ISSN:2184-044X ISBN:978-989-54312-6-7 © 2019
DOI: 10.36315/2019v2end079

A COLLABORATIVE LEARNING PLATFORM TO ASSESS THE USE OF AGILE METHODOLOGIES IN ENGINEERING STUDIES

Francy Rodríguez[1], Diego Vigueras[2], Maria Cerrato Lara[3], Víctor Rampérez[1], Javier Soriano[1], & Guillermo Vigueras[1]
[1]Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid (Spain)
[2]Departamento de Psicología y Sociología, Universidad de Zaragoza (Spain)
[3]Facultat d'Educació, Universitat Internacional de Catalunya (Spain)

Abstract

The success of using agile methodologies for collaborative work in industry, has led to adopt these methodologies for teaching Software Engineering. The curricula has evolved in recent years in order to introduce the use of agile development, so that the students practice their use and train the required skills for project-based collaborative work. Agile methodologies are characterized by being iterative and incremental, with short cycles, constant deliveries and a high level of interaction among team members. These characteristics constitute a challenge for educators and students since, in short periods of time, it is necessary to evaluate and provide feedback to individual and group work, regarding aspects like methodology usage, tools management, and collaboration within the team. For that reason, a Computer Supported Collaborative Learning (CSCL) environment has been developed to assist academics in evaluating and providing feedback to students. The CSCL environment is based on the collaborative platform GitLab, which has been adapted to implement concepts associated to SCRUM, an agile methodology widely adopted. Additionally, the use of GitLab allows to automatically collect information regarding individual and team work of students. Using GitLab data collected, a Learning Analytics platform has been developed in order to analyse group and individual work during the execution of student projects using SCRUM. The objective is to determine if SCRUM helps students to elaborate better software, by evaluating methodology adoption and quality of the resulting software. A prototype of the platform was developed and used in a Software Engineering undergrad course at a Spanish University, in which 79 students divided into groups of 3-4 people, developed two independent projects. Preliminary results show that the proposed CSCL environment helps in providing insight for evaluating and giving feedback to students. Additionally, the data collected by the CSCL environment showed a good correlation of SCRUM adoption by students and quality of resulting software.

Keywords: *Collaborative learning, Learning analytics, Undergraduates, SCRUM methodology, Gitlab, software*

1. Introduction

Due to the increase in the use of agile methodologies in the software development industry (Kropp & Meier, 2013), agile development has been introduced as part of the Software Engineering (SE) courses with the aim of making students knowledgeable of the methods and tools most used in companies, and develop skills for collaborative work (Anslow & Maurer, 2015). Agile methodologies are characterized by an iterative and incremental development strategy, with emphasis on collaboration and direct communication. The development is made in short iterations and, at the end of each one, a product that complies with a set of predefined requirements must be delivered. During each iteration, the development team must make decisions, and adapt to unforeseen events. It is necessary to evaluate both students and work teams and provide feedback promptly so that they can apply corrections and improvements between the iterations (Bai et al., 2018).

On the other hand, there is no consensus on which metrics are the most appropriate to evaluate the performance of groups and students, in terms of collaboration and communication (Alperowitz, Dzvonyar, & Bruegge, 2016). In this work, we propose to create a Computer Supported Collaborative Learning (CSCL) environment endowed with a Learning Analytics platform to evaluate the collaborative work of the students instead of just evaluating the final product. The projects are developed following the agile

methodology SCRUM, widely adopted in industry nowadays. Additionally, students used GitLab, an online tool for collaborative software development, which in turn is used for automatically register the development activity.

2. Background

In the field of education in SE, there are different studies dealing with the teaching of agile development through projects. Some studies focus on how to define projects, and strategies for working in class (Anslow & Maurer, 2015), other authors provide recommendations and lessons learned related to the practical teaching of these methodologies (Mahnic, 2012), including suggestions for tools (Scharf & Koch, 2013). In other cases, the studies analyze the gaps in the industry in terms of skills related to agile methodologies and suggest how to develop these skills in SE courses (Kropp & Meier, 2013). There are also proposals to modify curricula to adapt to learning objectives related to this type of methodologies (Kropp, Meier, & Biddle, 2016).

On the other hand, there are studies in which neural networks and data mining have been used to, based on the information generated during the SCRUM process, to understand past events, and predict success factors or future risks in the use of the agile methodology as well as its final result (Helwerda, Niessink, & Verbeek, 2017). In a similar way, this work automatically collects the information generated during the SCRUM process, but data is mined to evaluate the students in terms of methodology adoption, instead of using such information with predictive purposes.

