Applying Scrum Project Management Methods in Biomedical and Electrical and Computer Engineering Capstone Design Courses

David Lee[1], Carl Wick[1], and Hernan Figueroa[2]

[1]Biomedical Engineering, George Washington University

[2]Electrical and Computer Engineering, George Washington University

Mar. 15, 2018

Abstract

The Biomedical engineering and Electrical and Computer Engineering Departments at George Washington University have traditionally used a waterfall project management methodology for their two and three semester capstone design courses. We noticed that this approach resulted in incomplete senior design projects. Our analysis of these failures showed two major problems. First, students do not have experience with how things are made. So, they are unable to produce the detailed designs required by a waterfall planning scheme. Second, they are afraid to start building subsystems, so they delay building until the last moment. This leaves student teams without time to fix failures, revise their plans, and integrate components. So, we chose to utilize an *agile* project management technique used extensively in the software industry. We implemented a variant of *scrum project planning*, which is based on starting with a top-level design, start to build subsystems for that design, and modify your design as you learn. It is a structured and supervised try–fail–fix approach to project management that we believe is well suited to capstone design. Our first semester modifications were generally well received by faculty and students. We compared the state of designs from previous years, which should be complete, with those submitted this year, which are not expected to be complete. We found that, overall, this year's designs are as complete as previous designs. However, teams from this year have completed several parts of their projects, while previous teams would only have started building a subsystem.

KEYWORDS

Capstone, Project Planning, SCRUM, Agile

Introduction

Biomedical Engineering (BME) and Electrical and Computer Engineering (ECE) departments at George Washington University have historically used waterfall project planning models[1] as the basis for full year capstone design courses. Waterfall models are predicated on student teams completing a detailed design plan before starting any prototyping. A drawback of this choice is that at the end of three successive Capstone Project Lab courses, about 40% of our students were not completing successfully their projects. Two factors seem to underlie the failures. First, students limited experience with real life project management resulted in plans that did not contain sufficient detail for successful execution. Second, course emphasis on early, detailed planning during the first semester left little time for students to learn how to build and test functional devices during the second semester. In response to these problems, we examined our existing course structure, compared it to the way faculty typically approach design problems, and decided that adopting an industry accepted *agile* planning approach—*scrum*[2] in particular—would be a worthwhile year-long capstone experiment. We felt that this methodology would force students out of their risk-averse "research until we know everything mode, and into a structured and supervised try–fail–fix mode used by innovative companies like Blue Origin.[3] We were interested in implementing two agile principles within a scrum project management approach: a) prioritizing development of a working prototype over comprehensive documentation and b) encouraging student responses to challenges. Our impetus for selecting scrum planning is that the software industry makes successful and extensive use of scrum management[4,5,2]. In addition, scrum project management has been adopted in capstone design courses in computer science and software engineering[6,7], mechanical engineering,[8] and ECE[9]departments at other universities.

Both waterfall and scrum project management strategies start with needs and vision statements, which are refined into requirements and specifications for the system as a whole. In a waterfall strategy, the system is then divided into subsystems, and requirements and specifications are developed for those subsystems. Only after full specifications are complete, are design decisions made starting at the system level and then progressing down through each subsystem level. Prototyping and its associated learning does not start until all subsystems are designed. In contrast, scrum planning starts with top level requirements for the system. A top level preliminary design based on those requirements is created. Then, a list of tasks—called a product backlog— required to prototype the preliminary design is made. The product backlog is organized from high priority—critical and difficult—to lower priority—nice-to-have and easier—tasks. From the product backlog, a teams product owner selects the most critical tasks and arranges them as smaller products that can be built, tested and demonstrated in a 2-4 week time frames called *sprints.*

The scrum team then implements a selected product function in a single sprint. The team goal for each sprint is to produce a potentially complete piece of the final device. At the end of a sprint, the team presents their completed product to the product owner (advisor) and, if applicable, clients. Feedback provided during this sprint review is used by the team and product owner to revise the needs and visions statements. This revision includes adjustments to the system requirements based on on what the project team has learned. The team and product owner then update the product backlog. This process is repeated until the design is fully implemented and tested. We decided that this framework would push our teams to test their ideas early, useif not embracea structured try-fail-fix approach, and incorporate client feedback in their design revisions.

PREVIOUS COURSE STRUCTURE

BME has a two semester capstone sequence while ECE has a three semester sequence. In both cases, open ended design projects were either assigned to or created by student teams with instructor guidance and approval. Previously, the first semester of BME and the first two semesters of ECE capstone design were primarily devoted to planning. Initial construction and testing of a subsystem was started during the second semester in the ECE course. BME students selected a critical component and demonstrated that they could use that component at the end of their first semester. In each case, the last semester was used for prototype, test, and refinement. Each year BME typically fielded about 10 four or five person teams, while ECE typically had 5 or 6 three person teams. An overview of these courses is presented in Table-1.

