FLIPPING THE SCRIPT ON PROJECT MANAGEMENT IN EDUCATION: THE OUTCOMES OF APPLYING AGILE DEVELOPMENT METHODOLOGIES IN A CLASSROOM SETTING

Saif Abid, Maxim Antipin, and Hamid S. Timorabadi
University of Toronto Faculty of Applied Science
Saif.Abid@mail.utoronto.ca, Maxim.Antipin@mail.utoronto.ca, h.timorabadi@utoronto.ca

Abstract *– In this paper, an application of Agile Development Methodologies (ADM) to university project oriented courses is presented. A multidisciplinary student team applies Waterfall and Agile (Scrum) project management strategies over a period of 10 months to a project based capstone course. The study primarily focuses on evaluating the two methodologies across five categories – student confidence, student awareness, stakeholder confidence, stakeholder awareness, and project success. Results suggest that overall Agile can be more effective than Waterfall. Due to practices such as daily standups and frequent sprint planning, the student team and stakeholders found they were not only able to stay up to date with the progress of the overall project but also found there was enough time allocated to address the ever-changing nature of requirements brought on by the project. The lessons learned and recommendations provided in this study are generalized such that they can be applied in other project based courses as well.*

Keywords: Education, Agile, Capstone, Design project.

1. INTRODUCTION

The modern education system often makes use of projects and assignments as a means for educating students beyond lectures. These projects predominantly follow a waterfall model: a highly structured linear approach with well-defined deliverables. The simplicity along with the relatively short timelines imposed by the constraints of the curriculum are contributing factors in proceeding with waterfall approach. However, the waterfall model comes at a cost: ***a)*** students are not provided with an opportunity to play any major role in the management of the projects since deliverables are often predefined; ***b)*** feedback is ineffective and does not positively impact the learning process since it is typically provided only after project completion; ***c)*** and the short timelines do not provide students with the opportunity to face challenges that compound in to more complex issues as a function of time.

In the Canadian university engineering curriculum, one course that does not share any of these typical parameters is the capstone design project [1]. This is a project-oriented course offered during the 4th year of the engineering program, and thus culminates 4 years of engineering education. In this course, students form small teams and work with industry professionals and/or supervising professors over a span of 8 months to tackle open ended problems. Unlike other courses in undergraduate studies, design project provides students with an opportunity to independently self-organize, architect, and implement a novel solution. Project deliverables are not always well-defined, there are potentially major changes in requirements throughout the project, and the timeline is long enough for both technical and non-technical challenges to compound in to more complex issues as a function of time.

Stakeholders, students and instructors, spontaneously retreat to the earlier training and apply the waterfall model to implement capstone design projects. The indeterminate, non-linear nature of capstone projects highlights the shortcomings associated with the waterfall model that are not immediately evident in other coursework. Common issues encountered in capstone projects include infeasible technical designs, inability to cope with unforeseen constraints, and general failure to deliver a functional solution by the deadline. Capstone projects that are delivered on time are sometimes met with disappointment by the stakeholders, as a lack of continuous feedback throughout the 8-month process creates disconnects among various expectations and project outcomes.

Industry faces similar challenges with the waterfall model: in 2009, researchers published a case study analysis of Ericsson AB applying Waterfall for large scale development [2]. The findings indicated that there were significant challenges with the requirements and verification phases of the project [2]. To address these types of challenges, industry has adopted Agile Development Methodologies (ADM) as the preferred project management, for instance, Ericsson AB switched over to ADM [2]. ADM emphasizes an iterative style, with feedback based on

continuous delivery of functional components. ADM is usually not taught or used in traditional curriculum, leaving students ill-prepared for both their capstone projects and their post-graduate endeavors.

This paper explores the flexibility afforded by the capstone design project course to flip the norm on how the project was managed and executed. Instead of a typical waterfall model, the researchers followed ADM principles to evaluate the efficacy of industry proven methodologies in a classroom setting. Although ADM is designed to be applied to software projects [3], an application of the core principles to a multidisciplinary capstone project is reported that incorporated software, hardware, and product design with a real-world application.

