Agile/Scrum for Capstone Project Management

Alisha Sarang-Sieminski and Rebecca Christianson
Olin College of Engineering

Agile/Scrum is a philosophy and methodology for improving performance on complex projects. While largely used in software development, this approach to project management can be applied to projects in any field. Agile/Scrum focuses on transparency, adaptability, individual team member autonomy, accountability, and continuous improvement. We have implemented a version of Agile/Scrum in Olin College's senior capstone course. In this paper, we provide an overview of the mechanics of Scrum as a project management tool and discuss its implementation in a design capstone context. Our goal is to inform other Capstone programs, so they can assess whether Scrum is an appropriate tool for them and learn from our experience.

Keywords: Scrum, Project Management, Accountability

Corresponding Author: Alisha Sarang-Sieminski,alisha@olin.edu

Overview of Agile/Scrum

Scrum is one of the most commonly used methodologies for implementing an Agile philosophy to execute projects.[1,2] Agile is a philosophy for managing complex projects, which was developed in the 1990s by software developers and draws upon "lean" manufacturing.[1,2] The Agile Manifesto states that Agile values "Individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, [and] responding to change over following a plan."[3] The idea is to promote a more "lean" method of doing complex projects than more traditional "heavyweight" waterfall method used widely in software and engineering.[4] Key aspects of Agile/Scrum are iterative development, collaborating with customers with evolving needs, and cross-functional teams with a high degree of autonomy.

In traditional Scrum, the key roles are the Product Owner and the Scrum Master.[1,4] The Product Owner holds the vision and often holds a position of authority. They are responsible for managing and prioritizing tasks in the Product Backlog: a list of discrete tasks with a clear definition of done and a team-wide understanding of their size. The Scrum Master is responsible for making sure the team is enacting Scrum properly and for removing impediments (either internal or external). The other members of the team are self-organizing in deciding how to get the work done.[1] The high level of autonomy and an emphasis on individual accountability are intended, and demonstrated, to improve morale, motivation, and productivity.

In Scrum, work is broken down into short "Sprints" of ~2 weeks.[1,4] Each sprint has several ceremonies to give them structure, bring clarity, display work, get feedback, and reflect on the team's productivity.[1] First, a Sprint Planning Meeting is held at the beginning of each Sprint. Here, a Sprint Goal is articulated, likely by the Product Owner, and tasks to be completed during the Sprint are moved by the team from the Product Backlog to the Sprint Backlog. Work is "kept visible" by displaying it on a sprint board, either electronic or physical, on which each task is tagged as To Do, Doing, or Done. There is a strong emphasis on having a "minimal shippable product" at the end of each sprint.

At the end of each sprint are a Sprint Review, which is publicly facing, and a Retrospective, which is internally facing.[1] At the Sprint Review, the team presents work that was completed during the sprint to stakeholders. In the software context, the team should present a demo of their working software to the customer or other company stakeholders to get feedback and understand evolving customer needs. Only the team attends the Retrospective. Here, the team reflects on what went well and what did not in the most recent sprint. They identify actionable items to improve their function as a team (sometimes called the Kaizen). Again, the retrospective keeps the team focused on individual accountability in team functioning, task completion, and communication. It is intended to promote a level of self-awareness.

In addition to these larger ceremonies, Daily Stand-ups are held in which each team member briefly reports on tasks they are working on or have completed since the last stand-up.[1] While brief, the Stand-ups allow team members to understand what everyone is working on, identify and quickly deal with impediments to progress, and cross-pollinate ideas.

Another key aspect of Scrum is the concept of "timeboxing" ceremonies or other tasks.[1] It is the simple idea that a time limit is set to accomplish the ceremony and it is important to adhere to that limit. Like many concepts in Scrum, this is not novel or limited to Scrum, but it is a good practice.