3. Objective

Create a Computer Supported Collaborative Learning (CSCL) environment in the hand-on of the subject Middleware of a degree in Computer Engineering in a Spanish University. Additionally, a Learning Analysis Platform is implemented and used to automatically extract information that allows evaluating how students apply the SCRUM Agile methodology, the performance of individual and team work, as well as their evolution through development iterations. The evaluation of methodology adoption is compared with the quality of the software developed by students, in order to study possible correlations between both factors: SCRUM adoption and quality of software.

4. Method

During the practices of the Middleware course, two projects were implemented using SCRUM. In each project, a different software product was developed. Groups of 3-4 people were formed, in order to assign the different SCRUM roles. The Project Owner (PO) role, focused on the business, is in charge of transmitting the vision of the project to the team, formalizing and prioritizing the requirements. The SCRUM Master (SM) role, whose objectives are: leading the team, ensuring that rules and process are met, managing risks, and working with the PO to maximize the quality of the product. Finally, the Developer role is purely in charge of development tasks with the necessary technical knowledge to generate the product.

In SCRUM, a *sprint* is the temporal unit to represent a development cycle or iteration. In our case, each *sprint* has a duration of one week. On the other hand, each software requirement consists of a non-formal description, called *story*. Before each *sprint*, the PO presents the prioritized *stories*, then the team collectively decides how many and which ones they can commit to. Based on this, the team performs the task planning required to complete the *stories*, and meets regularly in order to synchronize, check the development process as well as analyze possible problems. Due to the initial time frame of the course, the number of *sprints* for each project is established to 5 for the first one, and to 4 for the second one.

Evaluation of students is performed assigning the following weights to four different aspects: methodology (40%), functionality (50%), code quality (5%) and usability (5%). Additionally, four categories are defined for the evaluation of the methodology: suitable use of roles, feedback, reliability of planning, and development activities. In turn, each category is evaluated based on metrics, which values are automatically obtained through the Learning Analysis Platform by querying the GitLab framework.

4.1. Participants

In this pilot test participated 79 students of a Software Engineering course at a Spanish university. Students were arranged in groups of 3-4 people, resulting 20 groups. A mixed strategy, between random assignment and affinity choice, were used to define the development groups. First, groups were created randomly, and then, a maximum of one student per group was allowed to change groups. Subsequently, students are asked to distribute the SCRUM roles within the team so there will be a PO, a SM and, one or two Developers.

4.2. Instrument and data collection

Students are asked to record all their activities during project development using the GitLab framework. This framework provides storage for projects, and allows teams to work concurrently on the same project while tracking changes that each team member makes on the software. SCRUM elements such as *roles*, *sprints*, and *stories* are mapped with GitLab elements.

To obtain information from GitLab, we develop a tool that automatically extracts information from GitLab; the extraction of data can be scheduled to run periodically. Due to the large amount of data obtained from GitLab, we selected the set of data of interest for this study. Some of the data comprises: the role that creates/assigns a given *story*, number and frequency of meetings, planning fulfillment, and number and frequency of software updates per team member.

In addition to information extraction, the developed tool allows the generation of different graphs to assist in the analysis of SCRUM usage and team performance, contrasting information such as: *stories* assigned/completed within the same *sprint*, or the number of product updates within a *sprint*. Such graphs are obtained by performing different data queries, where the scope can be customized: by project, by *sprint*, or by date. The data extraction and information visualization capabilities are the basis of the Learning Analysis Platform for providing a useful tool to give feedback to the teams.

