Teaching Agile Software Development Competences

The Agile Competence Pyramid

Martin Kropp
University of Applied Sciences
Northwestern Switzerland
Windisch, Switzerland
martin.kropp@fhnw.ch

Andreas Meier
Zurich University of Applied Sciences
Winterthur, Switzerland
andreas.meier@zhaw.ch

***Abstract*—Agile methodologies have come a long way over the last decade. Several recent surveys [1], [2] show that agile methodologies like Scrum, Extreme Programming and, more recently, Kanban have successfully been adopted by many companies to develop software. However, the same surveys show that only few of the agile practices are used and even fewer are applied consequently and thoroughly. This is to a great extent due to the lack of skilled personnel. Although teaching agile software development has drawn some attention in recent research and has been discussed in several papers, we do not yet seem to be able to "deliver" the appropriately skilled personnel. What is the reason for this, and more importantly, how can we improve the situation? In this position paper we propose a more holistic approach for teaching agile software development, in which the required agile practices and values are not only integrated theoretically but also practically applied and repeated until they become a habit to students and software engineers.**

***Index Terms*—Agile, Software Development, Education**

INTRODUCTION

Many IT-companies and -departments have adopted agile software development. In the Swiss Agile Study [2], a survey conducted by the authors, these findings have been confirmed. More than half of the participating companies are using an agile methodology – Agile has become mainstream!

Unfortunately, this also has a significant impact on the agile team constitution. The early adopters of agile approaches were all highly mature and technically skilled experts in their fields. They had internalized the agile philosophy, were very productive and produced high quality results. Today's agile teams, however, are "normal" software teams, with architects, seniors and juniors in one team, and many of them are not yet familiar with the agile philosophy. Even though those teams have improved in software development to some extent, they are far less productive than the early adopter expert teams. Survey results show that quality has partially even gone down and overall costs increased. One reason for this is that many of the important agile practices are not applied as thoroughly [2] as the agile pioneers proposed.

In this position paper we will analyze the situation on the industry and education side in more detail to find out which engineering and management skills are missing and propose the "pyramid of agile competences" as a guideline to teach these skills. We will suggest a holistic teaching approach, which integrates the necessary agile skills and the core agile values, into the education of agile software development.

THE STUDY

The study, conducted among almost 140 Swiss IT companies and almost 200 IT professionals, clearly shows the benefit of agile companies in much faster time-to-market; better change responsiveness and much more satisfaction with the process, as compared to traditional plan-driven companies. On the other side, it also shows that there are some difficulties in improving the quality and maintainability of software.

A look at the concrete practices applied, shows that typical agile management practices like iteration planning, or time-boxing are applied by most of the agile companies, while some very important agile technical practices like Test-Driven-Development (TDD), or Continuous Integration and continuous quality control with metrics are still not standard in many companies.

One reason for this could be that the developers just lack the relevant competences and skills. These needed skills and competences are discussed further in the following chapter.

THE AGILE COMPETENCE PYRAMID

The "pyramid of agile competences" in Figure 1 divides the needed skills and competences into three major categories:

Mastering the technical skills or engineering practices, builds the foundation for being able to develop high quality software. These engineering practices are especially defined by eXtreme Programming [6] and include best practices like unit testing, clean coding [3], test-driven development [7], collective code ownership, continuous integration and the like. Engineering practices are mostly competences that refer to the single individual.

On the second level come the agile management practices. They define how agile projects are organized and run. Agile management practices include iterative planning, short release cycles, small releases, strong customer involvement and highly interactive teams. Management practices are typically team aspects, which require the appropriate social competences.

On top of these competences come the agile values, which are articulated in the agile manifesto [4] and are based on characteristics like mutual respect, openness, and courage.

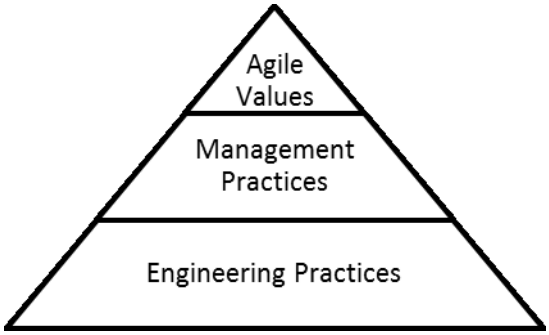

Fig. 1. Pyramid of Agile Competences

The pyramid visualizes the decreasing number of required skills from bottom to top. On the other hand, it reflects the increasing difficulty to teach these skills. Engineering practices can be taught very well in the classroom by lecturers and be learned by the individuals at their own pace. Management competences are, in our experience, best taught through projects in teams. The most difficult competences to teach are the agile values [4] on top of the pyramid, since they often require a change in the attitude of the individual.