Implementation in Capstone

Project management is taught in some form in virtually all capstone programs. Based on a survey of published information about capstones, it appears that most use a traditional waterfall method.[5,6,7,8] While there are a handful of reports of using alternate approaches, those using Scrum are primarily Software capstones.[2,9,10] In one report comparing (non-software) capstone teams using traditional project management vs. Scrum, the Scrum teams took more ownership over the process, used the products of planning more effectively, reported more satisfaction with planning, and delivered similar or more complete final products.[11]

At Olin College, SCOPE (Senior CapstOne Program in Engineering) is a year-long engineering design capstone with 14 teams each year. Teams are multidisciplinary and comprised of students from all majors offered – Engineering (with concentrations such as Bioengineering, Computing, Design, and Robotics), Electrical and Computer Engineering, and Mechanical Engineering. Teams of 4-6 students work on projects primarily sponsored by industry. While each team has a faculty mentor and company liaison, to whom they are accountable, the teams have a high level of autonomy in how the project is executed. They often work in collaboration with the company liaison to define the scope of work and direction of the project. Though Olin students have a lot of experience working on open-ended team projects by their senior year, SCOPE is their longest project to date and requires implementation of project management tools for successful outcomes.

Historically, students received a 2 hour lecture on traditional engineering project management approaches and tools. They were asked to choose a project manager (PM) and to plan their project using a Gantt chart, using what is referred to as a "waterfall" approach. The PM role came with a great deal of power, often including decision making and task assignment. As accurately planning a 9 month project is challenging for even seasoned engineers, it comes as no surprise that student engineers were not very effective at anticipating what they could accomplish in their projects or articulating the appropriate deliverables in detail.

The issues associated with this traditional project planning approach and the rise in visibility of Agile-based project planning approaches caused the SCOPE faculty to begin to explore the use of Agile/Scrum as a project management tool that would enhance the student experience and improve project outcomes. We anticipated two types of benefits. First, we hoped that the focus on team-wide accountability would shift the teams away from an extrinsically motivated hierarchical structure to one in which team members were more engaged because they were more intrinsically motivated and held themselves accountable to the team. Second, we believed that several aspects of Scrum would improve team performance. Overall, the Scrum framework is focused on constant improvement of process. Further, the 2 week sprints are an appropriate length of time for students to plan. Also appealing was the focus on defining "done" upfront. The hope was that defining done would require team members to agree on what the task was more explicitly and to articulate when they would know it was done instead of working until the time is up or spinning their wheels.

Initially, an external Scrum trainer was brought in to give a 2 hour lecture to students at the beginning of the semester to introduce them to Scrum and they were asked to try it out. However, the trainings were very high-level, largely aimed at selling the philosophy to students as opposed to offering them concrete tools. Most teams continued to use a traditional PM approach.

Over the past 2 years, a number of Olin faculty with Scrum Master training have developed and delivered in-house trainings. These trainings focused more on introducing specific Scrum techniques for students to use. We have observed that implementation of aspects of Scrum by SCOPE teams has increased in the past few years. However, while many of the underlying philosophies of Agile/Scrum resonate strongly with our program, we are finding that the full Scrum formalism, as originally described, is not completely appropriate for our senior capstone program.

Perhaps the most obvious difference between a senior capstone environment and a corporate environment is that our student teams do not function within a company structure, nor do they often have direct interaction with the eventual user of their product. The capstone projects are sponsored by a company, and the student team works more in the role of a team of contractors with input from a liaison appointed by the company. This requires rethinking of the team leadership structure recommended by Scrum. On one hand, it is tempting to define the sponsoring company as the team's 'customer' and have an internal product owner within the team, but this doesn't exactly fit the nature of the product owner within Scrum and obscures the fact that the team's product does have an actual customer other than the liaison. Likewise defining the company liaison as the product owner does not fit within our understanding of the role of this individual within our contractual agreement with sponsoring companies. We therefore redefined the role of product owner, splitting the Scrum-defined responsibilities between the company liaison

and a team member who chooses to take on this role. Scrum Master has retained nearly its original definition.

Another difference is that, while employees of a company are usually working full time for that company, our SCOPE students are only spending about a quarter of their time on SCOPE (~12h/week), since it is only one of approximately four classes they take at a time during their senior year. This requires rethinking of the timeline of the Scrum rituals. A daily standup does not fit well into a structure where students are not making daily progress on their project. For the SCOPE implementation of Scrum, we recommended to the students that they institute some form of standup, but left it up to the student the method and timing of this ritual. Concepts like the Retrospective align with previous coaching on teamwork reflection that students have received earlier in the curriculum.