Studies have reported applying Crystal and Scrum methodologies to cross-functional teams [4] [5]. While these studies have highlighted some of the advantages and disadvantages of applying ADM in an educational setting, this paper takes the approach of a single group applying both Waterfall and Agile methodologies for the two halves of the capstone design course (A simplified model of Scrum for the Agile half) and reports on the lessons learned by comparing the outcomes of both approaches.

The motivation of this study is to expand the knowledge-pool on project management strategies for project-based courses and provide cross functional teams with a comparison of the effectiveness of agile methodologies as opposed to waterfall, so stakeholders can make an informed and educated decision when choosing management methodology.

This study: ***a)*** provides guiding principles on how ADM can be brought into the classroom through a lesson learned approach, ***b)*** demonstrates the improvement of capstone project outcomes in a classroom environment when following ADM practices as opposed to a standard waterfall model, and ***c)*** shows how applying ADM in a classroom setting can better prepare students for their post-graduate endeavors and the demands imposed by the industry.

2. COMPARING WATERFALL AND AGILE SCRUM IN A PROJECT BASED COURSE

2.1. Current State of Waterfall and Agile Scrum

Waterfall is a project management methodology first introduced in 1970 by Winston W. Royce and is based on the principles that a project should first be thoroughly planned and only then the execution should begin [6]. Generally speaking, Waterfall can be looked as being the sequential application of the following 5 steps: Requirements and Analysis, Design, Development, Testing, Evaluation [7] [8]. The requirements and analysis phase is used to lay out the full requirements as well as an analysis of the project landscape/domain knowledge for the project at hand. Due to the nature of waterfall, once this phase is complete, it is not supposed to be altered. The design phase is used as a bridge between the project requirements and the soon to be developed product. During this phase, a detailed technical architecture is put in place as well as technical choices are decided upon. This may include everything from the infrastructure provider to the equipment used for the project development. The development phase is where the implementation of the project takes place [8]. During this phase, the software is coded. Furthermore, during this phase, the development team ensures that the technical architecture in the design phase is implemented to specification. The next is the testing phase. In this phase, the developed product goes through more rigorous testing practices. This means the product is subject to end-to-end and integration level testing. The last phase of a waterfall is evaluation or production phase. In this phase, the project is production ready and leaves the developer or staging environments. This phase also includes monitoring and maintenance best practices from the development team [8].

Waterfall is both a time and industry tested strategy. In 2002, researchers Phillip Laplante and Colin Neill of Penn State University got about 200 responses spanning across multiple industries to their survey regarding the use of various waterfall methodologies [9]. The responses showed that the most frequently used method is, in fact, waterfall with over 33% of reported respondents [9]. The researchers also mention that this accounted for thousands of projects over the 5 years leading up to the survey [9]. It is important to note that this survey was conducted in 2002. Since then, another project management strategy known as Agile has also become more widely accepted.

In 2001 the “Agile” manifesto was released [3]. The Agile manifesto sets in place guiding principles for software development projects to follow to ensure high quality end results while keeping the customer of the product as the key stakeholder [3]. The core difference between Waterfall and Agile methodologies is that Agile is based on iterative/incremental development with a very short and continuous feedback loop with key stakeholders. A known and popular implementation of Agile is the Scrum framework - a specific guideline on how to apply the principles of agile to a project lifecycle [10].

The scrum framework is based on 5 fundamental phases - sprint planning, sprint, daily scrum, sprint review, and sprint retrospective [11] [12]. Figure 1 shows the order of phases and their feedback cycle as provided by scrums official website: scrum.org.

