GLOBAL ACADEMIC RESEARCH INSTITUTE

COLOMBO, SRI LANKA

GARI International Journal of Multidisciplinary Research

ISSN 2659-2193

Volume: 06 | Issue: 05

On 31st December 2020

http://www.research.lk

Author: Dr. Rasika Aponsu

IIC University of Technology, Cambodia

GARI Publisher | Software Engineering | Volume: 06 | Issue: 05

Article ID: IN/GARI/JOU/ 2022/132 | Pages: 94-102 (09)

ISSN 2659-2193 | Edit: GARI Editorial Team

Received: 22.11.2020 | Published: 31.12.2020

SUPPORTING TOOLS FOR THE MANAGEMENT OF SOFTWARE ENGINEERING PROJECTS IN AN ACADEMIC SETTING

Dr. Rasika Aponsu

IIC University of Technology,

Cambodia

ABSTRACT

This paper provides a description of two software tools employed at the University of Ljubljana, located in Ljubljana, Slovenia, with the purpose of facilitating the administration of students' software engineering capstone projects. The initial tool is an internally designed instrument that facilitates the administration of Scrum-oriented projects undertaken inside the capstone course at the undergraduate level. The software application not only encompasses the essential features necessary for effectively managing Scrum projects, but also caters to the distinct requirements of educational personnel and researchers. The second tool under consideration is Kanbanize, a commercially accessible software solution employed for the purpose of effectively managing projects undertaken by students at the second level of the Bologna educational framework. This software application facilitates the fundamental principles of Kanban, including the management of the Kanban board, the restriction of work in progress, and the enhancement of lead time.

Keywords: Software Engineering, Software Project Management, Software Tools

INTRODUCTION

Acquiring proficiency in software engineering (SE) necessitates substantial engagement in practical exercises, enabling students to comprehensively grasp and contemplate SE principles. Hence, it is recommended that the software engineering curriculum incorporates project-based courses that offer a suitable amalgamation of theoretical knowledge and practical application [1]. In accordance with guidelines for undergraduate software engineering curriculum, it is imperative that students partake in a capstone course, spanning a duration of one academic year [2]. The course ought to include previously acquired knowledge, enhance comprehension of said knowledge, broaden the scope of knowledge, and enable the application of knowledge and skills within a realistic simulation of professional practice.

The University of Ljubljana offers a capstone course in software engineering at two academic levels, namely the first and second Bologna levels. The duration of the course is consistent across both levels, spanning a single semester encompassing a total of 15 weeks. In the initial stage of the Bologna process, the course is compulsory for all students enrolled in the SE curriculum. However, at the subsequent stage of the Bologna process, it has been designated as an elective professional course. In addition to the fundamental objectives that all capstone courses should strive for, the capstone courses offered at the University of Ljubljana also function as a means of transmitting agile and lean methodologies to students [3]. The projects undertaken by students at the initial Bologna level adhere strictly to the principles and methodologies of Scrum, whilst projects at the subsequent Bologna level serve as a platform for introducing the lean principles of Kanban.

The incorporation of capstone projects into the curriculum for teaching agile and lean software development offers several benefits. Firstly, it provides students with exposure to cutting-edge topics that are highly relevant in industry. Additionally, it encourages teaching staff to adopt contemporary teaching methods, specifically project-based learning. Lastly, the inclusion of capstone projects enables the opportunity to conduct empirical studies with students, thereby contributing to the body of empirical evidence surrounding agile and lean practices. Given that students engage in collaborative work on projects that simulate real-world scenarios, it is most effective to employ a computerized tool for the purpose of monitoring their progress and gathering data for empirical investigations. The tool should possess the capability to accommodate various development methodologies, such as Scrum and Kanban, respectively. Additionally, it should assist teaching staff in effectively monitoring students' progress, alleviate the administrative workload, and furnish empirical data to facilitate evidence-based appraisal of the development process.

The objective of this article is to provide a comprehensive overview of two distinct tools employed for the administration of students' capstone projects at the University of Ljubljana. The initial tool is an internally built application designed to facilitate the administration of Scrum-oriented projects undertaken as part of the capstone course at the undergraduate level. Another tool that is commonly used for project management at the second Bologna level is Kanbanize, which is accessible for commercial usage. The subsequent portions of this article are organized as follows: the subsequent two sections present the primary attributes of capstone projects based on Scrum methodology and the related tool. This is followed by two sections that outline the features of Kanban projects and the Kanbanize tool, respectively. The final section of the document presents a concluding statement.

