Kanban for software engineering teaching in a software factory learning environment

Muhammad Ovais Ahmad, Jouni Markkula & Markku Oivo

University of Oulu
Oulu, Finland

ABSTRACT: The software industry needs university graduates with up-to-date knowledge on software development approaches and the necessary knowledge and skills to tackle the practical issues of real-life software development. Various techniques are used in industry to provide a holistic view of projects and identify obstacles in software development as they occur. Kanban is a new technique that is spreading throughout the software industry. The Department of Information Processing Science at the University of Oulu in Finland built a software factory laboratory in 2012. The aim of the laboratory is to provide students with real-life software engineering experience and enable them to practise new processes and techniques, as well as interact with actual customers from industry. To determine student perceptions about Kanban, the authors conducted a one-and-a-half-year-long study with Master's degree students in a software development project. Results indicate that using Kanban in students' project work helps students achieve their goals. This article describes and promotes the use of Kanban in students' experiential learning projects.

INTRODUCTION

In today's dynamic business and technology environment, organisations are continuously adapting their structures, strategies and policies in response to new demands. Operational costs are reduced and time-to-market is shortened by improving productivity. The increasingly important question to software companies is, how to deliver value to customers with limited time and resources. Agile and Lean approaches have been considered as viable solutions for productivity problems that appear to be hard to handle in traditional plans driven software engineering. The Lean approach has an impact on the competitiveness of organisations through improvements in their process efficiency and a reduction in their operational waste. One of the key Lean tools is the Kanban approach for managing production operations [1]. In recent years, Kanban has become more popular in software companies. The existing literature shows that Kanban enhances understanding, visibility and control of work, as well as helping to identify bottlenecks in software development as they occur [2-4]. Kanban focuses on the flow of work. Its usage is motivated by its adaptability, its support for management through visualising progress and its ability to drive team members to cooperate and communicate.

The literature shows that Kanban is spreading virally throughout the software industry [2-7]. New software engineering graduates must have both theoretical knowledge and hands-on experience on using it. The software industry needs university graduates with the knowledge and skills to tackle the practical issues of present day real-life software projects [4]. Students need an opportunity to practise Kanban and acquire other skills that are necessary for software engineering [8][9]. Most universities are teaching practical software engineering skills and capabilities to the future software professionals in hands-on project-based courses in a range of settings.

The Department of Information Processing Science (DIPS) at the University of Oulu (UO), Finland, has developed a special learning environment, a *Software Factory* (SWF), to facilitate this. The SWF is a laboratory that offers an environment for both research and education. It provides a realistic software engineering environment, which improves the students' learning experience by providing them with insights into the conduct of real-life software projects with close customer involvement, intensive teamwork, and the use of modern software development tools and processes [4]. Kanban has been adopted for use in SWF student projects for the reasons that knowledge about it is increasingly demanded by software companies and that Kanban has characteristics that can be expected to help the students to acquire essential knowledge and the competence that is needed in professional software engineering projects.

The objective of this study was to evaluate the usability and effectiveness of Kanban as a pedagogic tool for teaching software engineering in hands-on SWF projects. The study was done by using a survey to collect students' experiences and perceptions of using Kanban. The survey was presented to the SWF students after they had completed their Kanban project.

THE CONTEXT: SOFTWARE FACTORY PROJECT COURSE

The DIPS developed a Software Factory (SWF) in 2012 for facilitating project-based software engineering courses. The Oulu SWF consists of one big room for meetings, group planning or videoconferencing and three smaller rooms for working. All rooms are equipped with computers, software development tools and Kanban boards. Excluding the use of Kanban, no particular development method was enforced upon anyone. In a broader sense, the SWF can be considered as an infrastructure platform that provides and supports software engineering research, education, and entrepreneurship. As a platform, it serves multiple purposes. It is a test bed for software engineering ideas and a source for original basic scientific software development research. It is an educational vehicle for universities; the artefacts produced in the factory serve to improve learning and provide teaching materials in close collaboration with industry [4][10].