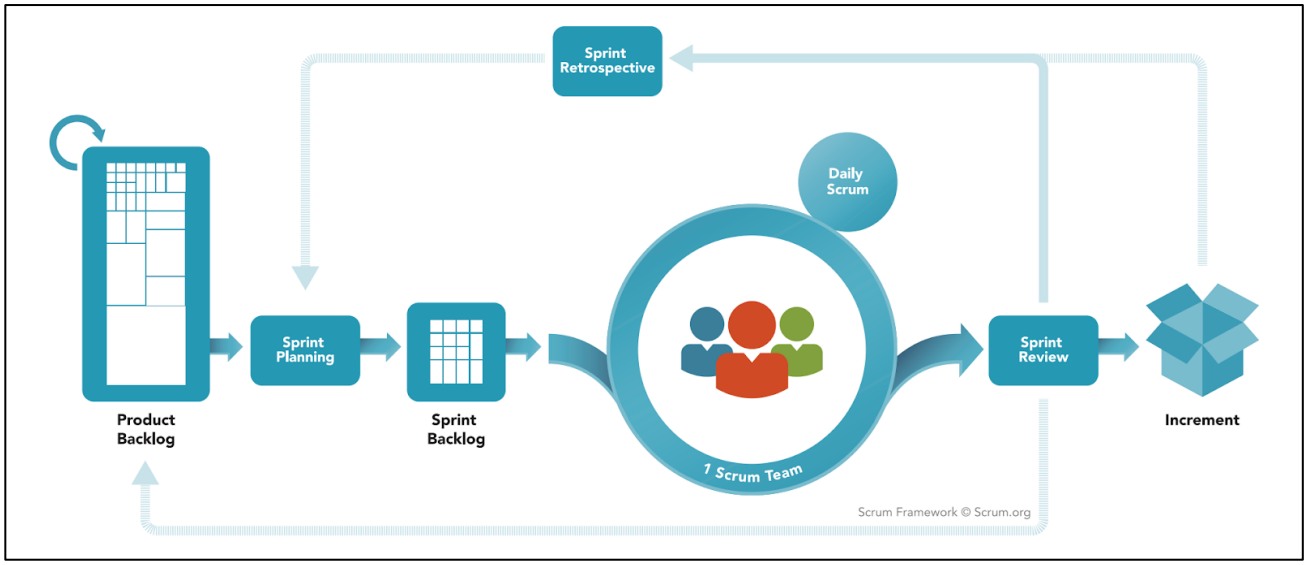

Fig. 1. The scrum framework processes and feedback loop [11].

During Sprint planning, the team selects prioritized tasks from the product backlog and puts them into the sprint backlog. The product backlog can be thought of as a curation of all tasks required to get the project done [12]. The product backlog grows as more information becomes available about the project throughout its development. Tasks (i.e. user stories) are identified and inserted into the sprint backlog. In the meantime, teams work together to identify the complexity of the task. This allows teams to have a quantitative metric overtime to measure their sprint performance. Once tasks are set into a sprint backlog a sprint is considered as started. A sprint's duration is meant to be long enough so that incremental progress can be made on the final product but short enough that too much is not completed without circling back with stakeholders or adjusting requirements if need be [12]. A sprint is typically 2 weeks in duration [10]. During each day of the sprint (the period in which work is being completed from the sprint backlog), a daily update (known as a scrum update) is provided by teammates to inform other members of the task they are working on. Typically, this means answering 3 questions: "What did I do yesterday", "What will I do today", "Is there anything blocking my progress" [12]. This allows the team to understand early where challenges are being faced and to address them early. Once a sprint is complete, the team enters a sprint review phase. In this phase the team reviews the tasks of the sprint that are completed and those tasks that are not completed are put back into the product backlog in addition to any new tasks. The last phase of the scrum process is known as sprint retrospective. During this phase, teams identify tasks that went well and those that did not in the process of sprint. Then an action plan is formed to ensure the team's progress while minimizing any future roadblocks [12].

Like Waterfall, Scrum is very much industry tested and ready. The State of Scrum 2017-2018 survey which is based on over 2000 members, reports that scrum is used across 91 countries, spanning 27 industries and is used not only in technology departments but spans departments such as accounting, marketing and human resources [10].