Scrum-Based Capstone Projects

The Scrum framework advocates for the utilization of iterative and incremental development methodologies. The execution of tasks is carried out in a series of iterations, commonly referred to as sprints, with the objective of delivering an increment of functionality that is capable of being shipped. In the context of Scrum, three distinct roles may be identified: the product owner, the team, and the Scrum master. The role of the product owner encompasses the representation of the various stakeholders involved in the project and their respective interests in relation to the project's outcomes. The individual in question keeps a meticulously organized inventory of essential criteria, referred to as the product backlog, typically comprising a collection of user stories. The implementation of the required functionality is under the responsibility of the team, whilst the Scrum master is responsible for ensuring adherence to Scrum rules and practices.

The capstone projects that follow the Scrum methodology consist of a total of four sprints [4]. The initial sprint, commonly referred to as sprint 0, spans a duration of three weeks and is designated as a preliminary sprint. During the initial phase of the project, known as sprint 0, students participate in a series of rigorous and structured lectures aimed at instructing them in the principles and practices of Scrum methodology. These lectures serve the purpose of familiarizing the students with the concept of user stories, which they will subsequently employ in the development process. The narratives are authored and given precedence by a subject matter expert, who assumes the role of the product owner, either as the instructor or a representative from a collaborating organisation. The students are organised into teams consisting of four members each, with the primary responsibility of developing the necessary functionality. The estimating of stories is conducted by each team through the utilisation of planning poker or the team estimation game, followed by the development of a release plan. The remaining portion of the course is structured into three consecutive Scrum sprints. The commencement of each sprint is marked by a sprint planning meeting, during which student teams establish the scope of the upcoming iteration and formulate the initial rendition of the sprint backlog.

Throughout the sprint duration, it is imperative for teams to consistently convene at the daily Scrum meetings and diligently manage their sprint backlogs. Furthermore, with the culmination of each sprint, the sprint review and sprint retrospective sessions are conducted. During the review session, students deliver their findings to the instructors, whereas the retrospective meeting serves as a platform for students and instructors to collectively assess the progress made during the preceding sprint and provide recommendations for enhancing future iterations. Upon the completion of three consecutive sprints, the initial release shall be deemed finished and thereafter provided to the customer. It is mandatory for students to furnish data pertaining to their original effort estimations, the quantity of work completed, and the quantity of work yet to be completed. At the commencement of each sprint, participants are advised to reassess their velocity and the outstanding user stories with the aim of formulating a more pragmatic plan for subsequent iterations. Instructors assess the alignment between students' projected outcomes and their actual accomplishments, evaluating the extent to which students' capacity for estimation and planning improves as they acquire greater familiarity with Scrum methodology and a deeper comprehension of user requirements. Considerable emphasis is placed on the concept of completion, necessitating that students elevate their code to a practical, real-world standard capable of withstanding interactions with end users. The data produced during the course serves the dual purpose of providing feedback on students' learning outcomes and contributing to the evidence-based assessment of Scrum procedures and practises. The initial focus of the research was on examining the students' perspectives regarding Scrum [5]. Subsequently, the course was utilised as a case study to explore the application of agile estimating and planning through the use of Scrum [6]. Additionally, the course facilitated the collection of data to assess the precision of planning poker estimates [7].

Tool Support for Scrum-Based Capstone Projects

The impetus behind the development of a tool for managing Scrum-based capstone projects was derived from the findings of a study [8], which revealed that current agile solutions suffer from either a lack of comprehensive capability or excessive complexity and usability challenges. Furthermore, the survey revealed that the current technologies pose challenges when it comes to customization in order to meet the unique requirements of end users.

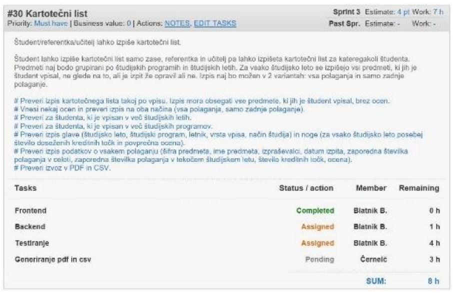
Sprint 0

In sprint 0, the tool accomplishes three main tasks. Firstly, it automates the process of forming student teams and assigning them to suitable parts of laboratory sessions. Secondly, it offers functionalities for the first development of the product backlog. Lastly, it automates the process of estimating the work required for the project.

