ELECTRONICS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 2011. No. 5(111)
ELEKTRONIKA IR ELEKTROTECHNIKA

EDUCATION IN ELECTRONICS AND ELECTRICAL ENGINEERING
T 000
STUDIJOS ELEKTRONIKOJE IR ELETROTECHNIKOJE

A Case Study on Agile Estimating and Planning using Scrum

V. Mahnic
University of Ljubljana, Faculty of Computer and Information Science,
Trzaska 25, 1000 Ljubljana, phone: +386 1 4768 447, e-mail: viljan.mahnic@fri.uni-lj.si

Introduction

Introducing agile methods into their development process represents an important challenge for many software companies. In the last few years several successful implementations of agile methods have been reported in the literature, e.g., [1–4]. According to Agile Adoption Rate Survey [5] performed by Dr. Dobbs Journal in 2008 agile teams report significant improvements in productivity, quality, and stakeholder satisfaction, and reasonable improvements in cost. A similar survey conducted by VersionOne [6] additionally reports enhanced ability to manage changing priorities and significantly improved project visibility. For this reason, agile methods are especially suitable for development of information systems with changing and emergent user requirements, e.g., [7]. On the other hand, the same survey has revealed that the lack of experience with agile methods and the conflict between the company's culture and core agile values are the leading causes of failed agile projects.

In spite of the fact that Scrum [8, 9] is the most widespread method in industry (according to [6] Scrum is used by 58% of respondents, Scrum/Extreme Programming hybrid by 17%, custom hybrid by 5%, Extreme Programming by 4%, etc.), a systematic review of empirical studies on agile software development [10] found only one study investigating Scrum. Consequently, one of the clear findings was that the coverage of the research area should be increased placing more focus on management-oriented approaches such as Scrum, which Dingsøyr et al. [11] consider an example of an area where there is a large gap and should be given priority.

In order to fill this gap empirical studies with students as subjects can be helpful in further assessment of the applicability of Scrum before it is actually deployed in industrial software environments. A properly designed study [12] can provide preliminary evidence about its strengths and weaknesses, thus reducing risks accompanying its adoption in practice.

In this paper we describe a case study that was conducted at the University of Ljubljana with the aim of studying the behavior of development teams using Scrum for the first time, i.e., a situation typical for software companies preparing to introduce Scrum into their development process. Within the framework of the capstone course in software engineering, which (as recommended by [13]) students take in their last semester 13 student teams were required to develop an almost real project strictly using Scrum. The data on project management activities were collected in order to measure the amount of work completed, compliance with the release and iteration plans, ability of effort estimation, etc., thus contributing to evidence-based assessment of the typical Scrum processes for possible use in software engineering practice.

Aims of the study and research questions

The aim of the study was twofold: (1) to analyze development teams' abilities of adopting Scrum concepts (e.g., estimation of user stories, release and iteration planning, concept of a user story being 'done'), and (2) to gather their opinions regarding the importance of particular practices for a successful Scrum project.

Regarding the first aim our hypothesis was that the estimates and plans will be less accurate at the beginning, but will improve from Sprint to Sprint. There is substantial evidence reported in the literature that the expert estimates tend to be over-optimistic [14] and that the planning poker estimation technique used by Scrum does not completely eliminate the over-optimism [15]. Therefore, we expected our study to yield similar results. Considering our previous experience [16] and results of a study on behavior of Scrum teams [17] reporting the problem of unclear completion criteria we also decided to pay special attention to the notion of 'done'. It was agreed that the Product Owner could accept only those stories that were fully tested and robust enough to survive an encounter with end users.

With regard to the second aim a survey was conducted at the end of the study in order to find practices that contribute most to the success of a Scrum project. Practices were rated using a 5-point Likert scale, the grade 1 representing the lowest and the grade 5 the highest level of importance.

The remainder of the paper is organized as follows: In the next two sections we describe the case study design and its results. Then a description of students' opinions regarding the importance of particular Scrum practices follows. Finally, the limitations of the study are discussed that should be considered when applying the results in the industrial environment.