As in industry, waterfall and scrum can be seen in the way courses are designed and delivered in university environment. Typical courses which include a lab or project-based component usually have a requirements and deliverables laid out for the students from the start. The student or student teams then develop, test and then submit their project. There is often not enough time in the curriculum or in the timeline to allow students to iterate and perform such courses in an agile fashion.

There are however a few courses which go against the grain and deliver their material in such a way to allow students the flexibility to choose their own project management style within their teams. In the University of Toronto's Electrical and Computer Engineering department courses which give students the flexibility to perform self-chosen project management practices are APS112: Engineering Strategies and Practices 2, CSC444: Software Engineering 1, and lastly ECE497: Capstone Design Project. These 3 courses are highlighted due to the breadth they cover in their industry of application. In APS112, the teams are multi-disciplinary and projects need not be software or hardware but rather they can span among many different disciplines of engineering (Civil, Industrial, Chemical, etc.). As this is a first-year course, there is more emphasis on the design then the actual implementation and evaluation of the project. In CSC444, the teams and project are purely software. Lastly, ECE497, is particularly interesting because, the team is comprised of all electrical and computer engineers and

projects also need not be only software or purely hardware. The Capstone project mimics very closely how an industry project is completed. The course allows students to work with a faculty member on a completely novel idea for the entire course year. This forces students to not only wear the hat of the engineer, but also the product managers and marketing - as they not only have to design, build, document and demo the project end to end but they also must continuously keep their stakeholders in the loop as well their end users.

As such, the ECE497 capstone project course pledges a very interesting opportunity to be used as a sandbox to experiment using a single baseline course the effectiveness of both waterfall and scrum in a multidisciplinary team. The conclusions of this experiment, in theory can then be brought back to other courses which have stricter guidelines on project management strategies and such courses can be re-evaluated if needed accordingly.

2.2. Methodologies

In this analysis, the students under examination, worked on a multidisciplinary capstone project involving multiple software and hardware modules.

The project goal was to architect and prototype an indoor navigation system for the visually impaired. The four main modules were: a geo-processing unit consisting of a wearable tag and 3 ranging anchors for trilateration of the user; an orientation sensor for determining the orientation of the user; natural language processing component to enable voice commands; and an augmented audio reality module that generated a virtual 3D positional sound. Figure 2 depicts the system overview of the project.

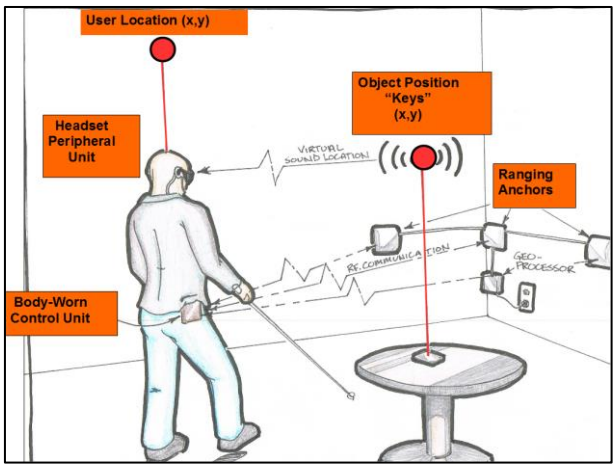

Fig 2. The Project's System Overview.

The students had a wide breadth of academic interests and came from a variety of industry experiences. Table 1 summarizes students' industry experience and academic interests. Two students were responsible for the architecture and development of software components, i.e. the software team. The other two students focused on the hardware components, i.e. the hardware team.

Table 1: Summary of the industry experience, academic interests, and project responsibilities of the students.

Student	Project Responsibility	Industry Experience (min. 1 year)
A	Software	Distributed Systems & Data
B	Software	Product Management
C	Hardware	Control Systems
D	Hardware	Engineering Economics

Only one student had exposure to agile methodologies prior to the experiment; the remainder of the students had worked primarily under waterfall or agile-waterfall hybrid project processes. The total duration of the project was 10 months: phase 1 constituted the first 5 months using a waterfall approach. Phase 2 constituted the next 5 months and incorporated agile methodologies.