In accordance with the Scrum principle of self-organisation, students are afforded the chance to exercise agency in determining their preferred collaborators. Upon the establishment of a team, students employ a designated tool to input the composition of said team, together with their preferred laboratory class section and the project they intend to undertake. The selection of the project and the preferred laboratory class section is determined according to the concept of first-come, first-served. Concurrently with the process of student team building, the individual from the teaching staff (or the representative from a company) assuming the position of the product owner utilises a tool to create the initial product backlog. This backlog comprises a collection of user stories. As depicted in the upper portion of Figure 1, the representation of each user narrative consists of a story card that encompasses a textual description, serving the purpose of planning and serving as a reminder for further discussions. Additionally, the story card includes a collection of acceptance tests, which are employed to ascertain the completion of a story. The responsibility of determining the priority and business value is also assigned to the product owner. Upon the conclusion of sprint 0, the tool facilitates the ability of student teams to estimate the level of effort necessary for the implementation of each user story. The application facilitates the automation of two estimate procedures, specifically planning poker and the team estimation game. Both methodologies necessitate the estimation of user stories in terms of story points. The tool facilitates this process by enabling each team member to contribute their estimate. This may be done by selecting from a set of predefined values (such as 0.5, 1, 2, 3, 5, 8, 13, 20, 40) or by defining a custom number.

Sprints 1, 2 and 3

The initiation of each regular sprint commences with the convening of a sprint planning meeting. During this meeting, the user stories that the team has committed to implementing are identified by selecting the correct tick box and subsequently transferred to the sprint backlog with a single click. Subsequently, the narratives contained inside the sprint backlog are broken down into individual tasks, with an accompanying estimation of the effort necessary for each job. Ultimately, the assignments of duties are allocated to individual members of the team. The application facilitates the documentation of the time and effort expended on each activity, as well as the quantification of the remaining workload. The lower portion of Figure 1 illustrates an instance of decomposing a user story into individual tasks. Each task is accompanied with its current status, the person responsible for its completion, and the remaining workload.



#30 Kartotečni list

Priority: Must have | Business value: 0 | Actions: NOTES, EDIT TASKS

Sprint 3 Estimate: 4 pt Work: 7 h
Past Spr. Estimate: - Work: -

Študent/referentka/učitelj lahko izpiše kartotečni list.

Študent lahko izpiše kartotečni list samo zase, referentka in učitelj pa lahko izpišeta kartotečni list za kateregakoli študenta. Predmeti naj bodo grupirani po študijskih programih in študijskih letih. Za vsako študijsko leto se izpišejo vsi predmeti, ki jih je študent vpisal, ne glede na to, ali je izpit že opravil ali ne. Izpis naj bo možen v 2 variantah: vsa polaganja in samo zadnje polaganje.

Preveri izpis kartotečnega lista takoj po vpisu. Izpis mora obsegati vse predmete, ki jih je študent vpisal, brez ocen.
Vnesi nekaj ocen in preveri izpis na oba načina (vsa polaganja, samo zadnje polaganje).
Preveri za študenta, ki je vpisan v več študijskih letih.
Preveri za študenta, ki je vpisan v več študijskih programov.
Preveri izpis glave (študijsko leto, študijski program, letnik, vrsta vpisa, način študija) in noge (za vsako študijsko leto posebej število doseženih kreditnih točk in povprečna ocena).
Preveri izpis podatkov o vsakem polaganju (šifra predmeta, ime predmeta, izpraševalci, datum izpita, zaporedna številka polaganja v celoti, zaporedna številka polaganja v tekočem študijskem letu, število kreditnih točk, ocena).
Preveri izvoz v PDF in CSV.

Tasks	Status / action	Member	Remaining
Frontend	Completed	Blatnik B.	0 h
Backend	Assigned	Blatnik B.	1 h
Testiranje	Assigned	Blatnik B.	4 h
Generiranje pdf in csv	Pending	Černelc	3 h
		SUM:	8 h

Figure 1: This illustrates an exemplar of a user story card. The depicted section of the diagram illustrates the user story in its original form, as it is articulated within the product backlog. The lower section of the document displays the tasks that have been incorporated by the team in the course of the sprint planning meeting.