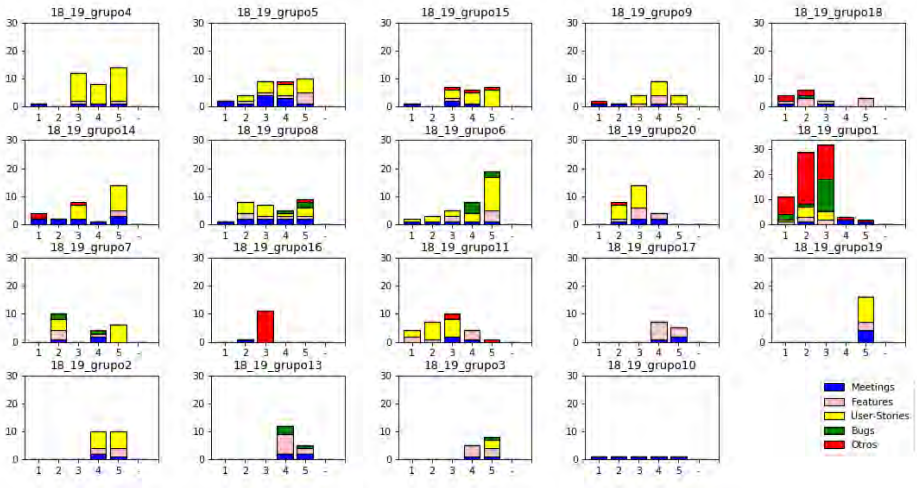
4.3. Statistical analysis

Using the data collected through the Learning Analysis Platform, the statistical analysis has been based on a correlation study by comparing the level of SCRUM adoption by students with the quality of the resulting software developed by students.

On one hand, the quality of the resulting software has been evaluated by one of the professors of the course. Thus, as mentioned above, the quality is obtained from the grade calculated as a weighted sum of three aspects of the students' projects: functionality (50%), code quality (5%) and usability (5%). In this way, using a homogeneous criterion, which in turn is applied by a single evaluator, enforces a fair evaluation process.

On the other hand, the level of adoption of SCRUM is obtained through the Learning Analysis Platform. The information automatically collected and the plots generated by the platform, assist in the task of assessing the adoption of the development methodology. Concretely, the development activity collected by the platform is combined for generating three types of graphs: the individual activity of each student within his/her group, level of adoption of SCRUM roles by students and the temporal distribution and qualitative/quantitative measurement of each group activity. Due to space limitations only an example of the latter type of graph is shown in Figure 1. Thus, Figure 1 shows temporal distribution of the development activity for each group by plotting, on a per-sprint basis, the development items and SCRUM activity (i.e. *meetings*, *user-stories*, *features*, *bugs and others*).

Figure 1. Graph showing the time distribution and qualitative/quantitative measurement of each group activity.



Using the data resulting from the evaluation of software quality and the level of adoption of SCRUM by each group, a correlation analysis is performed in order to assess if the use of SCRUM helps students during the development process.

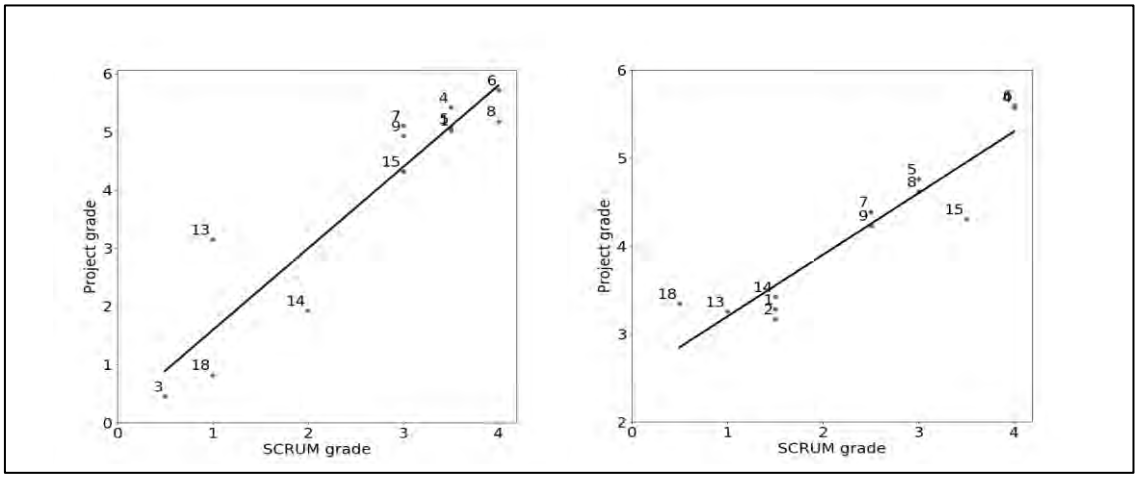
5. Results and discussion

According to the statistical analysis, described above, results have been obtained showing the relation between level of adoption of SCRUM and the quality of the software developed by students. This relation has been obtained for the two projects carried out by students within the context of the course.

5.1. Results

Results are shown in the form of correlation plot. Fig. 2 shows the correlation obtained for the two projects developed during the course. The plot on the left corresponds to results for the first project and plot on the right, corresponds to results for the second project. The different points plotted in Figure 2 represent a single group. The label associated to each point is the group number, displayed for tracking purposes. Additionally, the plots in Figure 2 are complemented with a fitting line, to show the correlation trend.

