Academia Journal of Scientific Research 6(12): 465-469, December, 2018
DOI: 10.15413/ajsr.2018.0185
ISSN 2315-7712
©2018 Academia Publishing

Research Paper

Case study of the Scrum Agile Approach in a Computer Science undergraduate subject

Accepted 15th November, 2018

Rodrigo Alves Costa

Universidade Estadual da Paraíba – Brazil.

E-mail: rodrigo@ccea.uepb.edu.br. Tel: +55 83 988865112.

ABSTRACT

Software development agile approaches are mainly characterized for prioritizing the development of functionalities through executable code rather than the production of extensive written documentation and presenting quick responses to changes. In this context, this work aims to report the application of Scrum in the undergraduate classroom environment with the subject of Software Engineering, presenting the entire project execution process. In the end, an experience that simulated a real situation was achieved, common in software development companies, which facilitated the learning of the content of the subject.

Key words: Education, Computer Science, Agile, Scrum.

INTRODUCTION

Agile software development approaches diverse from traditional development processes is mainly due to the fact that they prioritize the development of functionalities through executable code rather than the production of extensive written documentation, as well as, to provide quick answers to changes and collaboration with the client instead of following rigid plans and contractual negotiations (Leal, 2009). According to a study carried out by Ambler (2006), after collecting information from a large number of professionals using different methods, both agile and traditional, it was noticed that agile approaches actually improve project results in terms of quality, customer satisfaction and productivity, without significantly increasing its cost.

One of the most acknowledgeable approaches to agile management of software projects is Scrum (Schwaber, 2004). According to Sousa et al. (2015), management with Scrum involves the use of a framework of procedures and artifacts which lead to the objectives of self-organization of development teams. Such a framework can be used in the classroom environment in the teaching and learning processes, since Prikladnicki and Audy (2008) suggest that alternative approaches can help students to learn more effectively by applying student oriented exercises, like the substitution of regular lectures with the discussion of practical cases, the application of group dynamics, or the execution of a project from the beginning to its end by student workgroups.

In this scenario, this paper reports the experience of using the Agile Scrum approach in the classroom environment in a software development project in the Software Engineering subject in the undergraduate course of the State University of Paraiba.

RELATED WORK

Meireles and Bonifácio (2015) reported on the practical experience of undergraduate students in learning the subject of Software Engineering using the combination of problem-based teaching methodology (PBL) and the use of Scrum as a methodological tool in order to construct real projects applied to mobile devices, with the aim of contributing to the creation of a motivating environment, making the learning process more dynamic, collaborative and enjoyable. The work developed by such authors presents a strong relationship of similarity with the experience report presented in this paper, since it focuses on the teaching of Software Engineering linked to the development of products, thus, seeking to provide students with real life experiences. In contrast, the work differs since Meireles and Bonifácio (2015) reported that students were divided into four (4) teams, where each one had the freedom to choose the theme of the mobile application. In

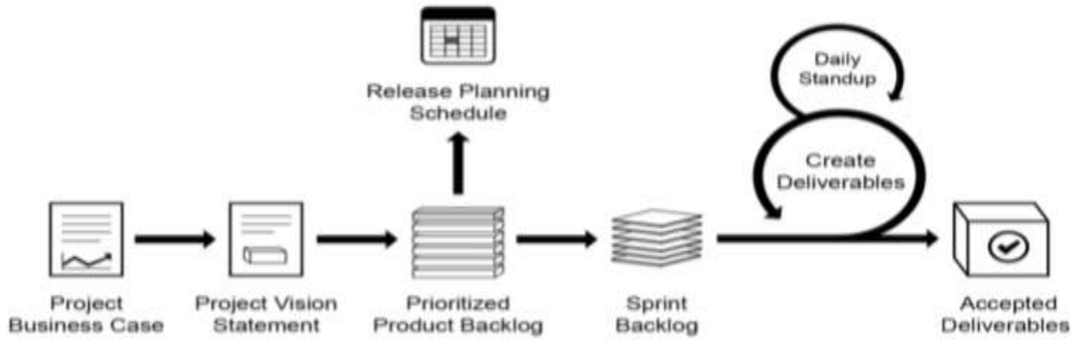

Figure 1: Scrum development cycle for a sprint. Source: (SBOK 2017).

the work presented here, although separated into similar teams, the students joined efforts to develop a larger tool, performing a collaborative work and, in addition, a series of software tests were conducted in order to evaluate the quality of the project developed.