Case study design

The case study was conducted in the Summer term of the Academic Year 2009/10 as a part of the capstone software engineering course that lasted 15 weeks and was taken by 52 students who were divided into 13 groups. Each group played the role of a self-organizing and self-managing Scrum Team responsible for the development of a Web-based student records system covering enrollment, examination applications, examination records, some statistical surveys, and a special module for the maintenance of all data required for the proper functioning of the system (i.e., the maintenance of various code tables, lists of required and optional courses, data about teachers of each course, etc.).

The initial Product Backlog comprised 60 user stories and was the same for all teams. It was prepared by the teacher who had considerable experience in developing the University of Ljubljana student records information system [18, 19], thus being able to play the role of the Product Owner. 55 stories described the required functionality for 4 different user roles (i.e., student records administrative staff, students, teachers, and data administrator), whereas five stories described constraints that had to be obeyed (e.g., the system had to enable remote access to data through the Internet, all outputs should also be printable, etc.). Each story contained a short description and a set of acceptance tests that had to be used to demonstrate that the story had been correctly and fully coded.

The Product Owner divided the stories into 4 groups on the basis of priority. There were 24 'must have', 5 'should have', and 4 'could have' stories required in the first release, which should have been finished by the end of the course. The remaining 27 'won't have this time' stories were specified merely to illustrate the desired functionality in the next release.

At the beginning of the course students were given 12 hours of formal lectures on agile principles, Scrum, and the use of user stories for requirements specification and iteration planning. The first three weeks also served as a preparatory Sprint (Sprint 0) before the start of the project. During Sprint 0 the development environment was prepared and students were given the aforementioned initial Product Backlog.

At the end of Sprint 0 each team was asked to estimate the stories of the first release using planning poker [20] and (considering its estimated velocity) prepare the release plan. A story point was treated as an ideal day of work and the estimates were constrained to specific predefined values of 0.5, 1, 2, 3, 5, 8, 13, and 20 as proposed by Cohn [21]. Initial estimates and release plans of all teams were recorded for further analysis.

The rest of the study consisted of three Sprints, each of them lasting 4 weeks. Strictly following the Scrum method each Sprint started with a Sprint planning meeting at which student teams negotiated the contents of the next iteration with the Product Owner, and developed the initial version of the Sprint Backlog. During the Sprint the teams had to meet regularly at the Daily Scrum meetings and maintain their Sprint Backlogs decomposing the user stories into constituent tasks and assigning responsibility for each task. Each student individually estimated how many hours it will take to accomplish each task he/she had accepted. The instructors did not interfere in the distribution of tasks among team members and the estimation of effort, but merely paid attention that the process ran smoothly and everybody obeyed Scrum rules.

At the end of each Sprint the Sprint review and Sprint retrospective meetings took place. At the Sprint review meeting the students presented results of their work to instructors while at the Sprint retrospective meeting students and instructors met to review the work in the previous Sprint, giving suggestions for improvements in the next one. After three Sprints the first release had to be completed and delivered to the customer.

Since it was impossible to expect students to work on the project every day, two Daily Scrum meetings per week were prescribed, one on Monday and the other on Thursday. At the Daily Scrum meeting each team member had to record the number of hours spent and the amount of work remaining for each task he/she was responsible for. When the team finished a story the Product Owner was asked to evaluate its implementation. The Product Owner strictly enforced the concept of 'done', rejecting all stories that did not conform to user requirements. If the shortcomings were not removed by the end of the Sprint a new story was defined in the Product Backlog requiring the completion of missing features in one of the remaining Sprints.

At the end of each Sprint the actual velocity of each team was computed considering only the stories that were accepted by the Product Owner. The unstarted stories and stories that were either rejected or newly defined by the Product Owner were re-estimated in order to create a more realistic plan for subsequent iterations.

Case study results

Results of the study are presented for each Sprint separately in Tables 1 to 3. Data clearly confirm the hypothesis that the plans are less accurate at the beginning, but improve from iteration to iteration.

