backlog Refinement) is the planning meeting with the Product Owner to get clarity for the next Sprint. The aim here is to clarify details, estimate effort and check the prioritization.

Event – See Scrum Event.

Extreme Programming – This is an agile software development method, which was mainly developed by Kent Beck. It follows the five central agile values: communication, simplicity, respect, feedback and courage.

Feature – Is a functional property of the product to be developed.

Forecast – The Developers provide a forecast in Sprint planning which backlog items they will implement in the next Sprint.

Impediment – Are problems of any kind that hinder the Developers in the effective execution of their work.

Impediment Backlog – Is a public work list of the Product Owner hanging in the team room with all impediments.

Increment – An increment is part of a whole. A Product Increment in Scrum is the result of a Sprint, as a finished part of the whole product.

Inspect & Adapt – A principle of continuous improvement. The own work is checked at regular intervals and, if necessary, the work processes are adapted in the next iteration.

INVEST – Good user stories meet INVEST criteria. They are: (I) Independent, (N) Negotiable, (V) Valuable, (E) Estimable, (S) Small, (T) Testable.

Iteration – A period of time during a Product Increment is developed. In Scrum this is called "Sprint", which always has the same duration.

Iterative incremental – In Scrum, this technique is used to create software through short feedback cycles. Iterative (in individual Sprints), increments (finished product parts) are supplied.

Customer – Is the person/company who orders the product result (the client, project sponsor) and also pays for it. He determines what the product should achieve.

Lean Management – Is a management philosophy that includes thinking principles, methods and procedures for the efficient design of the entire product value chain.

Magic Estimation – Similar effort estimation method for user stories like Planning Poker®, but much more efficient.

Manager – (Line) managers are also important stakeholders in Scrum projects. They can support the project and remove obstacles, but also create an environment in which agile principles and projects can develop.

MetaScrum – In Scrum@Scale, this is the scaled version of the Backlog Refinement Meeting of the Product Owners. The PO's coordinate a backlog themselves, which then feeds a Scrum of Scrums (SoS). For each SoS there is an associated MetaScrum.

Minimal Marketable Feature (MMF) – The smallest possible number of functions that can be marketed on their own. Software that has only one of these features has a benefit for the user for whom he would pay.

Minimal Viable Product (MVP) – The Minimal Viable Product is the minimum number of features that are necessary to find out what a customer wants. It is a strategy to get quick and broad feedback from the customer without working out the product to market maturity.

MuSCoW – The MuSCoW method is a prioritization technique for classifying backlog items roughly into four categories: Must have, Should have, Could have, Won't have.

PBI – Product Backlog Item. An element in the Product Backlog

PDCA Cycle – Is a four-step management process used for continuous improvement in software development: Plan (P), Do (D), Check (C) Act (A).

Planning – see Sprint Planning

Planning Poker® – Is a technique for estimating backlog items in Estimation Meetings. It is particularly important that the size of a feature is estimated and not the effort involved.

Product Backlog – The Product Backlog is a list of requirements (or user stories) that will be implemented in the current project. The Product Owner makes it available, prioritizes, takes responsibility for it and maintains it together with the Developers.

Product Increment – Is executable, tested and documented software that implements requirements from the Product Backlog. The Product Increment is the result of a Sprint.

Product Owner – The Product Owner is responsible for the economic success and the "What" in the Scrum project. He plans and controls the development in Scrum. He is the owner of the Product Backlog, prioritizes it and determines which features are implemented when.

Product Owner Team – A group of Product Owners who need to coordinate a shared Product Backlog that feeds a network of teams. For each SoS there is an associated Product Owner Team which aligns the teams' priorities along a single path.

Product Backlog Item (PBI) – This is a requirement usually described in a user story. It is a work unit small enough for the Scrum Team to complete it in a Sprint.

Product Backlog Refinement – This is how the ongoing maintenance of the Product Backlog is called (see also Estimation Meeting).