Carvalho and Mello (2012) presented the results of a research carried out in a small technology-based company, in which the objective of the study was to analyze the implementation of Scrum in software product development projects, as well as, to understand and measure the impact of such implementation company-wise. The authors realized that with the adoption of the approach, communication was improved, as well as, team motivation, project costs, time and risk were optimized, and team productivity was also increased. Branco and Pires (2008) present the experience in the use of Scrum in offshore projects, highlighting the perceived advantages.

Siller and Braga (2013), within the theme of Software Engineering, presented a software, more precisely an educational game for the teaching and practice of Scrum, showing the student how the process works in practice. The development process was guided by the INTERA methodology which, according to the authors, is specific for the development of learning objects.

Finally, it is possible to notice that there are several works that seek to promote in students the experience of a development process, but few specifically address the activities carried out in the academic environment as a means of learning, while the stages present in a production environment are closely followed and related to the theoretical content of the classroom. Thus, in this paper we will report the experience in using the Scrum approach as an aid in the teaching and learning process in the classroom environment.

USAGE OF THE SCRUM

Context

Scrum is an agile approach for the development of products and services (Rubin, 2012). According to Fadel and Silveira (2010), this approach does not define a specific technique for software development during the implementation stage. It focuses on how team members should work to produce a flexible system in a changing environment. In it there are three (3) important and well defined roles namely: (i) the Scrum Master is the facilitator and conflict solver; (ii) the Product Owner is the person responsible for the project itself; (iii) the Scrum team is the development team, in which everyone interacts to develop the product together (Wazlawick, 2013).

The Scrum process, seen in Figure 1, starts with the Product Backlog, which contains all the functionality defined with the client / user, improvements and defect corrections to be developed in the project (Oliveira and Lima, 2011). Based on this list, the Sprint is planned to run. Sprints are iterations during which product sets of Product Backlog (list of desired features) are implemented, and can vary their duration from 02 (two) to 04 (four) weeks. At the start of Sprint, the Product Owner, the Scrum Master, and the team participate in the two-part planning meeting. In the first part the Product Owner, the Scrum Master and the team participate select the tasks (Product Product Backlog) that will be part of the Sprint Backlog as prioritized by the Product Owner. In the second part, the Scrum Master and the team participate to subdivide the tasks in order to better understand them to facilitate the implementation of each of them (Oliveira and Lima, 2011).

During the Sprint run, the Scrum Master and the team conduct daily meetings (which are named Daily Scrum Meeting) to keep track of the progress of the project. At the end of Sprint, the team performs the presentation of the product increment developed for the Product Owner, which in turn validates if what was requested corresponds to what was developed. Thus, Sprint is finalized starting a new cycle until the final product is completed (Oliveira and Lima, 2011).

PROJECT EXECUTION

For the development of the main project, the group of 20 (twenty) people was divided into 03 (three) teams. The division of the teams happened through the affinity between the students verified through a SWOT analysis. A questionnaire with questions related to strengths and weaknesses, opportunities and threats was filled by each student.

Each team worked with different modules of the main software. In order to allow parallel development amongst the teams, development environment scenarios were created for each team using the GitHub tool (2016), and these modules were integrated, synchronizing these environments in a single project. One member of each team was responsible for this task. There was a pre-set order for such synchronization to occur, and at the end of the process the main environment was all changed.

The initial phase of the project was carried through research and studies sourced out from documents, book chapters and master's thesis related to each of the individual projects. Following the project comprehension phase, a meeting was held with the lecturer, who during the project assumed the roles of Product Owner and project customer. In this meeting, an interview was conducted to help clarify the objectives and requirements desired in the tool, as well as, concerns or questions and any other doubts from the development teams were solved.

Based on the requirements that were identified during the initial interview, each team was responsible for creating their Product Backlog according to their research using a Google Docs (2016) spreadsheets for making it accessible for everyone at the same time, allowing access from any place independent of the access device.

The fields composing the Product Backlog and that also subsequently make up the worksheet are: "desired" - requirement requested by the client; "In the form of ..." - column responsible for storing a justification or explanation of the requirement; "Type" - classification of the type of the requirement, that is, a characteristic of the system or a presentation to be performed; "Priority" - receives a score according to the degree of urgency or importance of a request, allowing to identify an order of implementation; "Status" - responsible for storing the status of an implementation, being able to change between: TO DO (to be done), WIP (in progress) and DONE (completed); "Sprint Estimated" and "Sprint Done" - the first one stores the Sprint number on which the team expects to meet the requirement, while the second one stores the actual Sprint number on which the request was actually served; "Responsible" - informs the person responsible for completing an item, finally; "Notes" - stores possible suggestions and additional information.