In the first Sprint the planned velocity estimates were too optimistic and only one team out of 13 (i.e., team T04) actually completed all functionality committed at the Sprint planning meeting. The actual velocity of all other teams was far behind the planned (mean value 11.00, median 8.00). The teams completed on average only 42% (median 35.71%) of story points planned and spent on average much more than one ideal day of work per story point (mean value 27.86, median 15.88 hrs/story point).

Analysis of results at the Sprint retrospective meeting revealed two important reasons for such a great difference between plans and actual achievement: (1) non-compliance with the concept of 'done' and (2) insufficient communication with the Product Owner on the part of students.

Many stories that teams declared completed were rejected either because of the Product Owner's strict insistence on providing fully tested, integrated and usable code or because they did not fully match the user requirements. Some teams complained that the non-compliance with user requirements was due to user stories not being precise enough in describing all the requirements details instead of being aware that the details should be worked out in conversations with the Product Owner. Therefore, all teams were strongly encouraged to increase the communication with the Product Owner during the subsequent Sprints and submit their user stories for review as soon as they were completed, not waiting till the Sprint review meeting.

Table 1. Planned and actual achievement in Sprint 1

Team	Velocity [Story Points]		Plan fulfillment [%]	Work spent [hours]	Hours worked per Story Point
	Planned	Actual			
T01	35.50	1.50	4.23	240.00	160.00
T02	30.50	11.00	36.07	141.00	12.82
T03	52.00	43.00	82.69	146.50	3.41
T04	13.00	13.00	100.00	74.00	5.69
T05	22.00	12.00	54.55	132.70	11.06
T06	23.00	8.00	34.78	127.00	15.88
T07	36.50	7.50	20.55	242.00	32.27
T08	14.00	5.00	35.71	103.00	20.60
T09	32.00	14.00	43.75	138.00	9.86
T10	25.00	6.00	24.00	126.50	21.08
T11	20.00	13.00	65.00	114.00	8.77
T12	25.00	7.00	28.00	134.50	19.21
T13	12.00	2.00	16.67	83.00	41.50
Mean	26.19	11.00	42.00	138.63	27.86
Median	25.00	8.00	35.71	132.70	15.88

Table 2. Planned and actual achievement in Sprint 2

Team	Velocity [Story Points]		Plan fulfillment [%]	Work spent [hours]	Hours worked per Story Point
	Planned	Actual			
T01	43.00	23.00	53.49	140.00	6.09
T02	46.00	31.00	67.39	255.00	8.23
T03	46.50	46.50	100.00	109.00	2.34
T04	20.00	12.50	62.50	160.00	12.80
T05	36.00	36.00	100.00	180.00	5.00
T06	33.50	23.00	68.66	170.50	7.41
T07	35.50	14.50	40.85	148.00	10.21
T08	36.00	35.00	97.22	181.00	5.17
T09	25.00	20.50	82.00	116.50	5.68
T10	40.00	26.00	65.00	137.50	5.29
T11	33.50	25.50	76.12	199.00	7.80
T12	33.50	21.50	64.18	155.00	7.21
T13	18.50	18.50	100.00	108.00	5.84
Mean	34.38	25.65	75.18	158.42	6.85
Median	35.50	23.00	68.66	155.00	6.09

Table 3. Planned and actual achievement in Sprint 3

Team	Velocity [Story Points]		Plan fulfillment [%]	Work spent [hours]	Hours worked per Story Point
	Planned	Actual			
T01	35.00	35.00	100.00	181.00	5.17
T02	34.50	31.50	91.30	175.00	5.56
T03	7.50	7.50	100.00	29.00	3.87
T04	15.00	14.00	93.33	94.00	6.71
T05	25.50	25.50	100.00	60.50	2.37
T06	36.50	30.50	83.56	116.10	3.81
T07	25.00	15.00	60.00	125.00	8.33
T08	20.00	20.00	100.00	98.00	4.90
T09	24.00	23.00	95.83	111.00	4.83
T10	29.50	22.50	76.27	93.00	4.13
T11	36.00	36.00	100.00	169.00	4.69
T12	23.50	23.50	100.00	106.00	4.51
T13	29.00	27.00	93.10	144.25	5.34
Mean	26.23	23.92	91.80	115.53	4.94
Median	25.50	23.50	95.83	111.00	4.83