The Master's degree in Information Processing Science at UO, has a mandatory project-based course involving real software development task from an industry customer. Every semester, a pool of these projects are conducted in the SWF. The SWF project duration is 16 weeks and the tasks come from the local software companies. Each project involves a minimum of four members. The students are encouraged to tackle management and resource planning issues pertaining to large teams. The SWF project course has been designed to help the students to obtain an experience-based appreciation for their knowledge development and to acquire various skills that are needed in professional software development. The course is placed in the graduation phase of the Master's programme in order to offer the students the possibility to test and utilise as much of their newly acquired knowledge as possible in actual software development work. Each project team is assigned a project supervisor who provides the team with technical and non-technical guidance. The supervisor is also responsible for monitoring and assessing the team throughout the course of the project.

KANBAN IN SOFTWARE ENGINEERING

Kanban was developed by Taiichi Ohno and introduced into the Japanese manufacturing industry in the 1950s. Kanban literally means signboard or visualisation of inventory used in the scheduling system for just-in-time (JIT) production. Kanban runs the production system as a whole and has proved to be an excellent way to promote improvement. It was successfully used in practice by Toyota. The use of Kanban in software development originated in 2004, when David J. Anderson was assisting a small IT team at Microsoft that was operating poorly [3]. Anderson introduced Kanban to the team to enable its members to visualise their work and put limits on work in progress (WIP) [3]. Anderson based the Kanban in software development on five principles: visualise the workflow, limit WIP, measure and manage flow, make process policies explicit, improve collaboratively (using models and the scientific method) [3].

The motivation behind visualisation and limiting WIP was to identify the constraints of the process and focus on a single item at a time. This technique promotes the pull approach. In traditional software development, the work items are given, i.e. pushed, to each team member, who is, then, instructed to quickly finish as much of them as possible. The traditional development work is in the form of a chain where one team member's work item is handed over to another, i.e. from developer to tester. This causes delays in the whole work when the next member in the line is overloaded or has some problems in his/her work. Kanban works in an alternative way. Instead of pushing work items, it promotes the pull system. Each member of a team has one item to work on at a time. When he/she finishes it, he/she will automatically pull the next item to work on. In brief, Kanban aims to provide visibility in the software development process, communicate priorities and highlight bottlenecks [2]. This results in a constant flow of releasing work items to the customers, as the developers focus only on those few items at a given time [2]. Figure 1 shows the typical structure of a Kanban board and its principles in action.

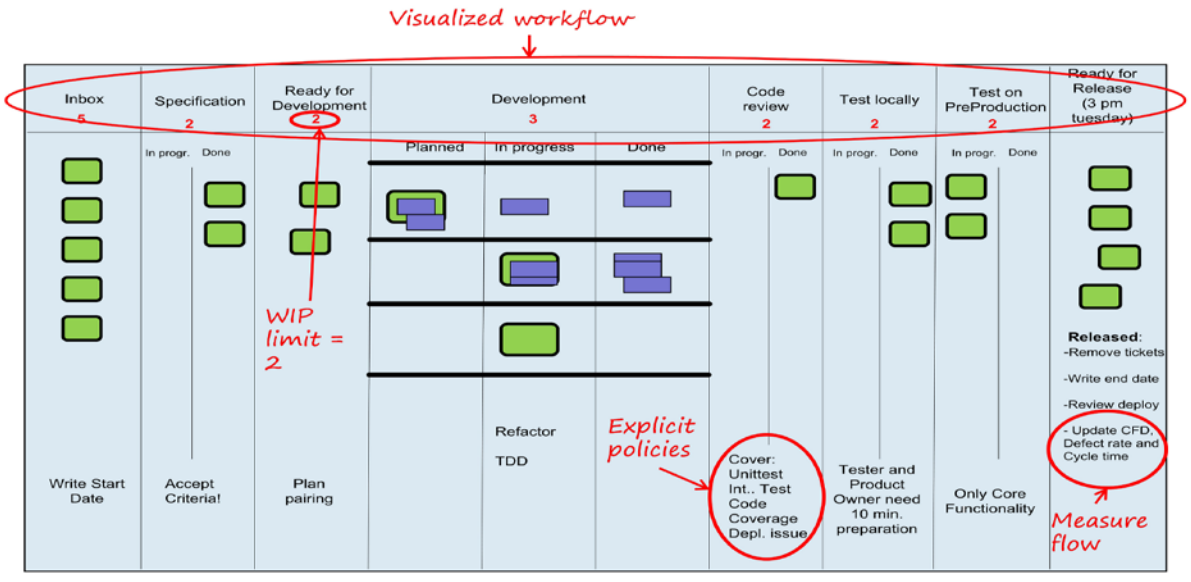

Figure 1. Kanban board and principles in action (adapted from Boeg [11]).