After analyzing and documenting the requirements, Sprint cycles started. To allow for this, before starting a Sprint, the teams selected the items that they understood should be implemented using the Sprint Backlog. In addition, for each new feature selected to be implemented, the progress status of the listed requirement was updated accordingly. In case such feature was not developed for any reasons by the end of the Sprint it returned to the Product Backlog to be allocated to the next or to a future Sprint Backlog.

The features selected for each Sprint Backlog were associated with a deadline for development and delivery to the client, and such deadline was defined at the beginning of the subject. The classroom agreed that each Sprint would last not more than 15 (fifteen) days. This duration was chosen due to the time available for a semester, meanwhile facilitating the follow-up and tracking of the course. There was only one moment when the stipulated deadline was not met in the second Sprint. SBOK (2017) accepts that during the first interactions, due to the need of calibration, some Sprints allow for flexibility for adjustment and calibration – the class as a whole agreed to take the measure to extend the length of this Sprint length to three (3) weeks.

For each team, a Scrum Master was elected, however, this position was rotating - that is, in each Sprint the responsibility was shared, and the strategy to elect a new member to take the position gave the students the opportunity to participate in this experience performing the leadership role.

This tactic of rotation was particularly interesting and had a positive impact during the development of the project. Initially, some members of the team did not have as much commitment as others. However, being a Scrum Master increased the level of commitment, as it was each member's responsibility to be the team's interface with the Product Owner, and all the criticisms, orientations and questions were directed to the Scrum Master who took the responsibility to pass along such information to the rest of the team. This student was also responsible to eliminate impediments following the activities of the group. However, it is worth mentioning that some members had difficulty taking the role of Scrum Master most of the time because they do not have a fully communication profile yet. Often times, difficulties like these were overcome with the help of the rest of the team members, which enhances the characteristic of a Scrum team as committed and self-organized.

During the progress of the Sprints, each team was responsible to conduct daily meetings (Daily Scrum Meeting) to disseminate knowledge and update on the progress of the project. These meetings took place in two forms, face-to-face and virtual. In the face-to-face meetings, the team took advantage of the classes bi-weekly appointments to establish the cadence of communications. In addition, it was necessary to resort to the intervals between the classes of the discipline and the space reserved by the lecturer for discussion and updating of the progress. In the virtual form, videoconference and chat resources were used, using tools such as Skype (2016), Hangouts (2016) and Facebook (2016) available in social networks.

In addition to the meetings needed to keep the team aligned and on track, the Trello (2016) tool was used. It is