Strictly following the aforementioned recommendations the difference between planned and actual achievement diminished significantly in the second Sprint. The actual velocity more than doubled and (in spite of the fact that the planned velocity was unreasonably high) the teams completed on average 75.18% (median 68.66%) of story points planned. They spent on average 6.85 (median 6.09) hours per story point which was almost in line with the concept of a story point being equal to 6 hours of work. The initial problems and learning curves were to a great extent mastered, and those teams that established good co-operation among team members, improved testing and integration, and delivered regularly user stories for evaluation, fulfilled their plans completely.

In the third Sprint the teams estimated their velocity to be approximately the same as in the second Sprint, which proved to be the right decision (mean value 26.23, median 25.50). The actual achievement was very close to the plan (mean value 23.92, median 23.50). The teams completed on average 91.80% (median 95.83%) of story points planned and 5 teams achieved 100%. Two teams (T03 and T05) completed all the stories planned for the first release even before the end of the Sprint. On the other hand, it became evident that the teams that had not established good internal communication remained far behind the plan (e.g., team T07).

The results of the study show that (in spite of over-optimistic and sometimes unrealistic initial estimates) the ability of estimating and planning quickly improves. Most teams were able to define almost accurate Sprint plans after three Sprints. In the third Sprint the velocity stabilized and the actual achievement almost completely matched the plan. Empirical data also show a continued increase of productivity. These findings can be considered when introducing Scrum into industrial software development.

Students' opinions regarding Scrum practices

Students' opinions regarding the importance of particular Scrum practices for a successful project are gathered in Table 4. Each practice was rated using a 5-point Likert scale, the grade 1 indicating the practice was not important and the grade 5 indicating the practice was very important. In order to test the extent to which the students' judgments are consistent, the intra-class correlation coefficient (ICC) was computed using the absolute agreement type of the two-way random effects model. The average measure reliability ICC value was 0.935, indicating that the survey data were reliable enough to be generalized. The one-sample t-test was used to determine how much students' rates deviate from the null hypothesis that their opinions were neutral having the arithmetic mean value of all questions equal to 3. Results in Table 4 show that all hypotheses were rejected; therefore, we can accept the alternative hypothesis that students considered all practices important.

Students rated highest team-work and good communication among team members. Student teams that established good communication and team-work indeed achieved far better results than teams that acted as a group of individuals.

Good communication with Product Owner received the second highest grade which was not a surprise since the Product Owner played a central role in students' projects. Projects' progress to a great extent depended on his timely answers to students' questions and prompt evaluation of user stories.

The concept of 'done' was also rated very highly although we were afraid that the students would perceive the Product Owner's insistence on producing stable and highly reliable code as an unnecessary pedantry. However, it seems that through the project work they recognized that only fully tested code that meets all user requirements can be used in practice.

Table 4. Students' opinions regarding the importance of Scrum practices (N=51)

		Mean	Std. dev.	One-sample t-test (p-value)
1	Team-work and communication among team members	4.82	0.44	< 0.001
2	Good communication with Product Owner	4.72	0.45	< 0.001
3	Concept 'done'	4.52	0.68	< 0.001
4	Clarity of requirements specified in the Product Backlog	4.28	0.67	< 0.001
5	Sprint Review Meetings	4.20	0.76	< 0.001
6	Good ScrumMaster	3.94	0.98	< 0.001
7	Sprint Planning Meetings and maintenance of Sprint Backlog	3.92	0.75	< 0.001
8	Sprint Retrospective Meetings	3.74	0.92	< 0.001
9	Daily Scrum Meetings	3.72	1.03	< 0.001
10	Release planning	3.72	0.86	< 0.001
11	Accurate user story estimation	3.56	0.95	< 0.001
12	Accurate velocity estimation	3.54	0.89	< 0.001

Clarity of requirements specified in the Product Backlog was ranked fourth with an average grade of 4.28 indicating that students consider a well prepared and maintained Product Backlog an important factor affecting the success of the project. During the project students occasionally complained that the user stories should contain a more extensive description. However, we were trying to convince them that the essence of the agile approach is not in writing detailed requirements specifications, but in acquiring missing details through communication with the Product Owner and end users.