The literature describes various benefits of using Kanban in a software development project [2][5-7][12]. For instance, Kanban helps users understand project activities, helps them easily identify and solve problems in team work and promotes collaboration in work. Kanban aims to provide visibility in work through its visualisation principle.

In a software development project, all the work items are displayed on the Kanban board, which makes them visible all the time to stakeholders. The development team and customers see the real-time progress of the work and get a better project view. If a work item on the board is not moving forward, it signals to the team that a member is having problems in his or her work. Whenever a team member is stuck in a task, the other team members can easily see it on the Kanban board. Then, the team finds a solution for it collaboratively, and the complex problem is solved efficiently. This promotes collaboration and helps the team identify bottlenecks in the project or in individual team member tasks. The Kanban WIP principle allows only a limited number of tasks to be worked on at a given time, which helps in the management of tasks. Thus, Kanban helps identify high-priority tasks.

At the Master's level, hands-on software project courses should enhance students' software design skills and familiarise them with up-to-date approaches in software engineering. Along with technical skills, students are also expected to learn other essential software process related skills and competences [4]. Various approaches are taught to software engineering graduate students, such as Agile and Lean.

Based on the literature review on Kanban in software engineering, one can expect that Kanban will also be an efficient tool in software engineering education. It can potentially help in the acquisition of various software project and team work competencies needed in professional software engineering, such as positive relationships building among team members, sharing responsibilities and negotiating with other groups [4].

RESEARCH DESIGN

The SWF aims to teach software engineering students relevant software engineering knowledge and skills for their future work. Learning software project skills is challenging. Various aspects, such as understanding the project and its activities comprehensively are difficult for inexperienced people. Kanban has certain features that make it suitable to the teaching of software engineering, in addition to its usage in professional software projects.

Kanban was chosen for use in the SWF for several reasons. First, since Kanban is increasingly being adopted in software companies, it should be taught to the students so they will be equipped with the knowledge of new methods and more prepared for the labour market. Second, Kanban has several characteristics that can help improve students' understanding of essential knowledge about software processes and projects, which they will need when they work with real software projects in companies. Third, using Kanban can help students acquire various competencies required in software projects. Kanban is also a relatively light method that should be easy to learn and use in software development. Therefore, the novices, as students, should be able to adopt it easily.

To evaluate Kanban in SWF teaching, the authors conducted a survey with students who used Kanban in their SWF projects. The survey focused on the following research questions:

- Does Kanban help students understand the essential aspects of a software project?
- Are students willing to use Kanban in their future professional software projects after they have experienced using it?
- Does a Kanban project in SWF help students acquire team work competencies?

The study participants were final year Master's degree students who participated in SWF Kanban projects in the 2013 academic year and spring 2014 semester. During that period, there were 24 Kanban projects, in which 96 students took part. A request to answer the on-line survey was sent to all the students after the completion of each project. The survey was open for two weeks. During that time, 52 responses were received. One of the responses was discarded because of incorrect and missing information. This led to a total of 51 responses, which formed the data for the analysis.

The survey questionnaire was designed to collect the students' perceptions and experiences of using Kanban in their SWF projects. The survey consisted of Likert-type scale questions with a scale of 1 (strongly disagree) to 5 (strongly agree). The questions asked the students about their perceptions of Kanban usage in project work and were complemented with open-ended questions.

The questions included the following statements: *Kanban boards in SWF help me understand my project activities; I believe that the visualisation of work via Kanban boards provides a better project view; I believe that Kanban boards in SWF help identify bottlenecks in my project;* and *I would consider using Kanban boards in a real-life software development context.* In addition, the survey included Likert-type scale questions regarding the competencies that the students gained during their project work. The competencies covered in the questionnaire were complex problem solving, responsibility sharing, shared vision development, positive relationship building, negotiation skills, rational argumentation and conflict resolution.

