MOVING FROM WATERFALL TO AGILE PROCESS IN SOFTWARE ENGINEERING CAPSTONE PROJECTS

Mohammad Alshayeb[1], Sajjad Mahmood and Khalid Aljasser

King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia.

ABSTRACT

Universities offer software engineering capstone course to simulate a real world-working environment in which students can work in a team for a fixed period to deliver a quality product. The objective of the paper is to report on our experience in moving from Waterfall process to Agile process in conducting the software engineering capstone project. We present the capstone course designs for both Waterfall driven and Agile driven methodologies that highlight the structure, deliverables and assessment plans.To evaluate the improvement, we conducted a survey for two different sections taught by two different instructors to evaluate students' experience in moving from traditional Waterfall model to Agile like process. Twenty-eight students filled the survey. The survey consisted of eight multiple-choice questions and an open-ended question to collect feedback from students. The survey results show that students were able to attain hands one experience, which simulate a real world-working environment. The results also show that the Agile approach helped students to have overall better design and avoid mistakes they have made in the initial design completed in of the first phase of the capstone project. In addition, they were able to decide on their team capabilities, training needs and thus learn the required technologies earlier which is reflected on the final product quality.

KEYWORDS

Agile Process, Waterfall Process, Capstone Project, Software Engineering

1. INTRODUCTION

IEEE computer society and the Association for Computing Machinery (ACM) recommend that software engineering students undertake a capstone project that integrates knowledge of software development life cycle, learned throughout the course of the undergraduate software engineering program, in a realistic simulation of professional experience. The traditional waterfall driven approach advocates that a well-planned process for capstone projects would be efficient approach for inexperienced software engineering students. On the other hand, agile driven approach provides hands-on environment to learn and apply software development life cycle knowledge and skills.

The main criteria for selecting one of these approaches is its effectiveness to help students in delivering a successful product at the end of the capstone project.We have been using Waterfall

[1]Corresponding author

Dhinaharan Nagamalai et al. (Eds) : ACSIT, ICITE, SIPM - 2018
pp. 107–114, 2018. © CS & IT-CSCP 2018 DOI : 10.5121/csit.2018.80808

model for twelve years. Waterfall model is known for its organized set of steps starting for the early stages of project proposal till the last stages of testing and delivery. On a 2-semester software engineering project (15 weeks in each semester), the duration is more than enough to cover all the stages of a software project. However, because of the structure of the course (shown in Section 2) and the required deliverables, students find themselves delayed and less motivated because they do not see their software products until the second half of the second semester. Waterfall model would be suitable if the software idea and its requirements are known upfront.In our capstone project, however, when following the Waterfall approach students face several challenges and difficulties. For example, with student little experience in design, after the implementation they realize that the code is not consistence with the design. Since they have only one cycle, it will be hard to go back and fix the design issues. This is even harder for the course instructor since the number of revisions on the design are different from one project to another.Furthermore, most students use new and emerging technologies which have not been discussed in the course they have already taken, therefore, when the implementation time comes, they have to learn new development frameworks and tools which hinder their speed of implementing the planned features which will delay the overall project. This will lead to little time for testing which can affect the final product quality. Finally, students indicated their dissatisfaction from the process as they only do documentation of the Software Requirement Specification (SRS) and the Software Design Document (SDD) in the first semester which causes lack of motivation towards remaining project tasks. This motivated us to move to use Agile in the implementation of the capstone project especially with the positive and promising results reported in literature for moving from Waterfall to Agile in capstone projects.

Over the last few years, a number of studies reported experience about the use of both traditional waterfall plan driven and agile driven approaches for software engineering capstone course [1-5]. Rover et al. [6] presented a case study of developing software applications in two-semester senior project. They found that the use of Agile was successful and beneficial. The results of the case study show that it leads to high satisfaction and helped in producing high quality software. Kuhl [7] reported his experience in moving from the traditional Waterfall model to Agile process. The approach is meant to replace the sequential, and documentation-intensive, steps of the waterfall model with shorter development cycles by releasing a small set of features in each cycle. The author reported that the average quality of the projects was improved when using Agile relative to using Waterfall method. This was obvious as teams were able to implement more functionality and a better level of testing. Ding et al. [8] presented a case study of using Agile and Scrum methods in capstone projects. He presented recommendations to consider when applying Agile in capstone projects. Coupal and Boechler[9] reported their experience and observation in using Agile process in capstone project. They observe that using Agile supports learning and provides a good learning experience and produces good quality products within an academic environment. Devedzic and Milenkovic[10] discussed how to overcome potential problems in teaching Agile along with some recommendation based on their experience in teaching Agile in different universities and different cultural settings. Mahnic[1] reported his experience in teaching undergraduate capstone course in software engineering using Agile. He provided and empirical evaluation of students' progress in estimation and planning. He also provided recommendations and lessons learned to consider when applying Agile in courses. The survey he conducted shows that the students were satisfied and the course met the expectations. These positive and promising results motivated us to conduct report our own study in our own settings. Rico and Sayani[11] described their use of Agile methods in software engineering capstone course. With little training in agile methods, the three teams were able to complete fully functional e-commerce websites. Knudson and Radermacher[12] provided suggestions from interviews of developers and managers who use Agile process and from student feedback for incorporating agile processes in the capstone course.Souza et al. [13]presented an evaluation of adapting Scrum method to evaluate the capstone project. El-Abbassy et al. [14]presented a framework for adopting and evaluating agile practices in computer science education.