The high importance of Sprint review meetings can be deduced from the high grades accorded to communication with the Product Owner and the concept of 'done'. All these practices together enable customers to

experience on-time delivery of increments and obtain frequent feedback on how the product really works.

The role of ScrumMaster was also considered important, but not as much as the role of Product Owner. We think this was because the teacher spent much more time playing the role of Product Owner than being the ScrumMaster. As a ScrumMaster he acted merely as a facilitator giving student teams the freedom to self-manage and self-organize as proposed by Scrum. Although he took care that everybody followed Scrum and obeyed its rules this role was less exposed than the role of Product Owner, thus giving an impression of less importance.

The importance of other Scrum meetings was rated between 3.72 and 3.92 which means that these meeting are also considered important, but less than other Scrum practices. We can attribute a slightly lower grade of these meetings to the fact that students often perceive meetings as an unproductive waste of time.

Planning and estimation practices were rated least important (although still statistically significantly above average), which was somewhat of a surprise since the study paid a lot of attention to story estimation and release and Sprint planning. Although the purpose of agile planning is not to produce exact plans we think that students underestimated the importance of this area. There may be several reasons for such opinions. A previous study on students' perceptions of agile methods [22] has shown that students feel least comfortable with planning activities and have low trust in their estimates. Many students also consider estimating and planning unproductive administrative work not being fully aware of the importance of good estimates and plans.

Limitations of the study

From the standpoint of using the results in industry the main limitation of the study is that it was conducted with students in an academic environment. However, in order to increase the degree of validity every effort was made to simulate an industrial environment as closely as possible. User stories were defined on the basis of a real student records information system used at the University of Ljubljana and the study design strictly followed the checklist for integrating student empirical studies with research and teaching goals [12]. The Product Owner strictly enforced the concept of 'done' requiring students to produce fully tested and integrated code resistant to user errors. The study relied on senior students enrolled in their last semester, thus blurring the line between these students and novice professionals. A previous study [23] has shown that these students perform similarly to industry personnel.

Another possible threat to validity is that students (due to other courses) could not work a normal workday, but met for a Daily Scrum twice a week. Considering the even distribution of the total course workload over 15 weeks each student was required to perform 6-8 hours (i.e., approximately one day) of work between two consecutive Daily Scrum meetings, thus simulating the real workload of a normal workday. The rest of the time the students could use for other academic duties. Regular execution of the Daily Scrum meetings worked fine encouraging students to work consistently rather than procrastinate. However, the 3-4 days interval between the meetings provided some room for reallocation of workload allowing students to work more than 8 hours between the two consecutive meetings, which could lead to an uneven distribution of effort over Sprints and skewed the statistics concerning velocity in extreme case. We noticed such an abuse on the part of the team T03, which reallocated a substantial amount of work from Sprint 3 to Sprint 2 in order to complete the project before the end of the course, but this did not affect significantly the study results.

On the other hand, the results of the study in a great deal depended on the proper role of the Product Owner. A knowledgeable and responsive Product Owner contributed a lot to smooth running of students' projects and consequently to better statistics regarding velocity and ability of planning. A non-responsive and/or not knowledgeable enough Product Owner could cause delays and unproductive working periods.