RESULTS

The first research question assessed whether Kanban is a good method for helping students understand the essential aspects of a software project. The students' responses to this question are presented in Figure 2.

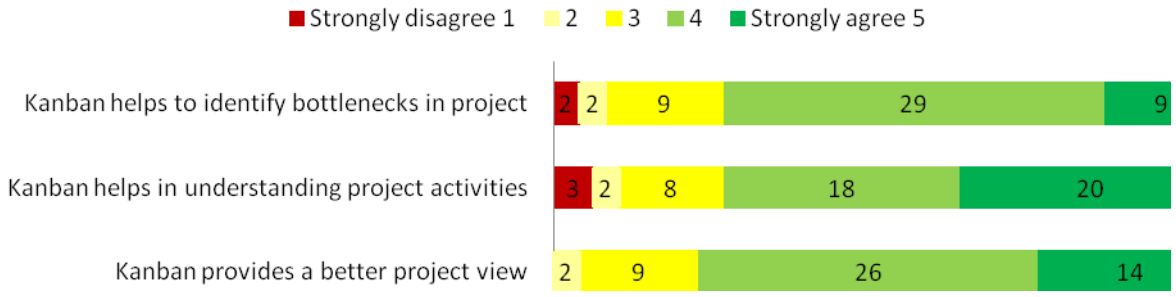

Figure 2. Students' perception of Kanban.

Figure 2 shows that the students had very positive perceptions of Kanban, particularly its ability to help students understand software projects. The majority of students stated that Kanban helped them understand project activities, provided a better view of the project and helped them identify bottlenecks in the project. Very few of them did not consider it helpful. Kanban appears to be useful for inexperienced software developers, such as students.

Visualisation is one of the key principles of Kanban; it is used to get a bigger picture and better understanding of the work. By looking at the Kanban board, every team member sees the progress of work for each task. This shows what needs to be done and what has not yet been done. In response to the open-ended question, the students expressed their views about the usefulness of Kanban as it is *Useful for the project group for visualising the state of the project during the project. Visualisation of different task types or priorities also helps present a more detailed overall picture. One can see the progress of the project at one glance*.

One student explained that Kanban helps the students track the progress of project activities that *...Kanban assisted us in knowing at what stage we were in our project, and who should do what, at any point in time. It was helpful to figure out what we were doing and what we shall be doing in the future. The timing for internal deadlines among project members was also aided by the use of board. It clarifies the progress of a particular task and the project as a whole.* Another student mentioned, *...I think it is very helpful because we have a better vision of the* atomic tasks *that we need to do and we see the progress of the group.* This shows that students understand the complexity of the problems and can break them down to a more easily solvable level.

The students working in two geographically distributed teams (one in Oulu, Finland and the other in Bolzano, Italy) reported the benefits of using Kanban in their work. They explained that they used both Kanban boards simultaneously in SWF and JIRA (JIRA is a tracker for teams work), creating good visualisation in Oulu and in Bolzano for their customers. One student said that Kanban helps in *...visualisation of different task types or priorities and helps present a more detailed overall picture*. Another student said that *...the Kanban board helps me follow my project procedure when I am lost*. Using Kanban made the problems visible as soon as they occurred.

The respondents recognised that Kanban helps them to identify bottlenecks in projects. As one of the students stated, *The Kanban board is a very good tool for task management. The boards were good to use for discussion in my project. It was good to use, we discussed the problems of our project when they occurred. It serves us well and we use it to analyse the task at hand. It is also good when work is not going forward due to some problem*. Furthermore, a student believes that *...Kanban board helps me to follow my project procedure when I am lost. Kanban should be used more in the school, which will be a good help for students to control their ongoing project or works*. Such positive responses show that Kanban helps students understand project activities quite easily, which supports their learning of software development.

The experienced usefulness of a learned method can be evaluated based on the person's intention to use it in the future. Figure 3 presents the survey results on the students' willingness to use Kanban in real software engineering projects.

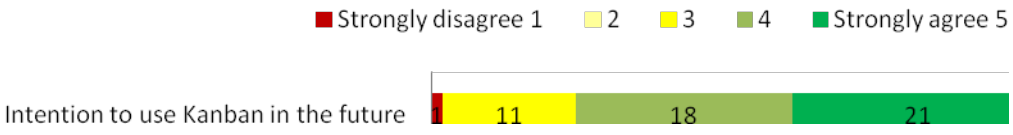

Figure 3. Students' intention to use Kanban.

