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

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.

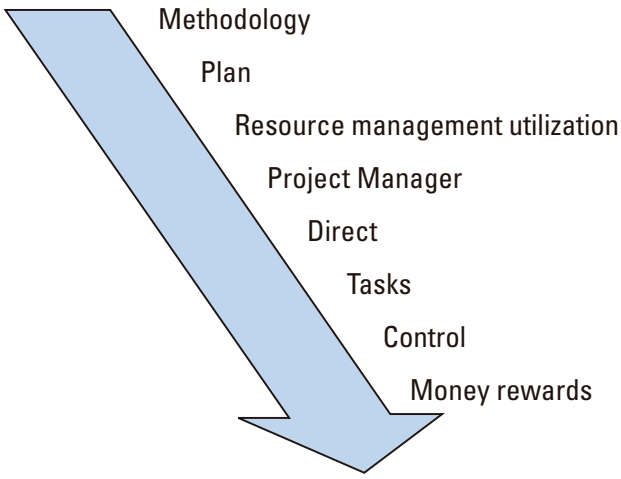


Figure 1.1 The industrial paradigm

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.

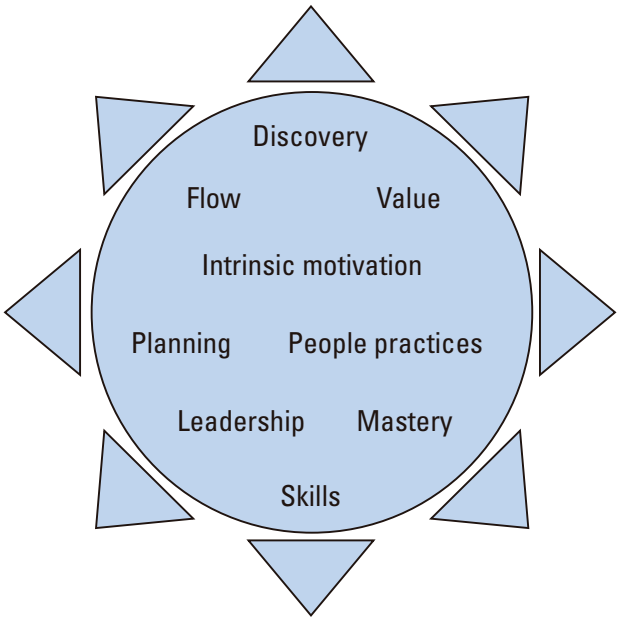


Figure 1.2 The Agile paradigm

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'.

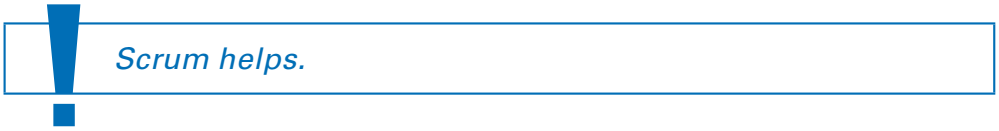
The preservation of the traditional foundation will keep existing data, metrics and standards in place, and the new paradigm will be measured against these old standards. Different paradigms by their nature consist of fundamentally different concepts and ideas, often even mutually exclusive. In general, 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.