BEST PRACTICE

SCRUM

A Smart Travel Companion

A Pocket Guide

Gunther Verheyen

Van Haren PUBLISHING

Other publications by Van Haren Publishing

Van Haren Publishing (VHP) specializes in titles on Best Practices, methods and standards within four domains:

- IT and IT Management
- Architecture (Enterprise and IT)
- Business Management and
- Project Management

Van Haren Publishing offers a wide collection of whitepapers, templates, free e-books, trainer materials etc. in the **Van Haren Publishing Knowledge Base**: www.vanharen.net for more details.

Van Haren Publishing is also publishing on behalf of leading organizations and companies: ASLBiSL Foundation, BRMI, CA, Centre Henri Tudor, Gaming Works, IACCM, IAOP, Innovation Value Institute, IPMA-NL, ITSqc, NAF, KNVI, PMI-NL, PON, The Open Group, The SOX Institute.

Topics are (per domain):

IT and IT Management
ABC of ICT
ASL®
CATS CM®
CMMI®
COBIT®
e-CF
ISO 20000
ISO 27001/27002
ISPL
IT4IT®
IT-CMF™
IT Service CMM
ITIL®
MOF
MSF
SABSA
SAF
SIAM

Enterprise Architecture
ArchiMate®
GEA®
Novius Architectuur Methode
TOGAF®

Business Management
BABOK® Guide
BiSL® and BiSL® Next
BRMBOK™
BTF
EFQM
eSCM
IACCM
ISA-95
ISO 9000/9001
OPBOK
SixSigma
SOX
SqEME®

Project Management
A4-Projectmanagement
DSDM/Atern
ICB / NCB
ISO 21500
MINCE®
M_o_R®
MSP®
P3O®
PMBOK® Guide
PRINCE2®

For the latest information on VHP publications, visit our website: www.vanharen.net.

Scrum

A Pocket Guide

Gunther Verheyen

Van Haren PUBLISHING

Colophon

Title:	Scrum – A Pocket Guide
Subtitle:	A smart travel companion
Series:	Best Practice
Author:	Gunther Verheyen
Reviewers:	Ken Schwaber (Scrum co-creator, Scrum.org) David Starr (Agile Craftsman, Microsoft) Ralph Jocham (Agile Professional, effective agile) Patricia Kong (Director of Partners, Scrum.org)
Text editor:	Steve Newton
Publisher:	Van Haren Publishing, Zaltbommel, www.vanharen.net
ISBN hard copy:	978 90 8753 720 3
ISBN eBook:	978 90 8753 794 4
Edition:	First edition, first impression, October 2013 First edition, second impression, January 2014 First edition, third impression, June 2014 First edition, fourth impression, November 2015 First edition, fifth impression, November 2016 First edition, sixth impression, July 2017
Layout and typesetting:	Coco Bookmedia, Amersfoort – NL
Copyright:	© Van Haren Publishing, 2013

For any further enquiries about Van Haren Publishing, please send an e-mail to: info@vanharen.net

Foreword by Ken Schwaber

An outstanding accomplishment that simmers with intelligence.

Scrum – A Pocket Guide is an extraordinarily competent book. Gunther has described everything about Scrum in well-formed, clearly written descriptions that flow with insight, understanding, and perception. Yet, you are never struck by these attributes. You simply benefit from them, later thinking, "That was really, really helpful. I found what I needed to know, readily understood what I wanted, and wasn't bothered by irrelevancies."

I have struggled to write this foreword. I feel the foreword should be as well-written as the book it describes. In this case, that is hard. Read Gunther's book. Read it in part, or read it in whole. You will be satisfied.

Scrum is simple, but complete and competent in addressing complex problems. Gunther's pocket guide is complete and competent in addressing understanding a simple framework for addressing complex problems, Scrum.

Ken, August 2013

Preface

The use of Agile methods continues to grow traction and Scrum is the most widely adopted method for Agile software development. The general level of interest in Scrum is therefore considerable.

Transforming an organization's development approach to Scrum represents quite a challenge. Scrum is not a cookbook 'process' with detailed and exhaustive prescriptions for every imaginable situation. Scrum is a *framework* of principles, roles and rules that thrive on the *people* doing Scrum. The true potential of Scrum lies in the discovery and *emergence* of practices, tools and techniques and in optimizing them for each organization's specific context. Scrum is very much about behavior, much more than it is about process.