Figure 4 presents the survey results on whether the Kanban project in the SWF teaches students the team work capabilities required for software projects.

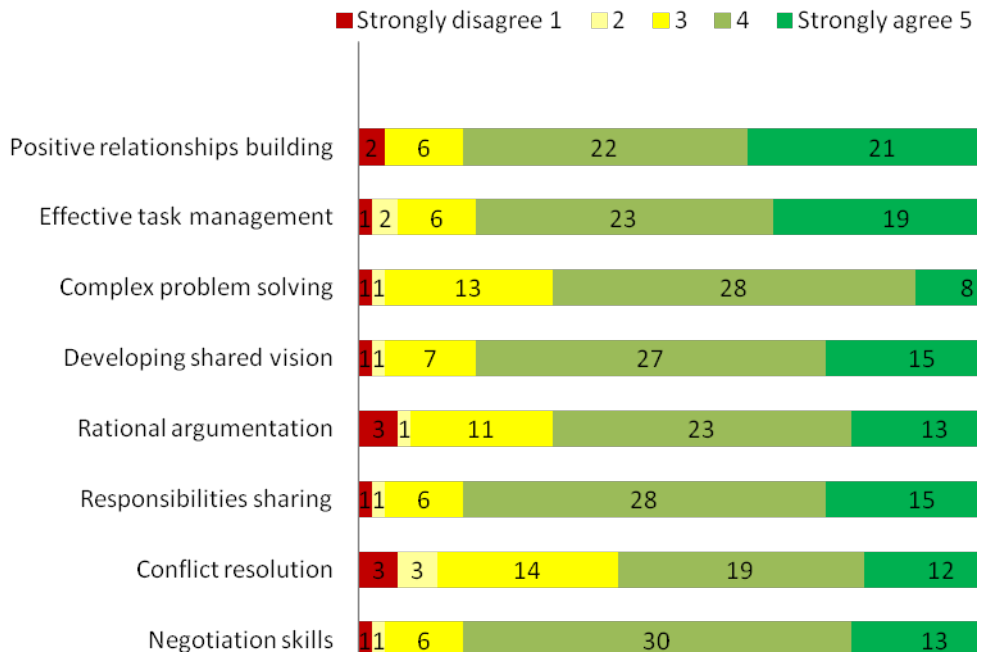

Figure 4. Learning of team work competencies.

The results show that overall, students' group work experiences, learning and competency acquisition were positive in the Kanban project in the SWF learning environment. The students felt that they obtained relevant skills and competencies, such as negotiating with other group members, building positive relationships among team members, effective task management, development of a shared vision and responsibility sharing. These experiences show that the students learned to work collaboratively in the SWF Kanban project, which helped them generate ideas, share their views and solve complex problems. These results show that teaching software engineering in the SWF using Kanban is effective, as the students are able to acquire and exercise the skills needed for team work in the challenging software industry.

Overall, the perception of students regarding Kanban was positive. Very few of them had negative responses. For example, one student reported that *...Kanban is good but we need more lectures on how to use it.* This indicates that proper training and education regarding Kanban is an antecedent for its efficient usage, as reported in earlier studies [4][12].

In summary, Kanban is a good pedagogic tool for teaching software engineering in hands-on software development project courses, such as SWF projects. It helps students understand software projects and current industrial practices. Kanban also facilitates project development, promotes team communication and supports collaboration. By learning Kanban and the skills and capabilities it promotes, software engineering students become better prepared to enter their software development careers. They develop a more complete understanding of software development processes and gain an appreciation of the effectiveness of modern software engineering methods, such as Kanban.

CONCLUSIONS

Software factory projects involving real-life projects from industry clients provide software engineering graduates with the necessary skills for their industrial careers. Kanban has been adopted in the University of Oulu's SWF projects to equip students with a well-rounded knowledge of technical and non-technical skills required to thrive in industry. This motivated the authors to carry out this study to provide insights on students' perceptions of Kanban usage in their software development projects in the SWF and the competencies gained in the SWF projects.