Phase 1 began with a series of project planning meetings, with the goal of defining the project requirements followed by the complete architecture of the entire system. These meetings included research about existing solutions, material planning, selection of solution stack, communication protocols, distribution of work across team members as well as internal milestones. The schedule and timeline considered course deadlines and was approved by the supervising professor. The final milestone for phase 1 was a minimal viable prototype, with the majority of the software and hardware components integrated together.

Phase 2 adopted "Scrum" as the agile framework. There were four major activities performed within this framework: sprint planning, daily scrum, sprint review, and sprint retrospectives [12]. The methodology of each activity conducted is as follows.

Sprint planning:

Sprint planning was performed in a weekly cadence for the software team and biweekly for the hardware team. The purpose of a sprint planning activity was to set weekly milestones, and any specific tasks required to meet them. All tasks were quantitatively and qualitatively assessed to determine if they were too broad in scope. In case a task required more than one week to complete then that task was further scoped down. The assessment metric considered was "estimated number of hours to complete". After a sprint planning activity, tasks were distributed and assigned per each student's workload, strengths and weaknesses. The sprint planning meetings were only attended by the students and not the supervising professor.

Daily scrum:

Students participated in the daily scrum on daily basis. Regardless of whether any progress was achieved on a task, each student answered four questions:

- What was achieved yesterday?
- What are the plans for today?
- Is there any task that I need assistance with?
- Is there anything that is hindering progress of my work?

Sprint review:

Sprint review was done at the end of every sprint. During the review, students assessed whether all the tasks for the sprint have been completed. Then students addressed any new changes to the requirements based on new information and/or research since the last sprint.

Sprint retrospective:

The sprint retrospective was a short meeting usually held after a sprint review. During this meeting students discussed amongst each other the difficulties and challenges faced in the previous sprint. The goal of this retrospective is to put together an effective plan for addressing these challenges in future sprints.

Furthermore, during the entire 10-month period, there was a weekly update provided to the supervising professor about the overall progress of the project. During these meetings, each student answered 2 survey questions, outlined in Table 2. These two questions were chosen to assess the positive and negative project outcomes the students wanted to achieve and avoid, respectively. Some of the desired project outcomes included increased project confidence, higher team efficiencies, reduction in project risk and failure, while avoiding knowledge gaps and information silos. The results from these weekly surveys would provide a method of assessing the effectiveness of the different project management approaches employed by the students.

Table 2: Survey questions filled out by student's weekly

Am I confident the project is on track for completion?
Am I aware of the work the other members on the team are doing?

3. RESULTS

The results and trends of the survey questions across the Waterfall and Agile phases of the project are shown in Figures 3 – 6. Also, aggregated results are shown by category of interest in Table 3.

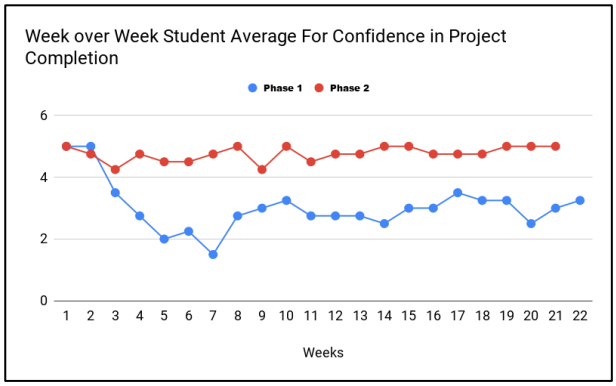

Fig. 3. Weekly average survey results for students' level of confidence that the project will be completed successfully; 0 to 6 scaling.