The benefits an organization realizes with Scrum depend on the will to remove barriers, think across boxes and embark on a journey of discovery.

The journey starts by playing Scrum. This requires knowledge of the rules of Scrum. This book describes these. This book shows how Scrum implements the Agile mindset, what the rules of the game of Scrum are, and how these rules leave room for a variety of tactics to play

the game. The introduction of all essentials of Scrum and the most common tactics for Scrum makes this book a worthwhile read for people, teams, managers and change agents, whether they are already doing Scrum or want to embark on the journey of Scrum.

Ten years ago I started my journey, my path of Agility via Scrum. It has inevitably been a cobblestone path. On my journey I have used Scrum with plenty of teams, in various projects, and at different organizations. I have worked with both large and small enterprises and have coached teams as well as executive management. I was in the fortunate position of then moving to Scrum.org. It's where I shepherd the 'Professional' series of Scrum trainings, courseware and assessments.

I thank Ken Schwaber, David Starr, Ralph Jocham, and Patricia Kong for reviewing early versions of this book and improving it with much appreciated feedback.

I thank all at Van Haren Publishing for their trust and confidence, and for giving me the chance to express my views on Scrum with this book.

I thank my colleagues at Scrum.org for our daily collaboration, the positive action and the energy, and especially Ken Schwaber for our exquisite partnership.

Enjoy reading, and... keep Scrumming.

Gunther, June 2013

Reviews

This Scrum Pocket Guide is outstanding. It is well organized, well written, and the content is excellent. This should be the de facto standard handout for all looking for a complete, yet clear overview of Scrum.

(Ken Schwaber, Scrum co-creator, Scrum.org)

Gunther has expertly packaged the right no-nonsense guidance for teams seeking agility, without a drop of hyperbole. This is the book about agility with Scrum I wish I had written.

(David Starr, Agile Craftsman, Microsoft)

During my many Scrum training activities I often get asked: "For Scrum, what is the one book to read?" In the past the answer wasn't straight forward, but now it is! The Scrum Pocket Guide is the one book to read when starting with Scrum. It is a concise, yet complete and passionate reference about Scrum.

(Ralph Jocham, Agile Professional, effective agile.)

'The house of Scrum is a warm house. It's a house where people are WELCOME.' Gunther's passion for Scrum and its players is evident in his work and in each chapter of this book. He explains the Agile paradigm, lays out the Scrum framework and then discusses the 'future state of Scrum.' Intimately, in about 100 pages.

(Patricia M. Kong, Director of Partners, Scrum.org)

Contents

1 The Agile paradigm

1.1 TO SHIFT OR NOT TO SHIFT

The software industry was for a long time dominated by a paradigm of *industrial* views and beliefs (figure 1.1). This was in fact a copy-paste of old manufacturing routines and theories. An essential element in this landscape of knowledge, views and practices was the Taylorist[1] conviction that 'workers' can't be trusted to undertake intelligent and creative work. They are expected to only carry out executable tasks. Therefore their work must be prepared, designed and planned by more senior staff. Furthermore, hierarchical supervisors must still vigilantly oversee the execution of these carefully prepared tasks.

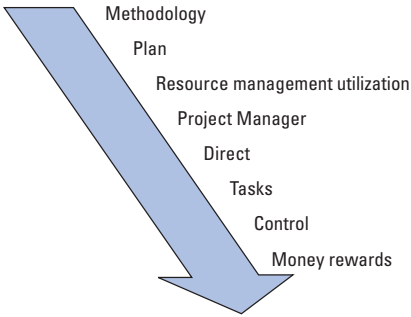

Figure 1.1 The industrial paradigm

Quality is assured by admitting the good and rejecting the bad batches of outputs. Monetary rewards are used to stimulate desired behavior. Unwanted behavior is punished. It's like carrots and sticks.

The serious flaws of this paradigm in software development are known and well documented. In particular, the Chaos reports of the Standish Group have over and over again revealed the low success rates of traditional software development. The latest of these reports is dated 2011 (Standish, 2011). Many shortcomings and errors resulting from the application of the industrial paradigm in software development are well beyond reasonable levels of tolerance. The unfortunate response seems to have been to lower the expectations. It was accepted that only 10-20% of software projects would be successful. Success in the industrial paradigm is made up of the combination of on time, within budget and including all scope. *Although these criteria for success can be disputed, it is the paradigm's promise.* It became accepted that quality is low, and that over 50% of features of traditionally delivered software applications are never used (Standish, 2002).