While some SCOPE teams are pure software teams, some teams are pure hardware development, and many are a mix of interacting hardware and software work. It is relatively easy to implement Scrum for the software development teams, as that reflects the environment it was optimized for and many computing students have experience with Scrum from internships. However, while Scrum devotees claim that it can be used for any process, it is more challenging to understand how to use Scrum for hardware projects or for those projects which include a more integrated development process. Similarly, Scrum was created to specifically address the development phase of a project. Our capstone projects range from research and blue-sky design phase to more defined implementation. To address this within SCOPE, we recommended that students focus less on the Scrum idea of having a fully 'shippable product' at the end of each sprint, and more on creating a list of tasks for their sprint which each had a well-defined definition of done regardless of the type of work currently being undertaken by the group.

An issue that is not unique to our students is the challenge of impressing upon students the importance of documentation. Our students often just want to build things and don't understand the details that can be lost in a hand-off to new personnel; they tend to leave insufficient documentation to allow for future work on a project to continue in an efficient manner. The tension between documentation and "doing" is worsened by using Scrum because it was developed for a fast-moving software context and for a project embedded within a company with a long institutional memory compared with an academic environment in which the students cycle every year. We therefore need to supply additional reinforcement of the need to document technical and design decisions sufficiently. We repeatedly emphasize that effective documentation is part of the process and the work is not "done" until they have captured key points relevant to passing the projects to key stakeholders (e.g. the sponsors, future teams).

Student Experience and Lessons Learned

Students designated as Product Owners and Scrum Masters on each team were asked to fill out a survey about their use of Scrum at the end of the first semester this year. Seventeen students responded, representing 10 of 14 teams. Teams were asked about the type of work

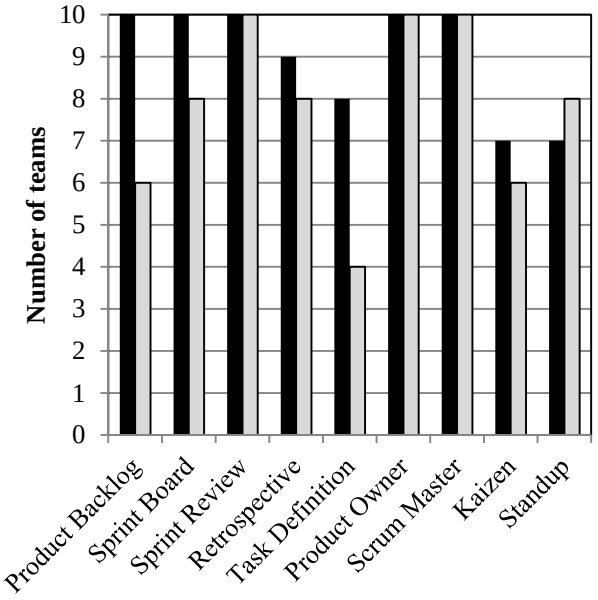

Figure 1. Teams using Scrum artifacts at the beginning (black bars) and end (grey bars) of the first semester.

they engaged in for the first semester; 9 teams engaged in research, 8 in user-oriented design, 5 in technical design and specification, 1 team built something, 2 teams started coding, and 1 team engaged in mathematical modeling.

They were then asked about which of the Scrum artifacts they started using and which they were still using. All teams started using the Product Backlog, Sprint Board, Sprint Reviews, and Product Owner and Scrum Master roles (Figure 1, black bars). Most teams used the other artifacts. By the end of the semester, all or most teams were still using the Sprint Board, Sprint Reviews, Retrospective, team Roles, the Kaizen, and Standups (Figure 1, hashed bars). This level of engagement with Scrum and retention of artifacts was quite high relative to our (unquantified) observations from previous years. The decrease in the use of the Sprint Board and task definition was somewhat surprising as this is a tool teams have found to be particularly useful in previous years. Also surprising was the high use (and even increase) in the Standup. While some teams reported not meeting daily or using an electronic, rather than in-person, method, students indicated that this was a useful tool for keeping

communication amongst the team going smoothly.

The students were also asked whether Scrum artifacts were really helpful, somewhat helpful, or not helpful in making progress on the project. Figure 2 shows the number of teams reporting that each artifact was Somewhat Helpful or Really Helpful in "making forward progress". The more positive response was used in cases where multiple team members responded.