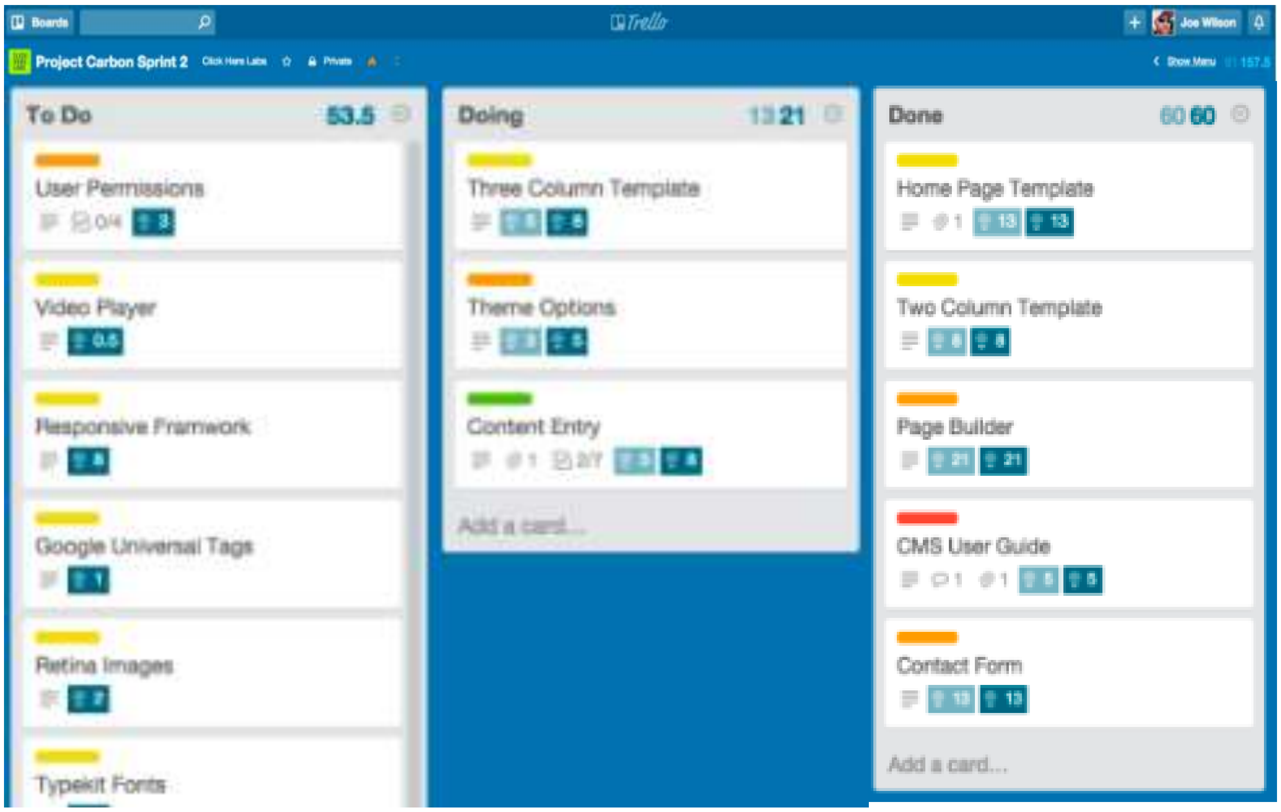

Figure 2: Trello 2018 extract from one of the Projects. Source: Author.

an activity manager than can be used to track progress of tasks and it was used to replace the Scrum dashboard, which allowed each team to effectively control the tasks, establishing an organization, and at the same time it was possible to classify the activities to be performed in "Uninitialized tasks", " In Progress "and" Completed ", as can be seen in Figure 2.

After the completion of each Sprint, the finished activities were reorganized and grouped according to Sprint numbering, thus, maintaining a history. At the end of each Sprint, the teams performed validation activities, such as performing unit testing, integration, performance and system tests and compiling obtained results that were recorded through spreadsheets. At the end of a Sprint, it was necessary to perform Sprint Reviews to the Product Owner, in order to demonstrate the functionalities implemented and, consequently, defining the next activities.

PROJECT COMPLETION

When the scope of the development of all the three teams was complete and all the requirements were validated as finalized, the three modules were merged into a single system, unifying and creating a stable version of the developed tool, which was presented as the final work of the subject. Thereafter, there was a rearrangement among the teams, through which the students started to work profile-oriented.

Therefore, the 20 (twenty) students were split into 03 (three) new teams, in which each member chose the area of their greatest interest. This was done as a means to renew commitment by the students and thus allow for the activities to be completed more quickly. However, it was difficult to define the time required for the development of each task by each team member. The reason for that was because a high degree of adaptation was required by each student as they needed to adapt to the new way of working required in the new structure, especially due to the fact that the members in newly formed teams had never worked together as a Scrum team. In addition, each member had a schedule of availability and particular working pace, which also required some time for adaptation. The assembled teams were basically organized as follows: development, testing and documentation.

Throughout the project, the deliverables were managed using GitHub, which was chosen for being an open source tool for continuous integration and for including features that allow for documentation, code management, open source repository and enable the creation of groups of collaborators. Thanks to GitHub, source code, test results and all tool documentation are available in the repository that can be accessed publicly at a public address available on the Web.

CONCLUSIONS AND RECOMMENDATIONS

The classroom environment needs new approaches in order to allow for the modernization of learning and to ensure that the teacher and students develop new experiences. For a real-world experience, based on real methods applied in the labor market to be generated and simulated in the teaching environment, usual practices in the academic environment have to be adapted to say the minimum.

The practical application of the Scrum method along with the development process presented positive and negative results. The benefits resulted in characteristics which are typical on software development corporations, such as obtaining software implemented on an increasingly fast pace and with satisfactory quality, promoting moments of learning during the project execution and, simulating, other real industry situations. The points to improve are not directly related to the methodology, but rather a consequence related to the lack of practice and experience of the key team players engaged in the approach who are still undergraduate students. Among the points of criticism raised were the short deadlines for deliverables, since students reported being accustomed to longer periods and slower rhythms to deliver work; in addition, there were problems related to the handling of the Github tool, since the teams did not present previous knowledge of it, and it was necessary to study the tool and then perform among the teams to share obtained expertise.