Although it is not widely and consciously admitted, the industrial paradigm did put the software industry in a serious crisis. Many tried to overcome this crisis by fortifying the industrial approach. More plans were created, more phases scheduled, more designs made, more work was done upfront, hoping for the actual work to be undertaken to be executed more effectively. The exhaustiveness of the upfront work was increased. The core idea remained that the 'workers' needed to be directed with even more detailed instructions. Supervision was increased and intensified.

And still, little improved. Many flaws, defects and low quality had to be tolerated.

It took some time, but inevitably new ideas and insights started forming following the observation of the significant anomalies of the industrial paradigm. The seeds of a new world view were already sown in the 1990's. But it was in 2001 that these resulted in the formal naming of 'Agile', a turning-point in the history of software development. A new paradigm for the software industry was born (figure 1.2); a paradigm that thrives upon heuristics and creativity, and restoring the respect for the creative nature of the work and the intelligence of the 'workers' in software development.

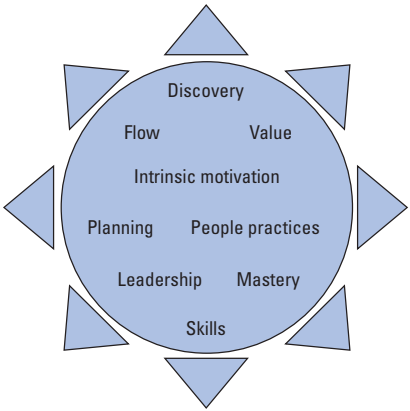

Figure 1.2 The Agile paradigm

The software industry has good reasons to move fast to the new paradigm; the existing flaws are significant, widely known and the presence of software in society grows exponentially, making it a critical aspect of our modern world. However, by definition, a shift to a new paradigm takes time. And the old paradigm seems to have deep roots. An industrial approach to software development even continues to be taught and promoted as the most appropriate one.

Many say that Agile is too radical and they, therefore, propagate a gradual introduction of Agile practices into the existing, traditional process. However, there is reason to be very skeptical about a gradual evolution, a slow progression from the old to the new paradigm, from waterfall to Agile.

The chances are quite high that a gradual evolution will never go beyond the surface, will not do more than just scratch that surface. New names will be installed, new terms and new practices will be imposed, but the fundamental thinking and behavior of people and organizations will remain the same. Essential flaws will remain untouched; especially the disrespect for people that will lead to the continued treatment of creative, intelligent people as mindless 'workers'.

The preservation of the traditional foundation will keep existing data, metrics and standards in place, and the new paradigm will be measured against these old standards. Different paradigms by their nature consist of fundamentally different concepts and ideas, often even mutually exclusive. In general, no meaningful comparison between the industrial and the Agile paradigms is possible. It requires the honesty to accept the serious flaws of the old ways, and for leadership and entrepreneurship to embrace the new ways, thereby abandoning the old thinking.

A gradual shift is factually a status-quo situation that keeps the industrial paradigm intact.

There is overwhelming evidence that the old paradigm doesn't work. But much of the evidence on Agile was anecdotal, personal or relatively minor. The Chaos report of 2011 by the Standish Group marks a turning point. Extensive research was done in comparing

traditional projects with projects that used Agile methods. The report shows that an Agile approach to software development results in a much higher yield, even against the old expectations that software must be delivered on time, on budget and with all the promised scope. The report shows that the Agile projects were three times as successful, and there were three times fewer failed Agile projects compared with traditional projects. It is clear that against the right set of expectations, with a focus on active customer collaboration and frequent delivery of *value*, the new paradigm would be performing even better.

Yet, Agile is a choice, not a must. It is one way to improve the software industry. Research shows it is more successful.

The distinct rules of Scrum help in getting a grip on the new paradigm. The small set of prescriptions, as described in the following chapter, allows immediate action and results in a more fruitful absorption of the new paradigm. Scrum is a tangible way to adopt the Agile paradigm. Via Scrum, people do develop new ways of working; through discovery, experimentation-based learning and collaboration. They enter this new state of being, this state of *agility*; a state of constant change, evolution and improvement.