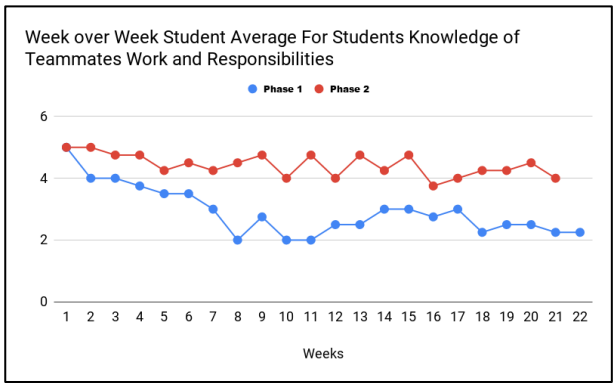

Fig. 4. Weekly average survey results for students' level of knowledge on what other members of the team are working on and are responsible for; 0 to 6 scaling.

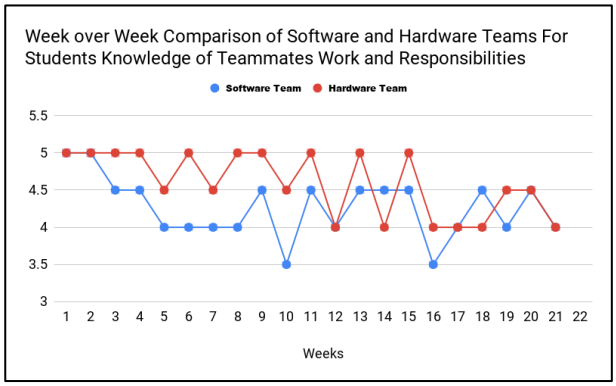

Fig. 5. Weekly average rating for hardware and software teams in phase 2 illustrating students' level of knowledge on what other members of the team are working on and responsibilities; 0 to 6 scaling.

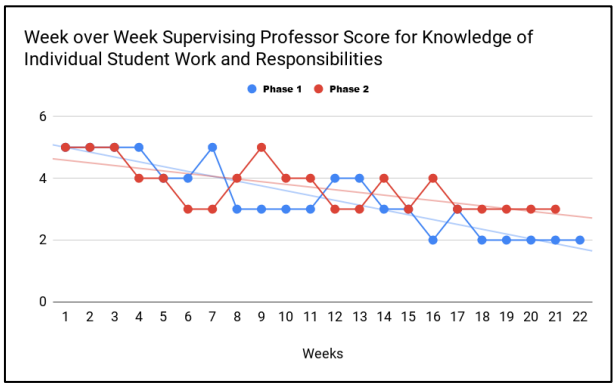

Fig. 6. Weekly scores from the supervisors regarding individual students' knowledge, work, and responsibilities; 0 to 6 scaling.

Table 3: Aggregated metrics across four categories of interest for phase 1 and phase 2 of the project.

Category	Phase 1 Waterfall	Phase 2 Agile
Student Confidence	Average score of 3.02 with a variance of 10.18.	Average score of 4.76 with a variance of 0.95.
Student Awareness	Average score of 2.91 with a downwards trend as the project progressed.	Average score of 4.43. Average software team score of 4.26, compared to the hardware team awareness score of 4.60.
Stakeholder Confidence	Average score of 3.09.	Average score of 4.48.
Stakeholder Awareness	Average score of 3.36 with a downwards trend as the project progressed.	Average score of 3.71 with a downwards trend as the project progressed.

To quantify project progress in the two phases, different approaches had to be considered to provide a meaningful comparison between the Waterfall and Agile methodologies.

Phase 1 project progress was quantified by the amount of time the project had slipped from its initial goal of a minimally functioning prototype. It was originally planned to be delivered by the end of phase 1, i.e. week 22. However, by week 22 the project was 8 weeks behind schedule and the updated projection of the technical delivery was week 30. This delay equates to a completion rate of 65% of the planned milestones for phase 1.

During phase 2, the sprint completion rates were considered where all 21 software sprints were completed successfully. Out of the 10 hardware sprints, only 1 sprint was left uncompleted by the hardware team. This leads to a sprint completion rate of 100% for the software team, and 90% for the hardware team.