In this paper, we report on our experience and the changes made to move from Waterfall process to Agile process in conducting the software engineering capstone project. The modifications involve the restructuring of the course, the deliverables and the assessment plans. The paper also presents the survey results which indicate the strengths of using Agile process. Finally, discussion and some direction for further improvement in future offering of the course is discussed.

The paper is structured as follows. Section 2 discusses the course description and structure, Section 3 presents the Agile structure. Section 4 details the evaluation of the Agile adoption. Finally, section 5 presents the conclusion and future work.

2. COURSE DESCRIPTION AND STRUCTURE

The two-semester capstone project is designed to follow Waterfall development methodology in a real-world context. Students are required to work in teams of five to six members to develop a realistic project based on user requirements provided by industrial sponsors or domain experts. The course is allocated seven credit hours that are distributed over two semesters (each semester consists of fifteen weeks). Senior students who have completed a set of software engineering and computer science core courses including software requirements engineering, software design and project management register the capstone project. The capstone project has no formal lectures and overall course delivery is structured as a set of weekly deliverables. The project teams meet with the instructor every week to submit their deliverables. The two-semester capstone project has the following objectives:

1. To employ knowledge gained from courses throughout the program such as development of requirements, design, implementation, project management, and quality assurance to develop a software solution to a real-world problem.
2. To apply all appropriate project management techniques.
3. To learn how to work in teams.
4. To enhance communication and writing skills.
5. To instill life-long learning skills.
6. Understand the impact of computing solutions in a global and societal context.

The course delivery schedule and assessments are shown in Table 1.

Table 1: Course Schedule and Assessment

Semester	Deliverables	Assessment
First semester	Project proposal	5%
	Project Plan	10%
	Software requirements specification (SRS)	45%
	Software Design Document (SDD)	30%
	Test cases	5%
	Prototype	5%
Second Semester	SRS and SDD review	10%
	Project Plan	10%
	Implementation	50%
	Testing	10%
	User and Deployment Manuals	10%
	Release - Final delivery of system	10%

In the first semester, student work on the requirements and design of the project, they also develop the test cases. At the end, they develop a prototype. In the second semester, they review their requirement and design document and then they implement the system. After implementation, they test and write documentation. Finally, they release the product.

3. AGILE BASED PROJECT SETTING

The students enrolled in the waterfall-basedproject setting course were facing a number of challenges, as discussed in the introduction. The challenges were associated with student outcomes associated with design, implementation and selection of appropriate technology/tools for the projects. The main reason for these challenges were lack of experience with latest frameworks, tools and associated technologies. As a result, design specifications completed in the first semester of the project often required significant refactoring. Furthermore, teams need time to self-learn new technologies and platforms. Hence, teams were left with little time to work on the implementation and produce a system which satisfied the stakeholder requirements.

Table 2 shows the new agile based project setting. In the first part, students develop project plan, software requirements specification; software design document for 30% of features; implementation; testing and first release of the product. In the second part, students complete the design and implementation of the product, alpha and beta testing of their code and evaluate the final product.

Table 2: Agile based project schedule

<table>
<tr><th colspan="15">First Semester</th></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
<tr><td colspan="2">Project Plan</td><td colspan="5">Requirements Specifications</td><td colspan="3">Design of 30% features</td><td colspan="4">Implementation</td><td>Testing & Release 1</td></tr>
<tr><th colspan="15">Second Semester</th></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
<tr><td>Requirements review</td><td colspan="4">Design Specification</td><td colspan="6">Implementation</td><td>Beta Testing</td><td>User Manual</td><td>Release 2</td></tr>
</table>

After running the capstone project using Agile for two semesters, it was obvious that students have improved their skills in:

1. Self-learning: students were able to self-learn new development languages, tools and technologies early as they have to start implementation in the first semester. This allows students to have better understanding of the system requirements.