Nevertheless, despite its practicality, experience shows that adopting Scrum often represents a giant leap. This may be because of uncertainty, letting go of old certainties even if they prove not to be very reliable. It may be because of the time that it takes to make a substantial shift. It may be because of the determination and hard work that is required.

1.2 THE ORIGINS OF AGILE

Despite the domination of the plan-driven, industrial views, an evolutionary approach to software development is not new. Craig Larman has extensively described the historical predecessors of Agile in his book 'Agile & Iterative Development, A Manager's Guide' (Larman, 2004).

But the official label 'Agile' dates from early 2001, when 17 software development leaders gathered at the Snowbird ski resort in Utah. They discussed their views on software development in times when the failing waterfall approaches were replaced by heavy-weight RUP implementations, which did not in fact lead to better results than the traditional processes. These development leaders were following different paths and methods, each being a distinct implementation of the new paradigm; Scrum, eXtreme Programming, Adaptive Software Development, Crystal, Feature Driven Development, DSDM, etc.

The gathering resulted in assigning the label 'Agile' to the common principles, beliefs and thinking of these leaders and their methods. They were published as the 'Manifesto for Agile Software Development' (Beck, et.al., 2001). (See figure 1.3).

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.

Figure 1.3 The text of the Manifesto for Agile Software Development

I often overhear the desire "to do Agile". And all too often it is the desire for a magical solution, another silver bullet process that solves all problems. It makes me state that *"Agile does not exist"*. Agile is not one fixed process, method or practice. Agile is the collection of principles that the methods for Agile software development have in common. Agile refers to the mindset, the convictions and the preferences expressed in the Manifesto for Agile Software Development.

The manifesto does help to grasp the ideas underpinning Agile. If you use it as a source to gain a deeper understanding of Agile, then I strongly advise looking at the 12 principles, see: http://agilemanifesto.org/principles.html

1.3 DEFINITION OF AGILE

I prefer to describe 'Agile' in terms of the following key characteristics that are common to the portfolio of Agile methods:

- People driven;
- Facilitation;
- Iterative-incremental process;
- Measuring success;
- Change.

1.3.1 People driven

Agile software development is not driven by a predictive plan describing how to implement analyzed, designed and architected requirements. Agile acknowledges that requirements cannot be predicted in every possible detail in an upfront way.

Agile is not a process of handing over different types of intermediate deliverables to different specialist departments, where each department performs its specialized work in isolation.

Agile is driven by the continuous *collaboration* of people ranging over all required departments; whether they are called business, IT, marketing, sales, customer service, operations or management.

People are respected for their creativity, intelligence and self-organizing capabilities. People are respected for their ability to understand and resolve a problem without being overloaded with too much ceremony and bureaucracy. A ceremonial overload only replaces this collaborative thinking, innovation and accountability of people with bureaucracy, paper results, handovers and administrative excuses.

People are respected in the time they can spend on their work via the idea of *Sustainable Pace*. Work is organized in such a way that the tempo is sustainable, indefinitely.

1.3.2 Facilitation

Agile replaces the traditional command-and-control mechanisms of assigning individuals on a daily basis with executable micro-tasks and totalitarian authorities for invasive control.

Agile teams are *facilitated* by servant-leadership. Boundaries and a context for self-management exist, upon which teams are given objectives and direction. Subtle control emerges from the boundaries.

1.3.3 Iterative-incremental process

Agile processes are not free-play approaches. Agile processes are defined and they require high discipline.

Products are created piece by piece ('incremental') with each piece being made up of expansions, improvements and modifications. The built pieces and the total product are frequently revisited ('iterative') to assure overall integrity.

Agile requires explicit attention from all players on quality and excellence. Agile replaces the idea that these can simply be poured into documents and paper descriptions.

1.3.4 Measuring success

Progress in software development cannot be measured and guaranteed on the basis of mere compliance with predictive plans and milestones, documents, handovers, signatures, approvals or other ceremonial obligations as is the case in the industrial paradigm.

Agile makes it explicit that success and progress in software development can only be determined by frequently inspecting *working software* and the actual *value* it holds for the people who will have to use it.

It is a natural part of software development that the people having to use the software can never be sure on the usability and usefulness until they actually get their hands on it. No paper documentation or virtual process can replace this.

Agile recognizes no business versus IT discord. The two are needed for success, from the perspective of creating both useable *and* useful software products.

1.3.5 Change

Even when requirements and implementations are predicted in an upfront way, they are prone to change. Markets and competitors