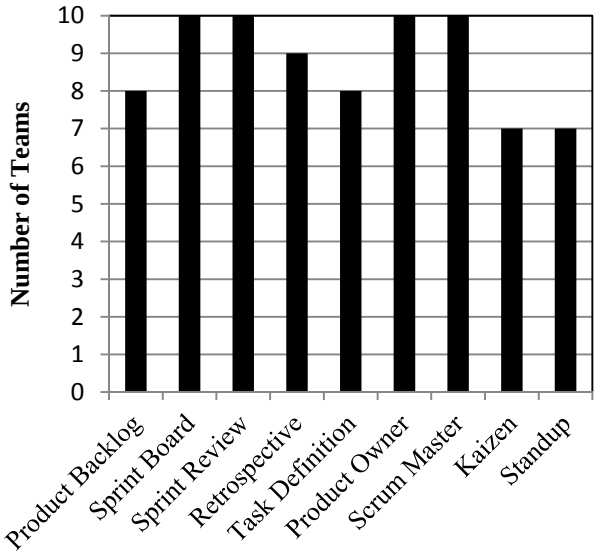

Figure 2. Number of teams reporting that Scrum artifacts helped with the progress of their project in the first semester.

It is interesting to note that students reported that the Product Backlog was useful, even if they didn't continue to use it. The Product Backlog was initiated in the beginning of the semester in conjunction with the company liaison and formed part of an extended process of learning about the project direction and goals. Also, some students who didn't report using certain artifacts or even Scrum at all in the quantitative responses often referenced using modified versions of them in their free comments. In working directly with teams, many were using the 2-week sprint unit of detailed project planning. While a concern with this approach is losing sight of the end-goal, teams were actually quite good at keeping it in mind and using the focused sprints to align their trajectory with the end goal.

While we have not yet extracted all of the lessons learned from this implementation of Scrum, the most important message appears to be that the aspects of Scrum in which the students were given both a well-defined rationale as well as some flexibility in implementation (team roles and standups in particular) appear to be the most successful in terms of both usage by the teams and perceived effectiveness in helping the team processes. From a course administration perspective, the task visibility aspect of Scrum appears to have helped increase equitable division of work between team members (as evidenced by peer- and self-assessments) and the tension over documentation within Scrum has led to some valuable conversations with teams regarding the role of appropriate documentation.

As the year progresses, we will continue to work to articulate key differences between the corporate setting and our senior capstone environment which affect the implementation of Scrum. We are also exploring other project management approaches in this space that might be more applicable to the types of projects we work on. We intend to make appropriate modifications to optimize Scrum for a senior capstone project setting or adopt (and modify) an alternate methodology.

References

1. J Sutherland and K Schwaber, "The Scrum Guide," http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf (accessed March 2016)
2. J.G. Kuhl, "Incorporation of Agile Development Methodology into a Capstone Software Engineering Project Course," *Proceedings of the 2014 ASEE North Midwest Section Conference*, Iowa City, IA, 2014.
3. K. Beck, *et al.*, "Manifesto for Agile Software Development," http://www.agilemanifesto.org (accessed March 2016)
4. M.E. Grimheden, "Can Agile Methods Enhance Mechatronics Education? Experiences from Basing a Capstone Course on Scrum," *Proceedings of ASEE*, San Antonio, TX, 2012.
5. J.T. Allenstein, et al., "From the Industry to the Student; Project Managmenet of an Industry-Sponsored Multidisciplinary Capstone Project," *Proceedings of ASEE*, San Antonio, TX, 2012.
6. K.F. Li, "Capstone Team Design Projects in Engineering Curriculum," *IEEE International Conference on Teaching, Assessment, and Learning for Engineering*, Hong Kong, 2012.
7. A. Goold, "Providing Process for Projects in Capstone Courses," *Proceedings of the 8th annual conference on Innovation and technology in computer science education*, New York, 2003.
8. D. Bowie, *et al.*, "Teaching Engineering Project Management via Capstone Designs that Develop a Viable Product," Proceedings of ASEE, Indianapolis, IN, 2014.
9. D. Rover, *et al.*, "Advantages of Agile Methodologies for Software and Product Development in a Capstone Design Project," Proceedings of FIE, Madrid, 2014.
10. V. Mahnic, "The capstone course as a means for teaching agile software development through project-based learning," *World Transactions on Engineering and Technology Education*, 13(3): 225-230, 2015.
11. M.E. Grimheden, "Increasing student responsibility in design projects with agile methods," Proceedings of ASEE, Atlanta, GA, 2013.