Fostering Agile Methodologies in Cross-Disciplinary Capstone Design Course through Process Management Tools

Kurt Pedrosa, Richard Tubbesing, Richard S. Stansbury, and Jianhua Liu
Department of Electrical, Computer, Software, and Systems Engineering
Embry-Riddle Aeronautical University
Daytona Beach, FL 32114

Abstract

Agile methodologies have gained increased popularity within industry in recent years versus more traditional methodologies such as the Waterfall or V-models. For a cross-disciplinary capstone design course with students with varied background and novice experience working with any methodology, it is a challenge to guide the students adhere to the selected methodology and avoid ad hoc development. This paper discusses the adoption of project management and version control tools by Atlassian designed to support the Scrum agile methodology. This paper presents a background of the course and its past use of agile methodologies, a literature survey on the application of Scrum and agile in capstone design courses, a discussion of how Atlassian tools and methodologies were integrated into the 2015-2016 academic year, and finally an assessment of the tools and methodology based upon student and faculty feedback.

Keywords

Capstone, Engineering, Process, Agile, Software Tools

1. Introduction

At Embry-Riddle Aeronautical University (ERAU), the capstone design course for the Electrical, Computer, Software, and Systems Engineering (ECSSE) Department supports cross-disciplinary teams for electrical engineering, computer engineering, software engineering, and computer science. Exposure to engineering process at the undergraduate level varies between programs with computer and software engineering students introduced to the Waterfall Model[1], Team Software Process (TSP)[2], and agile methodologies[3], and electrical engineering students exposed to the V-model[4].

The course currently utilizes the Scrum agile methodology[5], a framework focusing on the product development through preliminary planning, development of a backlog of prioritized product features, and then implementation by developing the features in the backlog in two-week incremental sprints. Each sprint begins with the refinement of requested product features (or user stories) into sub-tasks that are executed. The sprint addresses design, implementation, integration, and testing to produce a potentially shippable product by its end.

For the 2015-2016 academic year, to facilitate the use of Scrum based on feedback from the ECSSE industry advisory board and anecdotal experience from faculty/staff teaching the course over the past nine years using agile methodologies, an industry-grade tool set from Atlassian was selected to provide Scrum process management, version control, issue tracking, documentation, continuous testing and deployment, and code review.

This paper discusses the transition to Scrum using Atlassian tools in the capstone design course. First, the paper provides a brief overview of the capstone design course and its use of agile methodologies since 2006. Next, a brief literature survey of agile methodologies in academic student team projects is presented. The rationale for and implementation of the ECSSE capstone design's integration of Atlassian tools with a Scrum methodology is presented. Finally, an assessment of the new approach is performed using surveys and anecdotal lessons learned.

2. Background

This section provides a background of the capstone design course and agile methodologies.

2.1 ECSSE Capstone Design Course

The ECSSE department has a history of offering a cross-disciplinary capstone design course. Prior to 2011, the computer engineering and software engineering programs had a joint capstone design course. In 2011, the department started a common capstone design course for computer engineering, computer science, software engineering, and electrical engineering programs. The course has two instructors with one focused on the software and computing aspects while the other on the electrical, electronics, and embedded hardware aspects of the course. The capstone design course spans two semesters.

The first semester is focused on development of a proof-of-concept prototype. The students are first presented with a number of possible projects. The students self-select their teams with faculty coaching. Each team addresses one of these projects. The students receive lectures on process, version management, written and verbal communication, project management, and teamwork. Students work with their customers to develop requirements and preliminary design. They then begin development following an agile methodology (i.e. Scrum). The term culminating event is a poster session and product demonstration to the customers, the department faculty, and other interested parties.

The second term is focused on producing a robust functional prototype for the customers, i.e. faculty/staff members both in and out of the department. The instructors spend the majority of their time guiding team management and providing just-in-time training to address problems as they emerge. Each project ends with a full product demonstration, formal design presentation, and complete requirements and design documentation.

The 2015-2016 academic year has 23 students divided into four teams. The MEERS (Mobile Extreme Environment Research Station) team is developing a communication system and sensor network for monitoring a habitat that mimics a Mars mission habitat. The EcoCar Center Stack team is developing a vision-based situational awareness tool and infotainment system for the EcoCAR3[7] competition. The DME (distance measurement equipment) [8] team is developing an airspace surveillance tool using DME. The Scheduler team is developing an academic scheduler to assist program coordinators with term schedules and program flow charts.

2.2 Agile Methodologies

Since 2006, the capstone design course has been using agile methodologies[3] to guide the engineering process. An agile methodology is a type of engineering methodology based upon the

Agile Manifesto[9]. Under the Agile Manifesto, teams are guided to focus more on satisfying the customer's needs through direct and frequent interactions with the customer and iterative development versus more traditional engineering methodologies like waterfall, which required a substantial upfront effort to define a system's requirement specification, followed by a comprehensive design period, and finally development and testing. A variety of agile methodologies exist, including Extreme Programming (XP)[3], Crystal Clear, Scrum, etc.