The availability of data regarding the remaining work enables the monitoring of work progress through the utilisation of burn down charts. Additionally, the

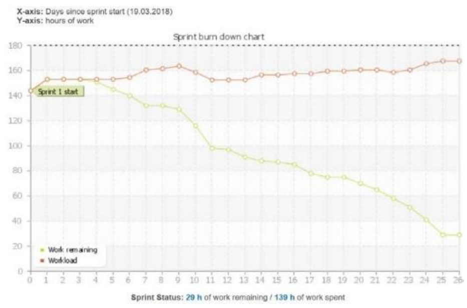
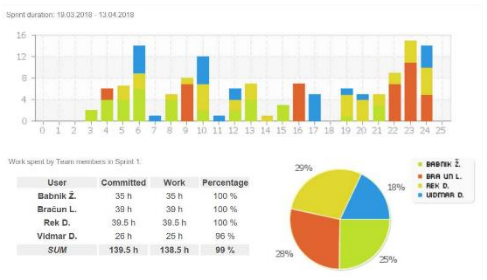


Figure 2: Sprint Burndown Chart

In order to ensure the efficient operation of the course, it is imperative that student teams engage in their project work without succumbing to procrastination, while also ensuring an equitable distribution of burden among all team members. Figure 3 depicts the team engagement chart, a tool utilised by both students and teachers to effectively monitor and assess these two dimensions. The upper portion of the graphic displays a stacked bar chart that illustrates the quantity of work completed by individual students on each day of the sprint. The lowest section of the document displays the distribution of workload, utilising both a tabular format and a pie chart.

quantification of the work completed facilitates the tracking of individual student contributions and the examination of disparities between estimated and actual effort. The tool offers stacked burn down charts at two distinct levels, namely the sprint level and the release level. Both figures depict the relationship between the quantity of unfinished tasks and the advancement of a student team in their efforts to diminish this workload. Figure 2 presents an illustration of a sprint burndown chart.



Work spent by Team members in Sprint 1:

User	Committed	Work	Percentage
Babnik Ž.	35 h	35 h	100 %
Bračun L.	39 h	39 h	100 %
Rek D.	39.5 h	39.5 h	100 %
Vidmar D.	26 h	25 h	96 %
SUM	139.5 h	138.5 h	99 %

Figure 3: This displays a chart illustrating the level of team involvement. The upper portion of the diagram illustrates the allocation of work completed by individual team members on a daily basis during the sprint. The lower section of the document presents the individual contributions of each member, shown both in tabular and graphical formats.

Additionally, the application offers insights regarding the frequency of daily Scrum meetings held by all teams, as well as the tracking of work completed and work remaining. Students who neglect to furnish records of daily Scrum meetings or data pertaining to the time allocated for work completed and work still to be completed are promptly notified through electronic mail. Upon the conclusion of the sprint, the product owner designates the user stories that satisfy the acceptance criteria as completed, whilst incomplete stories are automatically relocated to the product backlog for potential inclusion in subsequent sprints. A comment may be included to each tale that has been rejected, providing a rationale for its rejection.

End of the Project

Upon the completion of the project, the tool facilitates the assessment of students' projects by generating a comprehensive report on their implementation. Based on the data gathered throughout the project, the tool autonomously calculates the individual and collective amount of story points attained by each student and the entire team. In instances where multiple students are collaborating on a shared user narrative, the allocation of story points is distributed proportionally based on the individual efforts put by each student during the implementation process.

Kanban-Based Capstone Projects

Kanban-based capstone projects, while maintaining the fundamental qualities of Scrum-based projects, also incorporate the lean principles of Kanban [9]. Considerable emphasis is placed on matters that appear to hold significant importance when implementing Kanban in the context of software development. These include the configuration of the Kanban board, the establishment of work-in-progress (WIP) restrictions, and the measurement of lead time. Furthermore, this study examines various methods of implementing Kanban, including a gradual implementation through Scrumban or a pure Kanban strategy that involves forsaking certain Scrum practises. Due to this rationale, student teams are segregated into two distinct categories: the Scrumban group and the Kanban group.