2. Better design: Students iteratively design the project and hence they were able to learn from the first cycle and avoid the mistakes in the second cycle.

3. Risk management: have a two released, students reduced the project risk as they have a working system early. As a result, teams have more confidence in their ability to meet stakeholder requirements and go through a through testing process before releasing the final product.

4. Project management: students have real work experience in Agile management and planning. Students were able to assess their capabilities early and hence prepare themselves for any needed training

4. AGILE APPROACH EVALUATION

To evaluate the impact of using Agile process instead of traditional Waterfall model, a survey over two semesters for two different sections was conducted. In the first offering, 16 students participated in the survey while there are 13 students participated in the survey. The two offerings were done by two different instructors; however, they followed the same approach.

The survey consisted of 9 questions, the questions are listed below:

1. The experience in the first semester helped the team in enhancing the project design of the second phase.

2. Having design and implementation in the first semester, forced the team to learn the new technology I need earlier.

3. The experience gained in the first semester helped the team in better understanding the requirements of the project.

4. The experience gained in the first semester helped the team in better understanding the team capabilities.

5. The experience gained in the first semester helped the team in better understanding the needed training for the team.

6. The experience gained in the first semester helped our team in building a realistic schedule for the activities in the second semester.

7. The new setting helped the team in reducing the risk of not completing the project.

8. I would you recommend using the Agile approach in the future offering of the course?

9. Recommendations and Comments.

Figure 1 represents a summary of the student responses, the figure shows the average satisfaction for each survey question. Figure 1clearly shown that all question scored at least 4.4 out of 5.

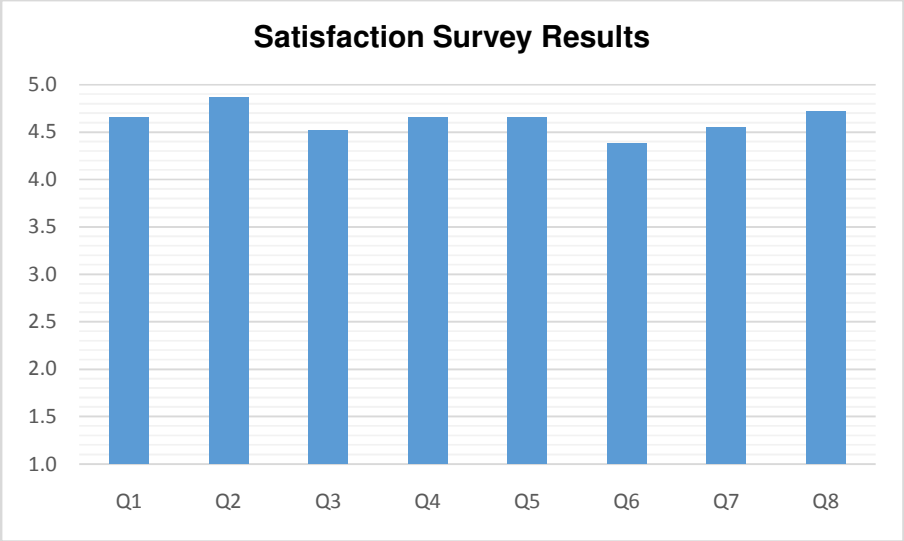

Figure 1. Satisfaction Survey Results

Below, we discuss the reasons for this high satisfaction.

Question 1: This is the first projects in which students perform the complete software life cycle. Students designed a system in their design course without full implementation, hence, they could not tell what works and what does not work in design. In this project, after developing the first phase, students are more confidence of how the design can translate to code and hence the design of the first part helped them in avoiding mistakes they have made in the design of the first part.

Question 2: Most students develop mobile applications for their senior project. As of now, the program does not offer such course. Hence, students have to learn different mobile development technologies in the first semester in order to implement the first cycle. This has a positive impact as they have a better learning curve when they reach the second semester to implement the remaining functionalities.

Question 3: With respect to requirements, students tends to understand the requirements more as they have to select 30% of them to implement, they usually select the most important requirements and thus analyze the requirements and have a better understanding to the who system requirements.

Question 4 & 5: Students form their own teams; therefore, they usually select other students whom they know, yet, students have different capabilities. When students are forced to implement part of the system, they have a better understanding of their capabilities and thus identify the needed training early. Identifying the needed training helped students to plan and get the required training earlier which speeded up their readiness for the next semester activities.

Question 6: Student lack planning experience, thus, their plan tends to optimistic in the first semester especially in the development part. In the second semester, they should have gained some experience to have more realistic schedule.

Question 7: Students have high satisfaction with regard to reducing the project since they have a working version form the first semester. This gives them more confidence and less pressure to complete the second part.