With the pyramid we also visualize the order in which the different practice levels should be taught. Teaching software engineers the management skills before they have a good working knowledge of the underlying engineering skills is like trying to build a pyramid from top to bottom. It might be possible, but at a very high cost. (Unfortunately, many companies are using this approach: They send their engineers to a two day Scrum Master course and expect them to be agile after that).

TEACHING AGILE SOFTWARE DEVELOPMENT

The authors are convinced that all agile competences and competence levels have to be considered in an agile software engineering curriculum in an integrated approach. That can mean, that the different competences are taught in one single software engineering course by applying the agile methodology on a concrete case study in group work. This could however also mean a complete agile software development curriculum, which comprises various courses accompanied by a large one-semester team project, where the taught competences are directly applied. Especially the latter approach requires a strong coordination of the different courses which is not easy with lecturers who insist on their independence of teaching.

The authors have started to adapt their appropriate software engineering courses in the mentioned directions and have made first very positive experiences and received encouraging feedback from the students.

A. A Bottom-up Teaching Approach

At UAS Northwestern Switzerland we have introduced the course Software Construction in the second semester of the undergraduate level (B.Sc,), which focuses on many of the XP practices like unit testing, version control systems, automated builds, continuous integration, continuous quality control, refactoring and clean code [8]. These practices are applied in a concrete one-semester case study in teams of three students. Based on these engineering practices, the following module Software Project Management, focuses on management practices like iterative and incremental development, and values like openness and interaction [9]. This course teaches the agile management approach by applying a modified version of the Scrum City game [10] in teams of six students.

The feedback from the students to both courses has been very positive. They especially like the very practice oriented approach provided by the case study and the game.

B. An Integrated Teaching Approach

In 2012, a Software Engineering course was taught at Zurich University of Applied Sciences, which was designed to give the students a good working knowledge of agile software development. The course was a typical 16-week semester class in the last year of the undergraduate level (B.Sc.). In this course, the students applied agile engineering- and management practices and special attention was paid to agile values, i.e. it was designed with the pyramid of agile competences in mind.

In Scrum teams of six to eight students, computer games were developed while strengthening the students' agile software development skills. An evaluation shows that the concept of this course was well received, and that participants learned a great deal about agile methodologies while having fun.

ACKNOWLEDGMENT

Many thanks go to the Swiss Hasler Foundation, which has funded the Swiss Agile Study 2012 project, together with the Swiss IT associations, swissICT, ICTnet and SWEN. The study was the inspiration to these thoughts on teaching agile software development competences.

REFERENCES

[1] Version One. State of Agile Development Survey results. http://www.versionone.com/state_of_agile_development_survey/11/, 20.10.2012

[2] Martin Kropp, Andreas Meier, Swiss Agile Study - Einsatz und Nutzen von Agilen Methoden in der Schweiz. (German) www.swissagilestudy.ch, 20.6.2013.

[3] Robert C. Martin, Clean Code: A Handbook of Agile Software Craftsmanship, 2009, ISBN 0-13-235088-2

[4] Agile Manifesto. http://agilemanifesto.org/, 20.1.2013.

[5] Ken Schwaber, Mike Beedle. Agile Software Development with Scrum, 2001, ISBN 0-13-207489-3

[6] Kent Beck, Extreme Programming Explained: Embrace Change. Addison-Wesley, 2004 ISBN 0-321-27865-8

[7] Kent Beck, Test-Driven Development: By Example. Addison-Wesley, 2003, ISBN 0-321-14653-0

[8] Ch. Denzler, M. Kropp. Software Construction. http://web.fhnw.ch/plattformen/swc (German). 04.09.2013

[9] S. Hof, M. Kropp. Software Project Management. http://web.fhnw.ch/plattformen/spm (German). 04.09.2013

[10] Scrum Lego City Game. Agile 42. http://www.agile42.com/en/training/scrum-lego-city, 4.09.2013