This research was based on the existing literature on Kanban in software development and team work in software engineering. The results show that Kanban helps students understand the essential aspects of a software project and successfully acquire the challenging software engineering project skills required by professional industrial software engineer work.

The students stated that the usage of Kanban in their SWF projects helped them understand project activities and easily obtain various team work competencies, such as building positive relationships both inside and outside the team, building a shared vision and effective task management. In the SWF, using Kanban and interacting with real clients deepen the student experience of the software projects and allow them to understand the importance of the so-called soft

skill of team work. The majority of students expressed positive views about Kanban in their project work and appreciated its value as part of their university education. The students applied Kanban principles in their project work and perceived increasing success in the outcomes.

Kanban is a good pedagogic tool in teaching software engineering to inexperienced software engineers. With relatively little training, students can easily understand the overall software process and participate in a team for solving challenging software design problems. The study also shows that students are willing to use Kanban in the future, which is good sign for the software companies who are increasingly using it to support their Lean approaches. As Kanban has been shown to be easy to learn and use even for inexperienced persons, it should have a low adoption threshold and short learning curve for all software engineering personnel.

These study results reinforce the authors' expectations that the SWF Kanban project course promotes successful software development and provides an environment that encourages learning and acquiring the necessary knowledge and skills for professional software engineering. The authors see a high value in bringing Kanban principles to hands-on software project learning activities. The study confirmed that the students find Kanban relevant to their learning. The SWF, as a hands-on learning environment with real-life software projects from the industry, was also found to be an excellent way to teach Kanban to the students, as they acquired understanding and practical knowledge of it rather easily and quickly in their Kanban projects.

REFERENCE

1. Liker, J., *The Toyota Way*. New York: McGraw-Hill (2004).
2. Ahmad, M.O., Markkula, J. and Oivo, M., Kanban in software development: a systematic literature review. *Proc. 39th Euromicro Conf. on Software Engng. and Advanced Applications*, Santander, Spain, 9-16 (2013).
3. Anderson, D., *Kanban: Successful Evolutionary Change for Your Technology Business*. Sequim, Washington: Blue Hole Press (2010).
4. Ahmad, M.O., Liukkunen, K. and Markkula, J., Student perceptions and attitudes towards the software factory as a learning environment. *Proc. IEEE Global Engng. Educ. Conf., 2014*, Istanbul, Turkey, 422-428 (2014).
5. Senapathi, M., Middleton, P. and Evans, G., Factors affecting effectiveness of agile usage - insights from the BBC Worldwide case study. *Proc. 12th Inter. Conf., XP 2011*, Madrid, Spain, 132-145 (2011).
6. Ikonen, M., Pirinen, E., Fagerholm, F., Kettunen, P. and Abrahamsson, P., On the impact of Kanban on software project work: an empirical case study investigation. *Proc. 16th IEEE Inter. Conf. on Engng. of Complex Computer Systems*, Las Vegas, NV, USA, 305-314 (2011).
7. Nikitina, N. and Kajko-Mattsson, M., Developer-driven big-bang process transition from Scrum to Kanban. *Proc. Inter. Conf. on Software and Systems Process*, 159-168 (2011).
8. Schneider, J. and Vasa, R., Agile practices in software development - experiences from student projects. *Proc. 2006 Australian Software Engng. Conf.*, Sydney, Australia (2006).
9. Tomayko, J., *Human Aspects of Software Engineering*. Herndon, VA: Charles River Media (2004).
10. Fagerholm, F., Oza, N. and Münch, J., A platform for teaching applied distributed software development: the ongoing journey of the Helsinki software factory. *Proc. 3rd Inter. Workshop on Collaborative Teaching of Globally Distributed Software Develop.* (CTGDSD 2013), San Francisco, United States, 1-5 (2013).
11. Boeg, J., *Priming Kanban: A 10 Step Guide to Optimizing Flow in Your Software Delivery System.* Aarhus: Trifork (2012).
12. Ahmad, M.O., Markkula, J., Oivo, M. and Kuvaja, P., Usage of Kanban in software companies: an empirical study on motivation, benefits and challenges. *Proc. 9th Inter. Conf. on Software Engng. Advances*, NICE, France (2014).