Initial Semester(s)		
Weeks	Task	Comments
1–4	Needs Assessment	BME teams assigned projects
		BME and ECE teams create detailed
		requirements and specifications
4–7	Initial Designs	Teams produce an overview design
8–10	Subsystems Defined	Teams produce subsystem needs and specifications
10-14	Create Design Documents	subsystem designs completed
	initial subsystem prototype	Teams build part of a subsystem
15	Final Design Review	Teams present their detailed designs
Final Semester		
1–8	Prototype Subsystems	Teams start to build
9–12	Assemble Subsystems	Assemble (nominally) tested
		subsystems into complete system
13–14	Test Complete System	Teams complete testing and fix problems
15	Final Presentation	Teams present prototypes

Table 1: Typical schedule for our waterfall based course

In our waterfall model, each team first performed a needs assessment and produced a requirements and specifications document. Next, they brainstormed and created an initial high level design, then refined elements of the high level design by defining subsystems. Our observations of this work is that it is a reasonable starting point, but lacks essential details and provides limited experimentation and familiarization with the hardware/software they would eventually use. Waterfall teams were expected to build, test, and refine their designs during the last semester. They started by building and testing subsystems, then integrated their subsystems into their overall device, and finally, tested their overall device against their specifications. In practice, they spend much of this last semester adding details they missed when planning their subsystems. They discovered missing pieces when they tried to build those subsystems. Mentors report meetings where they tell students to do a task one week. The next week, students report that they have a new design for the subsystem associated with that task, but have not started the task. This repeats until the fear of failing the course overcomes the fear of building something they do not know will work on the first try. Often this cross-over occurs around week 10 of a 14 week semester. As a result, many teams are testing their subsystems for the first time in the last week of the final semester. They do not have time to integrate their subsystems into a complete device, and then test that device against their specifications.

SCRUM BASED COURSE STRUCTURE

We developed our scrum based course by aligning major scrum roles[4] with, as much as possible, parallel roles in our existing courses. For example, BME projects come from clinicians and faculty researchers, so BME teams have external clients. ECE, on the other hand, primarily relies on student generated projects, so the team is also the client. Both departments have a faculty course director and team mentors. In scrum they act as product owner and scrum master, while students adopt the scrum team role. We opted to incorporate four three–week–long sprints in each semester. We selected three week sprints, because, given student course loads, two weeks would not allow enough time for try-fail-fix cycles. We chose not to use four week sprints, because we thought students would procrastinate. Table 2 is the schedule for the first semester BME and second semester ECE course.

During each sprint each team met with their mentors, or scrum master manager, weekly. ECE students and their mentor agreed on a set of features (from the product backlog) that should be demonstrated at the end of a Sprint. Similarly, BME students and their mentors selected tasks from an informal scrum backlog that were necessary for successful completion of the scrum product.

At the end of each sprint, each team was expected to have completed some tangible portion of their project. They were then required to present what they had accomplished to the larger group and to faculty advisors and clients. In the BME course, product demonstrations were done during class time in a conference exhibit hall format. Feedback from both peers and instructors was provided to teams in both ECE and BME classes. Further, each team also wrote a sprint review and updated the (running) design documents and project backlog at the end of each sprint. Our course does not precisely match a typical scrum format. One difference between our approach and standard scrum methods is that we assigned students some product owner responsibilities, because they need experience defining and managing projects. For example, students rather than the product owner created and, after each

Weeks	Task	Comments
1–2	Sprint I	BME teams assigned projects and define needs.
		ECE teams create first Sprint Product
3–5	Sprint II	Teams produce sprint product
5	Sprint II Demo	Teams demonstrate products in a trade-show format
	Revise Design Documents	update needs and product backlog
6–8	Sprint III	2nd product from revised backlog
8	Sprint II Demo	Teams demonstrate products in a trade-show format
	Revise Design Documents	update needs and product backlog
9–11	Sprint III	3rd product from revised backlog
11	Sprint II Demo	Teams demonstrate products in a trade-show format
	Revise Design Documents	update needs and product backlog
12–14	Sprint III	4th product from revised backlog
14	Sprint II Demo	Teams demonstrate products in a trade-show format
	Revise Design Documents	update needs and product backlog
15	Final Presentations	current state + plan for next semester

Table 2: The schedule for our first semester scrum based course

scrum, revised the product backlog. Another difference is that we replace the daily scrum meeting with a weekly meeting. We do, however, suggest that each team-member should do a “daily mental scrum meeting” by asking themselves: “what must I do today, so my team has a complete product at the end of the sprint?”