Question 8: High percentage of students recommend keeping using Agile in the future offering of the course. This overall result is obvious from the satisfaction of all previous questions.

Question 9: Recommendations and Comments

Below are some recommendations and comments mentioned by the students:

- "I can't think of any, but this approach helped completing the project smoothly."
- "Including one more agile cycles in 418, will be even better. Like dividing the remaining 70% to 30% - 40%."
- "I liked the way of starting to implement as early as possible."
- "It was great to develop the idea of the project in earlier stages."
- "In my opinion, agile use was a success."
- "I didn't try the old approach, but I imagine that this approach is way better, for all the points mentioned in the questions above. Many thanks to you and for the department's faculty for the continuous improvements you make."
- "I think it would've been better if testing was required during the implementation phase, especially unit testing."

It is obvious that students were satisfied with the Agile approach and speak highly about it.

5. CONCLUSION AND FUTURE WORK

This paper reports our experience in moving from Waterfall process to Agile process in conducting the software engineering capstone project. This move involves restructuring the deliverables and the assessment plans of the two-semester project. To evaluate the impact of the change, a survey was conducted to solicit students' feedback. Twenty-eight students filled the survey. Results show that using Agile helped students in have better design, early training, better risk management and more satisfaction.

In the future offering of the course, we plan to provide more emphases on unit testing and encourage student to use automated unit testing tools such as Junit as currently the focus is more on integration and system testing. Furthermore, we plan to divide the project into three iterations, one in the first semester with 30% of the project features and two in the second semester with 30% and 40% respectively. This decision is based on our observation that more iterations enable the students to have better understanding of the requirements and design and early enhancement of team capabilities by identify training needs. Introducing another iteration will also provide students with opportunity to gain experience in refactoring.

ACKNOWLEDGMENTS

The authors acknowledge the support of King Fahd University of Petroleum and Minerals.

REFERENCES

[1] V. Mahnic, "A Capstone Course on Agile Software Development Using Scrum," IEEE Transactions on Education, vol. 55, no. 1, pp. 99-106, 2012.

[2] M. I. Alfonso and A. Botia, "An Iterative and Agile Process Model for Teaching Software Engineering," in 18th Conference on Software Engineering Education & Training (CSEET'05), 2005, pp. 9-16.

[3] B. Lu and T. DeClue, "Teaching agile methodology in a software engineering capstone course," J. Comput. Sci. Coll., vol. 26, no. 5, pp. 293-299, 2011.

[4] K. Keefe and M. Dick, "Using Extreme Programming in a capstone project," presented at the Proceedings of the Sixth Australasian Conference on Computing Education - Volume 30, Dunedin, New Zealand, 2004.

[5] T. Smith, K. M. L. Cooper, and C. S. Longstreet, "Software engineering senior design course: experiences with agile game development in a capstone project," presented at the Proceedings of the 1st International Workshop on Games and Software Engineering, Waikiki, Honolulu, HI, USA, 2011.

[6] D. Rover, C. Ullerich, R. Scheel, J. Wegter, and C. Whipple, "Advantages of agile methodologies for software and product development in a capstone design project," in 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, 2014, pp. 1-9.

[7] J. G. Kuhl, "Incorporation of Agile Development Methodology into a Capstone Software Engineering Project Course," in The 2014 ASEE North Midwest Section Conference, 2014, pp. 1-8.

[8] D. Ding, M. Yousef, and X. Yue, "A case study for teaching students agile and scrum in Capstone course," J. Comput. Sci. Coll., vol. 32, no. 5, pp. 95-101, 2017.

[9] C. Coupal and K. Boechler, "Introducing agile into a software development Capstone project," in Agile Development Conference (ADC'05), 2005, pp. 289-297.

[10] V. Devedzic and S. R. Milenkovic, "Teaching Agile Software Development: A Case Study," IEEE Trans. on Educ., vol. 54, no. 2, pp. 273-278, 2011.

[11] D. F. Rico and H. H. Sayani, "Use of Agile Methods in Software Engineering Education," in 2009 Agile Conference, 2009, pp. 174-179.

[12] D. Knudson and A. Radermacher, "Updating CS capstone projects to incorporate new agile methodologies used in industry," in 2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T), 2011, pp. 444-448.

[13] R. T. d. Souza, S. D. Zorzo, and D. A. d. Silva, "Evaluating capstone project through flexible and collaborative use of Scrum framework," in 2015 IEEE Frontiers in Education Conference (FIE), 2015, pp. 1-7.

[14] A. El-Abbassy, R. Muawad, and A. Gaber, "Evaluating Agile Principles in CS Education," International Journal of Computer Science and Network Security, vol. 10, no. 10, pp. 19-29.