Until the 2014-2015 academic year, the teams applied the Crystal Clear methodology proposed by Allistair Cockburn[10]. Much of Crystal is similar to other agile methodologies. Unfortunately, it did not scale well for larger teams, a limit acknowledged by Cockburn. A variety of variations on Crystal were attempted, and team sizes were reduced. However, the methodology was too open-ended, which resulted in students devolving to ad hoc development when under stress.

Beginning in the 2014, the course started to use Scrum[11]. In Scrum, there are three roles for team members: a product owner, a ScrumMaster, and a development team. The product owner is a representative for the product customer and stakeholders who works with the team to develop requirements, provides feedback, and answers questions. At the beginning of the fall semester, faculty and staff members are invited to give presentations of their proposed projects to students in the capstone design class. If a proposed project was selected, the proposing faculty or staff member would act as the product owner and technical subject matter expert. Each team elects a teammate to be ScrumMaster, who facilitates the process and acts as both a team leader and a developer.

Scrum embraces the "just enough philosophy" in terms of documentation and design. Prior to beginning work, the product owner and development team develop a product backlog of user stories that must be completed. Initially, the user stories are high-level statements of how the user will use the product. These stories are prioritized. Those with higher priority are refined or groomed until the work can be completed in a single sprint. When stories reach this level of refinement, they can be used as a product requirement. Not all stories are groomed since the methodology allows the team to abandon or rewrite stories based on customer feedback.

Scrum breaks up work into sprints. A sprint is a timebox in which the team must deliver some revision to the currently working product. At the start of a sprint, the product owner works with the team to groom the product backlog. Next, the team plans the sprint by delegating the user stories and their subtasks to team members given time availability and required for the task. Once a plan is generated, the sprint begins. Daily development involves a task being taken from the sprint backlog, executed, integrated, tested, and then merged into the working product. At the end of the sprint, the product owner and other stakeholders assess the product based on acceptance criteria to determine if the work is complete. A team retrospective is performed to assess their working process and make revisions to current team practices. This process continues until a release candidate is achieved.

3. Literature Review

This section provides a survey of literature related to the application of agile methodologies and industry-grade engineering process management tools.

The agile methodologies were proposed to address issues with the waterfall method in general. Coupal and Boechler [12] specifically explain the pitfalls of using the waterfall method in software engineering courses and discuss the benefits of using agile methodologies. As explained by Dewi and Muniandy [13], many of the agile philosophies map well to common classroom objectives. For example, changes in the project are welcomed, even late in the project's life. These changes are partially due to the fact that students are not experienced in project planning and execution.

Many authors have discussed how to adapt agile methodologies, especially Scrum, to better suit the classroom, especially in software development related courses. For example, Pinto et al. [14] discuss how Scrum was adapted to be applied to a real-time systems class. Notably, they have the course instructor as the Product Owner and use document-sharing tools, such as Google Docs and internet messaging tools, to facilitate team meetings and communication, as it is often difficult for teams to find a common meeting time. Other examples include a software development capstone course of Mahnic[15], a computer science capstone course of Knudson and Radermacher[16], software engineering student teams of Gamble and Hale[17], and inter-disciplinary service-learning project courses of Nejmeh and Weaver[18].

Some reviewed papers provide detailed assessment and survey analysis on the use of Scrum in the academic settings. For example, Mahnic[15] presents empirical evaluations of students' progress in estimation and planning skills. It is indicated that students' ability to accurately estimate the time needed on a project increases as they gain experience. The percentage of story points completed changed from 42%, to 75.18%, and to 91.8% over the course of three sprints. This shows that the Scrum process can indeed allow students to learn better project management in the classroom. Gamble and Hale[17] provide a detailed account of their assessment that includes meeting check-ins, discussion posts, software repositories, and other sprint deliverables.

Tools usage is reported in some of these classes. For example, Gamble and Hale[17] use a tool called SEREBRO, which is similar to the Atlassian tools we use. Also, Nejmeh and Weaver[18] use a Scrum process management tool called PivotalTracker. This tool, much like the Atlassian tools, allows the students to capture epics and stories, estimate story points, assign owners to each story, etc. Tools such as the Atlassian tools, SEREBRO, and PivotalTracker help immensely in keeping the sprint organized and capturing artifacts needed for assessing students.

4. Integration of Scrum and Atlassian Tools into Capstone Design Course

This section presents the effort at ERAU to integrate Scrum implemented using the Atlassian toolset in the ECSSE capstone design course. First, a rationale for the transition is provided to justify the transition and why the tool set was selected. Next, an overview of the technical environment for the tools is presented. Third, the process adopted for the course based on Scrum and the tool set is discussed, including how it influences course deliverables and grading.

4.1 Rationale of using Atlassian tools

Scrum is a project development framework and it can be used in a project development without using a tool set. For instance, a common approach is to use a corkboard designed as a Scrum board with a product backlog area and a sprint backlog with swimlanes, i.e. team member

assignments, for the sprint. Yet, the authors saw the following problems in the past with the capstone design courses that used agile/Scrum frameworks without a comprehensive tool set.