FIRST SEMESTER SCRUM EXPERIENCE

The BME and ECE students had somewhat different first semester experiences with scrum management. BME students bid for designated projects, and were assigned to teams based on those bids. In contrast, ECE students had created their own projects and teams during the preceding spring semester. In practice, this meant that the ECE students started the semester with a sprint. The BME students started by doing a preliminary needs assessment. We altered the timing of their first sprint so that both groups would be on the same schedule for sprints II-IV. A problem resulting from our choice to have teams create their own product backlog is that teams had difficulty selecting their first sprint product. Some teams selected products that were much too simple, while others tried to tackle overly complicated or undoable products. We encouraged the mentors to provide more guidance after the first sprint. As the semester went along, we started noticing that teams were not including an overview of their project. They added tasks to the product backlog, but they neither connected those tasks to their vision nor indicated how results of their sprints altered their plans. So, we revised the sprint review assignment to include a specifications table and product overview into their sprint reviews. During both semesters we continually emphasized that they were not to be creating new documents; they were merely to revise documents they already generated. One particular aspect of scrum management that is giving our students problems is the definition of when a sprint is “done. Students often answer this question with vague, non-testable statements. For example, one team wrote: Done, when referring to a backlog item, will mean that we have a working, tested product that we are comfortable leaving in its current state given its intended purpose and we are ready to move on to the next item in the backlog. Notice that this teams definition of done does not have any reference to specific tasks or products. Further, there is no unbiased, quantitative test for the statement we are comfortable. This difficulty with making testable statements is also reflected in errors students make with specifications. In place of numbers, they use solution descriptions and/or vague statements. Continued classroom emphasis on testing to verifiable specifications helped to produce better and verifiable definitions of done for sprints. One of the key changes we will made between the first and second semester is to have mentors provide more direct control over the product backlog and task selection. For example, each team will select sprint products and write definitions of done with direct input from their mentor. Only once the mentor approves that document, will it be forwarded to the course director to grade. We anticipate that this will result in better defined and appropriate sprint products, which we hope will result in more functional sprint products.

ASSESSMENT

A common assessment tool for courses like this is student and mentor surveys. While both formal and informal feedback from students is valuable, it is limited by the experience of students. They may really like or hate the format compared to their other courses, but, for the most part, this is their first design experience, and they do not have a baseline for comparison. Hence, we are interested in adopting quantitative assessment tools of student performance.

One of the challenges is that the most critical skills they are learning are not suited to typical summative assessments. We want to know if our students have: developed good team work habits, learned to take intellectual risks, can perform critical self-assessments, and use technical knowledge from other courses. Another hurdle to quantitative analysis is that we do not yet have baseline student performance data for the waterfall based capstone programs. However, we can do retrospective studies of final design documents produced by previous student teams.

While it is too soon to have quantitative results, we have been very pleased with student and faculty acceptance of this new capstone project approach. In virtually all cases the projects appear well along in a real, tangible sense compared to previous years. Further, most students seem to enjoy the activity and idea of achieving a portion of their project each sprint. There were a few ECE students who felt our approach to scrum was too chaotic. However, most BME and ECE teams appear significantly ahead in having a clear vision for completing their project when compared to previous years. They now appear much more comfortable in their project decisions and in demonstrating what they have accomplished to this point.

References

[1] *ISO/IEC/IEEE 24765:2017(E) - ISO/IEC/IEEE International Standard - Systems and software engineering–Vocabulary.* 2017.

[2] *The SCRUM Guide: The Definitive Guide to SCRUM: The Rules of the Game.* 2017. URL: `http://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100`.

[3] *Lapsa, Andrew. personal communication.*

[4] Chris Sims and Hillary Louise Johnson. *Scrum: A Breathtakingly Brief and Agile Introduction.* 2012.

[5] Kenneth S. Rubin. *Essential Scrum. A Practical guide to the most popular agile process.* Upper Saddle River, NJ: Addison-Wesley, 2013.

[6] Viljan Mahnic. "A Capstone Course on Agile Software Development Using Scrum". In: *IEEE Transactions on Education* 55 (2012).

[7] Diane Rover et al. "Advantages of agile methodologies for software and product development in a capstone design project". In: *IEEE XPlore. Frontiers in Education Conference (FIE), 2014 IEEE* (2015).

[8] Martin Grimheden. "Can agile methods enhance mechatronics education?: Experiences from basing a capstone course on Scrum". In: *ASEE Annual Conference and Exposition, Conference Proceedings, American Society for Engineering Education* (2012).

[9] Robert B. Bass, Branimir Pejcinovic, and John Grant. "Applying Scrum project management in ECE curriculum". In: *Frontiers in Education Conference (FIE), 2016 IEEE* (2016).

David T. Lee
Dr. David T. Lee is an Associate Professor of Practice in Biomedical Engineering at George Washington University. He is the course director for the BME capstone courses. In that role, he is developing collaborations to create assistive and adaptive devices with faculty in Physical Therapy. His other responsibilities include developing experiential curricula for BME courses and teaching Introductory BME courses for first year students. He earned his Ph.D. degree in Experimental Condensed Matter Physics from Ohio State University.

Carl E. Wick
Dr. Carl E. Wick is currently a Professional Lecturer with the George Washington University Bioengineering Department where he assists with student capstone projects in and out of the classroom. He is a Professor Emeritus of the U.S. Naval Academy where he previously held a Systems Engineering professorship, was a Department Head, and was the Associate Director for Midshipman Research. He has a D.Sc. degree in Computer Science from the George Washington University.

Hernan Figueroa
Dr. Figueroa was the course director for the first and second semester Electrical and Computer Engineering capstone courses in 2017 at George Washington University. His research focuses on optimization algorithms and data-driven applications. Prior to joining GW, Professor Figueroa was an associate research scientist at the Sustainable Engineering Lab at Columbia University. He earned his Ph.D. degree in Electrical Engineering from the University of South Carolina.