Conclusions

Empirical studies with students as subjects can help industry in providing evidence-driven assessment of new processes, methods, and tools before their introduction in software engineering practice. While most software companies cannot afford extensive experiments, it is not a problem to conduct a study with several teams working on an almost real project within the framework of a software engineering capstone course. In this paper we described an example of such a study that concentrated on (1) the assessment of abilities of estimating and planning when using Scrum for the first time, and (2) gathering students' opinions regarding the importance of particular Scrum practices.

Results of the study have shown that the beginners are able to almost completely grasp Scrum's benefits after a couple of Sprints. Their ability of estimating and planning improved from Sprint to Sprint and after three Sprints almost all teams were able to define accurate iteration plans. The velocity also constantly grew, thus indicating the improvement in productivity.

The study has also revealed the importance of the role of Product Owner. Since the user stories serve merely as a remainder for conversation all user requirements details should be clarified in communication with the Product Owner. In order to assure smooth running of a Scrum project it is important that the Product Owner provides timely answers to questions regarding details of user stories, and makes quick evaluations of work completed strictly enforcing the concept of a user story being 'done'.

Students were overwhelmingly positive about the course because it enabled them to learn agile methods using project oriented approach, which also proved to be successful in other areas of engineering, e.g., [24, 25].

Acknowledgement

The author is grateful to teaching assistants Luka Fürst and Tomaz Hovelja for their help in conducting the course and analyzing data concerning student teams' performance.

References

1. **Schatz B., Abdelshafi I.** Primavera Gets Agile: A Successful Transition to Agile Development // IEEE Software, 2005. – Vol. 22. – No. 3. – P. 36–42.
2. **Fecarotta J.** MyBoeingFleet and Agile Software Development // Proceedings of the Agile 2008 Conference. – Toronto, Canada, 2008. – P. 135–139.
3. **Scotland K., Boutin A.** Integrating Scrum with the Process Framework at Yahoo! Europe // Proceedings of the Agile 2008 Conference. – Toronto, Canada, 2008. – P. 191–195.
4. **Laanti M., Salo O., Abrahamsson P.** Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation // Information and Software Technology, 2011. – Vol. 53. – No. 3. – P. 276–290.
5. **Ambler S. W.** Has Agile Peaked? Let's look at the numbers // Dr. Dobb's Journal, 2008.
6. **VersionOne.** State of Agile Survey 2010. Online: http://www.versionone.com/pdf/2010_State_of_Agile_Development_Survey_Results.pdf.
7. **Danubianu M., Socaciu T., Amariei, D.** Distributed Data Mining System for Tourism Industry // Electronics and Electrical Engineering. – Kaunas: Technologija, 2010. – No. 3(99). – P. 31–34.
8. **Schwaber K., Beedle M.** Agile Software Development with Scrum. – Upper Saddle River, USA: Prentice–Hall, 2002. – 158 p.
9. **Schwaber K.** Agile Project Management with Scrum – Redmond, USA: Microsoft Press, 2004. – 163 p.
10. **Dybå T., Dingsøyr T.** Empirical studies of agile software development: A systematic review // Information and Software Technology, 2008. – Vol. 50. – P. 833–859.
11. **Dingsøyr T., Dybå T., Abrahamsson P.** A Preliminary Roadmap for Empirical Research on Agile Software Development // Proceedings of the Agile 2008 Conference. – Toronto, Canada, 2008. – P. 83–94.
12. **Carver J. C., Jaccheri L., Morasca S., Shull F.** A checklist for integrating student empirical studies with research and teaching goals // Empirical Software Engineering, 2010. – Vol. 15. – P. 35–59.
13. **Hilburn T. B., Kornecki A. J.** Graduate Curricula in Software Engineering and Software Assurance: Need and Recommendations // Electronics and Electrical Engineering. – Kaunas: Technologija, 2010. – No. 6(102). – P. 67–70.
14. **Jørgensen M.** A review of studies on expert estimation of software development effort // Journal of Systems and Software, 2004. – Vol. 70. – No.1, 2. – P. 37–60.
15. **Moløkken–Ostvold K., Haugen N. C., Benestad H. C.** Using planning poker for combining expert estimates in software projects // Journal of Systems and Software, 2008. – Vol. 81. – No. 12. – P. 2106–2117.
16. **Mahnic V.** Teaching Scrum through team–project work: students' perceptions and teacher's observations // International Journal of Engineering Education, 2010. – Vol. 26. – No. 1. – P. 96–110.
17. **Moe N. B., Dingsøyr T., Dybå T.** Overcoming Barriers to Self–Management in Software Teams. // IEEE Software, 2009. – Vol. 26. – No. 6. – P. 20–26.
18. **Mahnic V., Drnovscek S.** Introducing agile methods in the development of university information systems // Proceedings of the 12th International Conference of European University Information Systems (EUNIS'2006). – Tartu, Estonia, 2006. – P. 61–68.
19. **Mahnic V., Rozanc I., Pozenel M.** Using E–business technology in a student records information system // Proceedings of the 7th WSEAS International Conference on E–Activities. – Cairo, Egypt, 2008. – P. 80–85.
20. **Grenning J.** Planning poker or how to avoid analysis paralysis while release planning. – 2002. Online: http://renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf.
21. **Cohn M.** User stories applied for agile software development. – Boston, USA: Addison–Wesley, 2004. – 268 p.
22. **Melnik G., Maurer F.** A Cross–Program Investigation of Students' Perceptions of Agile Methods // Proceedings of the 27th International Conference on Software Engineering. – St. Louis, USA, 2005. – P. 481–487.
23. **Runeson P.** Using students as experiment subjects – an analysis on graduate and freshmen student data // Proceedings of the 7th International Conference on Empirical Assessment in Software Engineering. – Keele University, UK, 2003. – P. 95–102.
24. **Mihhailov D., Sudnitson A., Kruus M.** Project–Oriented Approach to Low–Power Topics in Advanced Digital Design Course // Electronics and Electrical Engineering. – Kaunas: Technologija, 2010. – No. 6(102). – P. 151–154.
25. **Valdez M. T., Agreira C. F., Ferreira C. M., Maciel Barbosa F. P.** Teaching, Learning and Exploring the Use of Project–Based Learning // Electronics and Electrical Engineering. – Kaunas: Technologija, 2010. – No. 9(105). – P. 117–120.