The teams that fall under the Scrumban group, henceforth referred to as Scrumban teams, adhere to the Scrum principle of utilising iterations with predetermined durations. As a result, their projects continue to include sprint 0 and three standard Scrum sprints, but are enhanced through the utilisation of the Kanban board and work-in-progress (WIP) constraints. The board serves as a means of visualising the workflow, enabling team members to gain a clear understanding of the various stages and progress of work items. Additionally, the implementation of Work-in-Progress (WIP) restrictions acts as a mechanism to restrict team members from concurrently working on several work items, hence reducing lead time. The Kanban board include the sprint backlog column, which is established at each sprint planning session. It is imperative that the items included in the sprint backlog do not surpass the projected velocity. The teams affiliated with the Kanban group, hereafter referred to as Kanban teams, adhere more rigorously to lean principles by eschewing fixed-duration iterations and sprint planning. The obligation to uphold the sprint backlog and monitor velocity is no longer in effect for them. In contrast, the product owner is responsible for managing a limited set of highly prioritised stories. These stories can be selected by a team member for development once they have successfully completed their previous user story assignment. To maintain a seamless process, it is imperative for the product owner to swiftly assess user stories upon their completion notification. As a result, the review sessions are not scheduled at fixed periods, but rather occur in response to specific events. A review meeting is scheduled when a predetermined set of minimum marketable features (MMF), as specified by the product owner, is deemed ready for release.

No discernible distinctions exist between the groups in terms of their adherence to daily Scrum and sprint retrospective sessions. The retrospective meetings of Kanban teams are conducted with regularity, following the same intervals as Scrumban teams, namely at the conclusion of each Scrumban sprint. Additionally, all teams are obligated to convene at the daily Scrum meetings on a regular basis.

Tool Support for Kanban-Based Capstone Projects

The implementation of the course is facilitated by Kanbanize, a proprietary project management software specifically designed for the administration of projects utilising the Kanban methodology. The utilisation of Kanbanize facilitates the prioritisation of key Kanban principles, namely process visibility, work in progress limitation, and lead time measurement. To facilitate the visualisation of the workflow, the product backlog's user stories are depicted as cards on the Kanban board. These cards are organised in a sequential manner across columns, each column representing a distinct state that a work item can occupy during the development process. Each individual student team is responsible for managing their own board, where they track the progress of their work throughout the development lifecycle by moving cards from one state to another, until they ultimately reach the final column. The utilisation of Kanbanize facilitates the creation of a Kanban board that possesses a flexible and customizable structure. Therefore, individuals have the ability to determine the most appropriate columns for their own development process, with the option to further categorise each column into sub-columns and sub-sub-columns. A potential practise in workflow management is assigning a Work-in-Progress (WIP) limit to each column, denoting the maximum number of cards permissible in a certain workflow state concurrently.

The board depicted in Figure 4 serves as an illustrative example of the aforementioned concept. It should be noted that certain columns in Kanbanize may appear minimised owing to spatial constraints. However, users have the option to expand these columns to view them in their complete form

.

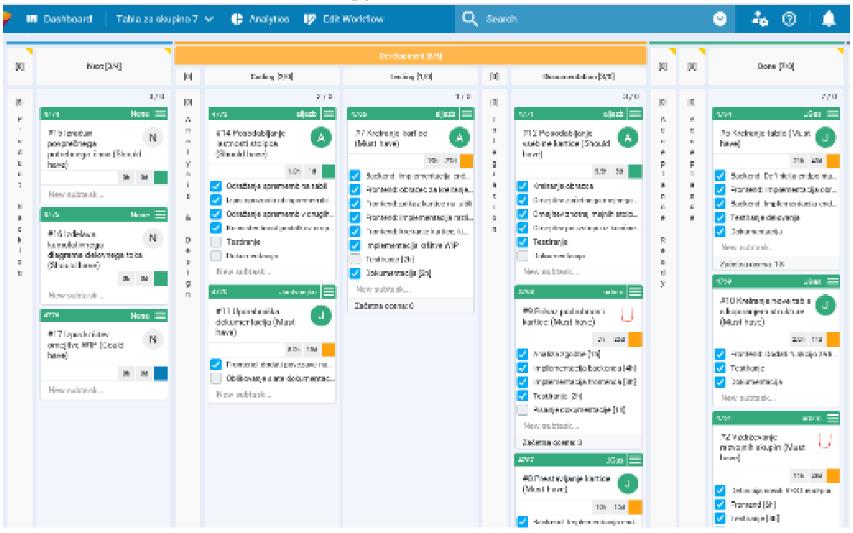


Figure 4: Kanban Board used by a Team belonging to the Kanban Group

When a novel user narrative is generated, the related card is positioned within the column designated as the product backlog. The subsequent column is designed to accommodate a restricted quantity of stories that hold a high level of importance, as prioritised by the product owner for initial implementation. The activities conducted by student teams are consolidated under the development category, which is subsequently subdivided into sub-categories such as analysis & design, coding, testing, integration, and documentation. The acceptance ready column functions as an intermediary space that separates the development team from the product owner. Upon the completion of a user story by a student team, the relevant card is transferred to the acceptance ready column. This action serves as an indication for the product owner to commence the process of acceptance testing. Subsequently, the product owner proceeds to transfer the card to the acceptance column. Upon successful completion of all acceptance tests, the product owner proceeds to relocate the card to the final column denoting completion.