Product Vision – The product vision is created by the Product Owner before the first Sprint. It leads the Scrum in the right direction and is an overarching goal that everyone must share, the Scrum Team, the customer and the stakeholders. The vision describes why the project is being undertaken and what the desired final state is.

Release – Is a software version that can be used productively by the user. One or more releases are the result of the Scrum project.

Release-Burndown-Chart – This is a burndown chart that shows how many story points still need to be implemented until the end of each release.

Release Planning – Is the medium-term planning of the project and gives the customer an overview when he will receive which functionalities.

Requirements – Requirements are usually defined in a user story in Scrum. This is why most people talk about user stories and not about requirements.

Requirements list – see Product Backlog

Scaled Daily Scrum (SDS) – Is an event of Scrum@Scale. For large projects, the SDS event reflects the Daily Scrum by optimizing the collaboration and performance of the network of Scrum Teams.

Scrum Events – There are five events in Scrum. Four are meetings of the Scrum Team: Sprint Planning, Daily Scrum, Sprint Review, Sprint-Retrospective and the fifth is the Sprint itself.

Scrum Guide – Published and regularly updated by Ken Schwaber and Jeff Sutherland and describes the official Scrum basics.

Scrum@Scale™ Guide – Published by Jeff Sutherland and Scrum Inc. and describes how Scrum can be easily scaled. The first version was released in February 2018.

Scrum Master – The Scrum Master is a Scrum role. He helps to apply Scrum correctly, supports the team and ensures optimal cooperation between Product Owner and the Developers. The Scrum Master removes impediments and helps the team to continuously increase its maturity and productivity.

Scrum of Scrums (SoS) – Is a team of Scrum Masters and other necessary skills of the involved Scrum subprojects. An SoS acts as a release team and must be able to directly deliver value to the customer.

Scrum of Scrums Master (SoSM) – Is a role of Scrum@Scale. The Scrum of Scrums Master (SoSM) is accountable for the release of the joint teams' effort.

Scrum-Roles – (Accountabilities) These are: Product Owner, Scrum Master, Developers.

Scrum-Team – Consists of the three Scrum roles: Product Owner, Scrum Master, Developers.

Scrum-Values – Scrum is based on five values: commitment, courage, focus, openness and respect. When these values are embodied and lived by the Scrum Team, the Scrum pillars of *transparency, inspection and adaptation* come alive and build trust among all participants.

Self-managing – Is an essential characteristic of Scrum Teams. The team organizes its work by itself. Decisions are made in the team, the team agrees on team-internal rules and working methods. Only the achievement of the Sprint goal is a yardstick for the team.

Selected Product Backlog – see Sprint Backlog.

Servant Leadership –The Scrum Master is a leader who serves the Scrum Team. This means that although he has influence, he has no authority over the organization of work in the team. He acts as coach and change agent.

Sprint – In Scrum iterations are called Sprint. A Sprint has a fixed duration in which the Developers convert the customer's requirements into functional software.

Sprint Backlog – In Sprint Planning 1, the Developers decide how many of the highest prioritized backlog items they want to implement in the following Sprint. The selected items (Selected Product Backlog) are then the Sprint Backlog for the upcoming Sprint. The Sprint Backlog corresponds to the forecast of the Developers, what they wants to achieve together in the upcoming Sprint.

Sprint Burndown Chart – see Burndown Chart

Sprint Planning Meeting – The Sprint Planning Meeting takes place at the beginning of a Sprint and consists of two parts. Both meetings last a maximum of four hours each for a 4-week Sprint.
In the first part, the Scrum Team defines the Sprint Backlog, which means WHAT it wants to do in the next Sprint. In the second part, the implementation details, the HOW, are clarified. At the end of the Sprint planning all user stories (WHAT) and the corresponding tasks are displayed on the taskboard.

Sprint Retrospective – Is performed after each Sprint Review. The Sprint and the events are analyzed and actions for the next Sprints are defined with the aim of constantly learning and improving.