Received 2011 02 10

V. Mahnic. A Case Study on Agile Estimating and Planning using Scrum // Electronics and Electrical Engineering. – Kaunas: Technologija, 2011. – No. 5(111). – P. 123–128.

We describe a case study that was conducted at the University of Ljubljana with the aim of studying the behavior of development teams using Scrum for the first time, i.e., a situation typical for software companies trying to introduce Scrum into their development process. 13 student teams were required to develop an almost real project strictly using Scrum. The data on project management activities were collected in order to measure the amount of work completed, compliance with the release and iteration plans, and ability of effort estimation, thus contributing to evidence-based assessment of the typical Scrum processes. It was found that the initial plans and effort estimates were over-optimistic, but the abilities of estimating and planning improved from Sprint to Sprint. Most teams were able to define almost accurate Sprint plans after three Sprints. In the third Sprint the velocity stabilized and the actual achievement almost completely matched the plan. Bibl. 25, tabl. 4 (in English; abstracts in English and Lithuanian).

V. Mahnic. „Agile" metodų taikymas projektų valdymui įvertinti ir planuoti // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2011. – Nr. 5(111). – P. 123–128.

Aprašoma „Scrum" projektų valdymo sistema. Ši sistema naudojama Liublianos universitete programinės įrangos kūrimo procesui organizuoti. Beveik realiems projektams įgyvendinti buvo sudaryta trylika studentų komandų. Bandant nustatyti atitiktį „Scrum" procesams organizuoti buvo renkami įvairūs duomenys (darbo pabaigimo lygis, atitikimas planams ir kt.). Nustatyta, kad, bandant ketvirtą kartą, pagal „Scrum" metodologiją procesą galima organizuoti be klaidų. Bibl. 25, lent. 4 (anglų kalba; santraukos anglų ir lietuvių k.).