4. LESSONS LEARNED FROM THE USE OF AGILE AND WATERFALL IN CAPSTONE DESIGN PROJECT

4.1. Student Project Confidence

4.1.1. Analysis: Students had a higher level of confidence in project completion during the Agile phase as opposed to the Waterfall phase. Students became more confident in completing the project and were more consistent in achieving milestones as weeks went by. The higher level of confidence and the consistency of the results may be attributed to the continuous delivery of functional components, sprint reviews and retrospectives.

4.1.2. Recommendation: Schedule and implement continuous delivery of functional components, sprint reviews and retrospectives. These agile principles promote constant, substantive progress, manifesting in a higher level of confidence for project completion amongst the team members.

4.2. Student Project Awareness

4.2.1. Analysis: Students had a higher level of awareness and understanding about other team member's tasks and responsibilities during the Agile phase than the Waterfall phase of the project. This may be attributed to the daily scrums and recurring planning meetings. There was a discernible downward trend in project awareness during the Waterfall phase, which may be attributed to tasks and responsibilities deviating further from the initial planning that had occurred at the outset of the project. The software team had a lower level of awareness and understanding of other team member's tasks and responsibilities than the hardware team during the Agile phase. This may be attributed to the longer 14 day sprint cycles for the hardware team, in addition to the logistics of hardware components not always being accessible to all of the students, while the software was available to all members at any time through source control.

4.2.2. Recommendation: Schedule and implement daily scrums and recurring planning meetings. These agile principles promote higher project awareness and understanding for all team members. Further explore the advantages and disadvantages of varying sprint lengths for cross functional teams.

4.3. Stakeholder Project Confidence

4.3.1. Analysis. The supervising professor had a higher level of confidence in project completion during the Agile phase than the Waterfall phase. The high level of confidence during the Agile phase may be attributed to the

continuous delivery of functional components. The low level of confidence during the Waterfall phase may be attributed to ambitious nature of the project, and the general bias of past capstone projects and their low success rate.

4.3.2. Recommendation: Schedule, implement, and present continuous delivery of functional components for stakeholders. This agile principle promotes a higher stakeholder confidence in project completion, provides continuous feedback for the team, and overall helps set realistic expectations for the final technical delivery.

4.4. Stakeholder Project Awareness

4.4.1. Analysis: The supervising professor generally had a higher level of awareness and progress of each team member's tasks and responsibilities during the Agile phase than the Waterfall phase of the project. However, both phases demonstrated a downward trend in the supervising professor's level of awareness and understanding. The higher level of awareness at the beginning of the Waterfall phase may be attributed to the detailed plan delivered at the outset of the project. The downward trend is then the consequence of the project deviating from the initial plan as a function of time, resulting in a limited understanding for the supervising professor. A similar but less pronounced downward trend in the Agile phase may be attributed to the supervising professor not being actively involved in the sprint planning, daily scrum updates, sprint reviews and retrospective.

4.4.2. Recommendation: Involve all stakeholders during sprint planning, daily scrum updates, sprint reviews and retrospectives. Provide a high level, executive summary to stakeholders about project status after the completion of every sprint.

4.5. Additional Learnings for Agile Teams

Two important tools that were used throughout phase 2 which may have further attributed to the overall improved results included Atlassian's Jira and Slack. Atlassian's Jira is a project management tool which includes the ability for teams to create sprint boards and manage sprint lifecycles such as scoring of task stories, assigning members, and setting sprint deadlines. Slack on the other hand is a team messaging service which allows for group and private messaging with team members. Slack was used in phase 1 and phase 2 of this project. During phase 1, Slack was used in an ad-hoc manner, for any general conversations or discussions relevant to the project.

During phase 2, the usage of Slack was more organized and structured, with dedicated channels for daily scrum updates (meeting minutes or actual conversations when in-person meetings were not possible), and the weekly sprints to ensure tasks were on track for completion.