Sprint Review – Takes place at the end of each Sprint with the aim of presenting the results to the Product Owner, the customer and the stakeholders in order to receive feedback. The Product Owner checks whether the Developers have implemented all user stories according to the forecast.

Sprint Cancelation – The Product Owner can cancel a Sprint if he sees that the Sprint goal cannot be reached or no longer makes sense. In this case, a Sprint Review is performed, and all completed "Done" Product Backlog entries are reviewed and, if useful, approved by the Product Owner.

Sprint Goal – The Sprint goal is defined by the Product Owner and describes the Product Increment to be delivered—preferably in one sentence. This is what the Product Owner wants to deliver to the customer at the end of the Sprint.

Stakeholder – Stakeholders are people who are affected by or interested in the project result but do not actively participate in the project, e.g. users, customers, sales, marketing, managers.

Sprint 0 – This is often referred to as an initial Sprint. It is usually shorter than a normal Sprint and produces no business value. Here, for example, the product backlog is prepared, the infrastructure set up, architecture questions clarified, and rules defined as to how the Scrum Team works together.

Story – see User Story

Story Card – Describes a backlog item (user story) on an index card with a maximum of two sentences. On the front is also the business value and the effort in story points. On the back are e.g. the acceptance criteria and the test strategy.

Story Point – Story Points are an abstract unit used to evaluate the effort (size) of a backlog item, always relative to another story. The effort can only take certain predefined values, which usually correspond to the Fibonacci sequence.

Task – In Sprint Planning, the Developers define the tasks which are needed to implement the user stories. These are then attached to the taskboard next to the user stories.

Taskboard – The taskboard visualizes the Sprint Backlog. It shows at any time, which Product Backlog items have been selected for the Sprint, which associated tasks have to be performed, and in which completion state these tasks are.

Team – see Scrum Team

Timebox – All Scrum events have a time limit (Timebox). The event is terminated after a certain time, even if the content has not yet been completed. The aim of the timebox is to avoid waste in order to focus on the really important things in the Sprint.

User – The user is the person who uses the project result. Scrum places the delivery of functionality for a user in the foreground, not the fulfilment of tasks. It is a proven method to describe the backlog items from the user's point of view in a user story.

User Story – Product backlog items (PBI), requirements or features, are usually described in user stories. User stories describe requirements from the user's perspective. User stories are written in a special format: As <role> I want to <goal/wish> to achieve <benefit/reason>.

Velocity (of development) – The sum of all story points that the Scrum Team has implemented in a Sprint. Over time, the velocity levels off to a relatively stable value. Knowing the velocity is important for the Product Owner so that he can create the release plan and adjust it during the course of the project

Vision – see Product Vision

Values – see Scrum Values

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 These 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 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

The Essential Guide to Agile Project Management and Scrum for all Team Members, Managers and Executives.

Reduce Your Development Cycles, React More Flexible to Market Changes and Develop New Products Faster!

Do you want to react much faster to market changes with innovative products, radically shorten your time to market, offer a much faster return on investment and more flexibility for your customers? Agile Project Management and Scrum offer not only that but also more interesting and satisfying work for your project team and a much more productive collaboration. Teams organize their own work and take on more responsibility.

This book introduces, clarifys and deepens your knowledge about Agile Project Management and Scrum to use it successfully. It describes Scrum comprehensively, systematically and easily understandable. Main topics are:

- Agile Project Management
- The Scrum Framework
- Scrum for large Projects

With its comprehensive glossary of definitions of all key terms, this book is equally suited as a comprehensive introduction and reference guide for business and for educational purposes.

This is a book for beginners in Scrum and Agile Project Management, but also for advanced readers who are preparing for a Scrum certification.

This book is based ont eh current Scrum Guide from November 2020 from Ken Schwaber and Jeff Shuterland

Go to **www.rolandwanner.com**
for additional content

Business & Economics / Project Management

The Essential Guide