The utilisation of Kanbanize analytics enables student teams to acquire a cumulative flow diagram, calculate the lead time, and juxtapose the duration that a card remained in each column with the actual effort expended in said column. The aforementioned statistics must be presented during every retrospective meeting and function as indicators of process efficacy. The diagram presented in Figure 5 illustrates a cumulative flow diagram as an exemplification. The quantity of cards in each column is exhibited on a daily basis. The breadth of the region corresponding to each board column represents the duration for which a card was in that column, but the gradient of the region associated with the done column signifies the velocity, namely the number of items completed every day.

Figure 6 illustrates the computation of lead time. The bars in the graph symbolise user tales, while the values displayed at the top indicate the lead time measured in days. In addition to the aggregate lead time, it is also feasible to acquire the duration that a card occupied in each respective column.

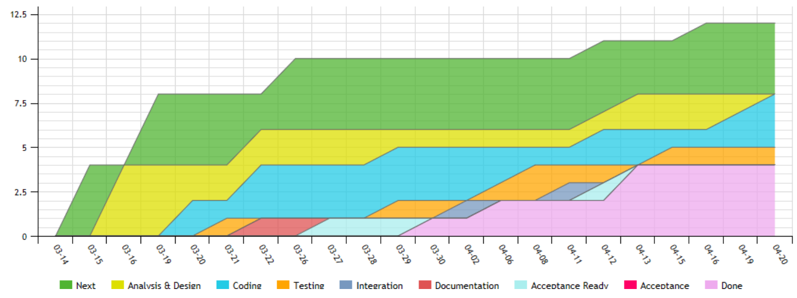


Figure 5: Cumulative Flow Diagram

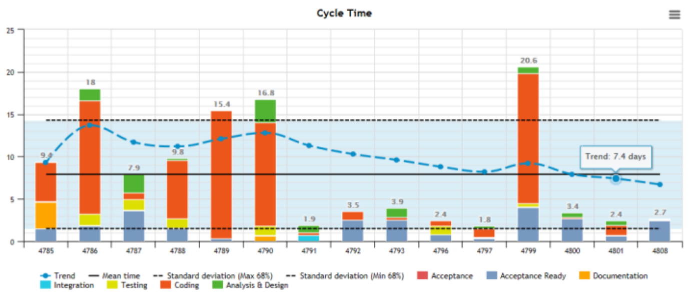


Figure 6: Lead Time Calculation

CONCLUSIONS

The presentation showcased two software technologies that facilitate the administration of students' software engineering capstone projects. The initial tool was created at the University of Ljubljana and provides support for projects utilising the Scrum methodology. The second tool is a commercially available software solution designed specifically for the purpose of effectively managing projects utilising the Kanban methodology. Both instruments have demonstrated their use and have been widely embraced by students.

REFERENCES

Borman, D., Should software engineering projects be the backbone or the tail of computing curricula? Proc. 23rd Conf. Software Engng. Educ. & Training, Pittsburgh, PA, USA, 153-156 (2010).

Joint Task Force on Computing Curricula, Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. IEEE CS and ACM (2015), 17 May 2018, https://www.computer.org/ cms/peb/docs/se2014.pdf

Mahnič, V., The capstone course as a means for teaching agile software development through project-based learning. World Trans. on Engng. and Technol. Educ., 13, 3, 225-230 (2015).

Mahnič, V., A capstone course on agile software development using Scrum. Proc. IEEE Trans. on Educ., 55, 1, 99-106 (2012).

Mahnič, V., Teaching Scrum through team-project work: students' perceptions and teacher's observations. Inter. J. of Engng. Educ., 26, 1, 96-110 (2010).

Mahnič, V., A case study on agile estimating and planning using Scrum. Electronics and Electrical Engng., 5, 111, 123-128 (2011).

Mahnič, V. and Hovelja, T., On using planning poker for estimating user stories. J. of Systems and Software, 85, 9, 2086-2095 (2012).