One of the challenges faced by the students was the nature of the cross-functional team and individual course schedules that did not align and were suboptimal for group work. Online messaging and project management tools helped students overcome these challenges, by enabling efficient remote work at any time of the day. Beyond the classroom, the experience of implementing and using these tools to practice the fundamentals of Agile scrum methodologies enables students to enter industry with relevant experience and ready for modern project management styles.

5. CONCLUSION

By applying Scrum, an implementation of the agile principles, a diverse group of students coming from various academic and industry backgrounds found that they had a higher level of confidence in project completion along with a higher milestone success rate as opposed to a traditional waterfall methodology.

To adapt scrum into project management project based course workflows where there are cross-functional teams, the experiment found the followings. First, having hardware teams perform on a longer sprint cycle than software teams may be beneficial given the incremental development time for the hardware team was significantly longer than that of the software team. Secondly, since each week new goals are set and changes occur incrementally, it is important for student teams to involve the supervisor in daily scrums, sprint planning and sprint review This assists supervisor to stay up to date with the project progress in a finer grained manner.

Acknowledgements

It is acknowledged that despite the successful application of Agile Development Methodologies (ADM) to design projects, a fair comparison has to be conducted between two equally weighted projects. These projects need to have identical goals and requirements. In addition, the teams need to be of equal skill sets and have the same timeline. Waterfall methods can be applied to one project while the other project follows the techniques offered by ADM. Consequently, the outcomes can be fairly compared.

References

[1] University of Toronto (2018). "ECE496 Design Project Course 2017-2018". [Online]. Available: https://internal.ece.toronto.edu/ece496.1718/index.html

[2] Kai Petersen, Claes Wohlin, Dejan Baca, "The Waterfall Model in Large-Scale Development," in *PROFES 2009:*

Product-Focused Software Process Improvement, vol 32, pp.386-400, 2009

[3] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robery C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas (2001). "Manifesto for Agile Software Development". [Online]. Available: http://www.agilealliance.org

[4] Richard Stansbury, Massood Towhidnejad, Jayson F. Clifford and Michael P. Dop, "Agile Methodologies for Hardware/Software Teams for a Capstone Design Course: Lessons Learned," in *2011 ASEE Annual Conference & Exposition*, pp. 22.151.1 - 22.151.13, June 2011.

[5] Martin Edin Grimheden, "Can Agile Methods Enhance Mechatronics Education? Experiences from Basing a Capstone Course on SCRUM", in *2012 ASEE Annual Conference & Exposition, (San Antonio, Texas; 10-13 June 2012)*, 14 pp., 2012.

[6] Winston W. Royce, "Managing the Development of Large Software Systems" in *Technical Papers of Western Electronic Show and Convention (WesCon)*, (Los Angeles, USA; 25–28 August 1970), 382-338 pp., 1970.

[7] Bonnie Shellnut, Allie Knowltion, Tim Savage, "Applying ARCS model to the design and development of computer-based modules for manufacturing engineering courses," *Educational Technology Research and Development*, vol. 47, issue 2, pp. 100-110. June 1999.

[8] Andrew Powell-Morse (2016). "Waterfall Model: What Is It and When Should You Use It?". [Online]. Available: https://airbrake.io/blog/sdlc/waterfall-model

[9] [9] P.A. Laplante, C.J. Neill, "The Demise of the Waterfall Model is Imminent and other Urban Myths of Software Engineering", *ACM Queue*, vol. 1, no. 10, pp. 10-15, February 2004.

[10] Scrum Alliance (2018) "State Of Scrum 2017-2018". [Online]. Available for Download: http://info.scrumalliance.org/State-of-Scrum-2017-18.html.

[11] Scrum.org. "The Scrum Framework Poster". [Online]. Available: https://www.scrum.org/resources/scrum-framework-poster

[12] Ken Schwaber and Jeff Sutherland (2017). "The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game". [Online]. Available: http://scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100