Figure 2. Scatter plots showing correlation among SCRUM and projects grades. Left: 1st project. Right: 2nd project.



5.2. Discussion

Different conclusions can be extracted from Figure 2. On one hand, it can be observed the level of correlation between the use of SCRUM, measured as SCRUM grade, and the quality of the developed software, measured as Project grade. Looking at both plots, there is almost perfect correlation, since teams with a low level of adoption of SCRUM have resulted in software with bad quality, in terms of functionality, code quality and usability.

On the other hand, Figure 2 also shows more relations between groups with a bad and good use of SCRUM. Student groups can be classified in two sets: one with a SCRUM grade lower or equal to 2 (considered as bad) and another one with SCRUM grade higher than 2 (considered as good). Thus, the set of groups with bad SCRUM grade for the first project would be: 3, 13, 14 and 18 (see left plot in Figure 2). The same set for the second project would be composed of groups: 1, 2, 13, 14 and 18. On the other hand, the set of groups with good SCRUM grade for the first project would be: 1, 4, 5, 6, 7, 8, 9 and 15. The same set for the second project would be composed of groups: 4, 5, 6, 7, 8, 9 and 15.

From a quantitative perspective and according to the sets described above, the ratio of groups with bad:good SCRUM grade, would be 4:8 for the first project and 5:7 for the second project. The fact that these ratios are similar across projects, shows that the number of groups with good use of SCRUM in the first project were not discouraged from continue using the methodology in the second project while still obtaining a good project grade.

From a qualitative perspective, the sets of groups mentioned above show that groups with good use of SCRUM and obtaining a good project grade remained mainly the same (i.e. 4, 5, 6, 7, 8, 9 and 15) with the exception of group 1 which made a good use of SCRUM in the first project but refrain from doing so in the second project. Nevertheless, the fact that group 1 obtained a lower project grade in the second project supports the initial hypothesis about the link between the use of SCRUM in the course and the quality of the software developed by students.

The qualitative and quantitative analysis of Fig. 2 also shows that groups using SCRUM did not increase from the first project to the second project. Additionally, from the twenty groups formed to take part in the course, only twelve groups actually participated in both the first and second project. We leave as a future work the analysis of the possible reasons behind this behavior. Nevertheless, as informally stated

by several students, the calendar and academic duties, carried out simultaneously with this course, was an important limitation for using SCRUM properly.

Finally, this study was possible thanks to the implementation and use of the Learning Analytics platform, since it helped in analyzing and understanding the amount of data generated by students groups while developing the assigned projects using the GitLab platform.

Acknowledgements

This work has been partially funded by Universidad Politécnica de Madrid through Education Innovation Project with reference IE1819.1003 and by Universitat Internacional de Catalunya (Spain).

References

Alperowitz, L., Dzvonyar, D., & Bruegge, B. (2016). Metrics in Agile project courses, 323–326. https://doi.org/10.1145/2889160.2889183

Anslow, C., & Maurer, F. (2015). An Experience Report at Teaching a Group Based Agile Software Development Project Course, (Cmmi), 500–505. https://doi.org/10.1145/2676723.2677284

Bai, X., Pei, D., Li, M., Li, S., & Ye, D. (2018). Continuous delivery of personalized assessment and feedback in agile software engineering projects. *2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 58–67. https://doi.org/10.1145/3183377.3183387

Helwerda, L., Niessink, F., & Verbeek, F. J. (2017). Conceptual Process Models and Quantitative Analysis of Classification Problems in Scrum Software Development Practices, 357–366. https://doi.org/10.5220/0006602803570366

Kropp, M., & Meier, A. (2013). Teaching agile software development at university level: Values, management, and craftsmanship. *Software Engineering Education Conference, Proceedings*, (November 2014), 179–188. https://doi.org/10.1109/CSEET.2013.6595249

Kropp, M., Meier, A., & Biddle, R. (2016). Teaching agile collaboration skills in the classroom. *Proceedings - 2016 IEEE 29th Conference on Software Engineering Education and Training, CSEEandT 2016*, 118–127. https://doi.org/10.1109/CSEET.2016.27

Mahnic, V. (2012). A capstone course on agile software development using scrum. *IEEE Transactions on Education*, *55*(1), 99–106. https://doi.org/10.1109/TE.2011.2142311

Scharf, A., & Koch, A. (2013). Scrum in a software engineering course: An in-depth praxis report. *Software Engineering Education Conference, Proceedings*, 159–168. https://doi.org/10.1109/CSEET.2013.6595247