Finally, it is very clear that the practical application of the agile Scrum methodology in a classroom environment was successful, presenting a functional system, while at the same time emphasizing in the students involved a sense of commitment throughout the process.

As future work, we intend to apply Scrum in a new subject with the same group of students, allowing for a comparison between the results obtained in the subject of Software, which will probably allow a more in-depth analysis. In parallel, the use of Scrum during the Software Engineering subject will go on, also allowing for the learning of new concepts related to Software Engineering.

ACKNOWLEDGMENTS

The author would like to thank UEPB Computer Science Course Coordination and the Centro de Ciências Exatas e da Natureza (CCEA) Direction for the support of the development of this research. Our thanks reach out to every student who participated in the entire development process using Scrum.

REFERENCES

Ambler, Scott W (2006). Agile Adoption Rate Survey Results: March 2006. Available in: <http://www.ambysoft.com/surveys/agileMarch2006.html

Branco, Jônathas Diógenes CC, Ciro CP, Carlo GS (2008). Utilizando Scrum em projetos off-shore. In: I Congresso Tecnológico Infobrasil, Fortaleza.

BRQ (2016). Metodologias Ágeis de desenvolvimento de software, http://www.brq.com/metodologias-ageis/.

Carvalho, Bernardo V, de; M, Carlos HP (2012). Aplicação do método ágil scrum no desenvolvimento de produtos de software em uma pequena empresa de base tecnológica. Gestão & Produção: Scielo, São Carlos. 19(3): 557-573.

Facebook (2016). Available: https://www.facebook.com/. Access: 16 sep. 2018.

Fadel AC, Silveira H, da M (2010). Metodologias ágeis no contexto de desenvolvimento de software: XP, Scrum e Lean. 26 f. Monografia (Especialização) - Curso de Faculdade de Tecnologia, Unicamp – Universidade Estadual de Campinas, Limeira.

Github (2016). Available in: https://github.com/. Accessed: 16 sep. 2018.

Google Docs (2016). Available in: http://docs.google.com/. Accessed: 16 sep. 2018.

Hangouts (2016). Available in: https://hangouts.google.com/. Acessado em: 16 sep. 2018.

Leal I (2009). Requisitos de Metodologias de Teste de Software para Processos Ágeis". October.

Meireles MAC, e Bonifacio BA (2015). Uso de Métodos Ágeis e Aprendizagem Baseada em Problema no Ensino de Engenharia de Software: Um Relato de Experiência. In: Anais do XXVI Simpósio Brasileiro de Informática na Educação (XXVI SBIE). pp.180-189.

Oliveira E, e Lima R (2011). Estado da Arte Sobre o Uso do Scrum em Ambientes de Desenvolvimento Distribuído de Software. Revista de Sistemas e Computação, Salvador. 1(2) :106-119.

Prikladnicki R, Audy J (2008). Desenvolvimento distribuído de software. Rio de Janeiro: Elsevier.

Rubin KS (2012). Essential Scrum: A Practical Guide to the Most Popular Agile Process. AddisonWesley Professional. Pp.504.

SBOK (2017). *A Guide to the Scrum Body of Knowledge (SBOKTM Guide)* – Third edition. SCRUMStudy, VMEdu, Inc.

Schwaber K (2004). Agile Project management with Scrum. Microsoft Press.

Sille F, e Braga JA (2013). Software Educacional para Prática do Scrum. In II CBIE – Workshops (WCBIE). pp. 152-161.

Skype (2016). Disponível em: http://www.skype.com. Accessed: 16 sep. 2018.

Sousa HT et al (2015). Apoio Automatizado ao Planejamento de Sprints em Projetos Scrum. In Anais XI Simpósio Brasileiro de Sistemas de Informação. SBC.

Trello (2016). Disponível em: https://trello.com/. Accessed: 16 sep. 2018.

Wazlawick RS (2013) Engenharia de Software: Conceitos e Práticas. Elsevier, Rio de Janeiro.

Cite this article as:

Costa RA (2018). Case study of the Scrum Agile Approach in a Computer Science undergraduate subject. Acad. J. Sci. Res. 6(12): 465-469.

Submit your manuscript at
http://www.academiapublishing.org/journals/ajsr