1. Without a good tool, grooming of the product backlog is awkward and is often ignored, which could result in some important features not being addressed by the end of the course.
2. Without a well-groomed product backlog, the team efficacy is severely diminished, as the team will be unable to adequately plan the sprint backlog or track progress.
3. Students have many other different commitments, and without a good tool for online collaboration, it is difficult for them to fully use their small time slots to do project work.
4. A tracking system allows both team members and faculty to monitor each team member's progress, allowing intervention if a student becomes stalled on a task.
5. As an alternative to personal engineering notebooks, an online wiki with change management allows the team to save design documents in a means that can be accessed anywhere and by all authorized users (i.e. teammates).
6. Many projects, especially hardware-related, do not use a version control tool, which can lead to lost work. The tool set encourages use of version control tools.

The Atlassian tool set was selected as it provides Scrum process management, version control, issue tracking, documentation, continuous testing and deployment, and code review. We use mainly the following four tools.

1. JIRA[19], a backlog management tool. It can be used to do the following.
    a. Creating and grooming the product backlog.
    b. Planning the sprint.
    c. Handling the daily Scrum, including logging time spent on each task.
    d. Creating the reports for sprint review and retrospective.
2. Confluence[20], a team collaboration and knowledge management tool. It can work in conjunction with JIRA and other Atlassian tools and can be used to do the following.
    e. Creating documentations, including meeting notes, product requirements, knowledge base articles, wiki of the product.
    f. Collaborating between the team members without having the team members working the same location and the same time.
    g. Capturing all the information, including the lessons and skills learned in each sprint, to one website so that it is easily searchable.
    h. Facilitating the generation of sprint retrospective.
3. Bitbucket[21], a web-based version control system based on Git[22]. With this tool, version control is an integrated part of the Scrum framework.
4. Bamboo[23], a continuous integration and delivery tool that ties automated builds, tests, and releases together in a single workflow for some projects of the course.

4.2 Technical Environment

All of the above Atlassian tools are being hosted by a Microsoft Azure cloud server[24]. This server runs CentOS Linux[25] operating system. Apache[26-28] handles all server requests made via the host website (http://db-atlassian.cloudapp.net) and proxies all the requests made to other server ports that are connected to the Atlassian tools. This website allows users to access the

Figure 1: Website allowing users to access different Atlassian tools from one location.

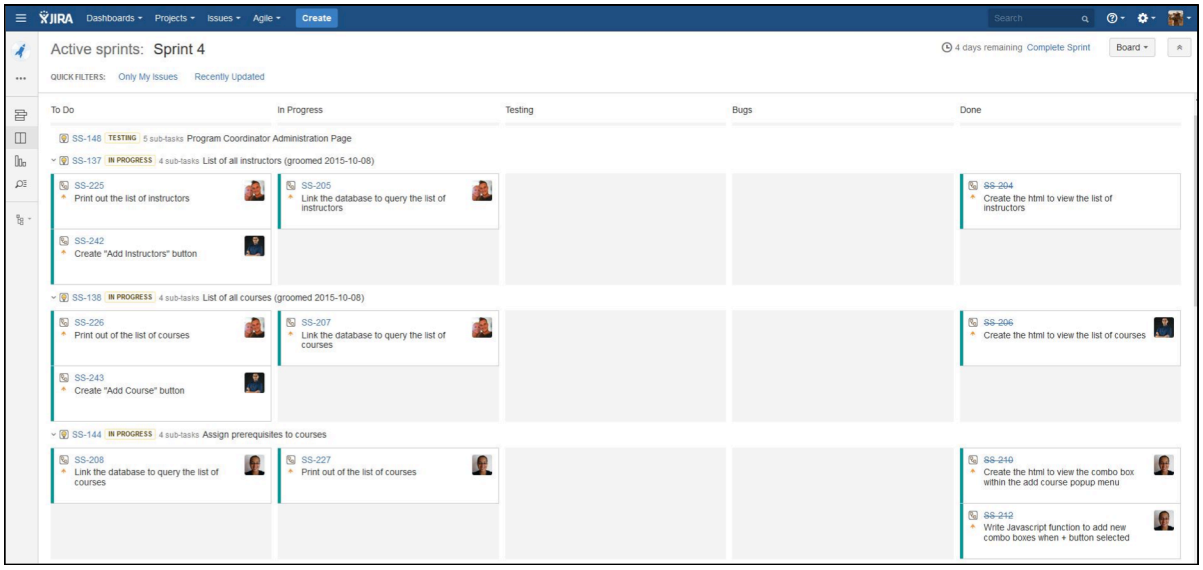

Figure 2: Example of JIRA agile being used to track tasks of a current sprint.

Atlassian tools as well as change their login information, and contact the server administrator. The layout of this website can be seen in **Error! Reference source not found.**. The implementation of this website has been a useful addition which helps centralize all access made to the server.

Once in JIRA, users can access their project boards and manage their tasks using interactive Scrum boards. Members of each team can drag, create, delete, and manage all of the project tasks in one place giving the team an up-to-date status of their progress. As seen in Figure 2, team members have a board-like view of all of their tasks as well as the tasks status. As tasks get completed they are moved from one status column to the other until they are marked as done.

One important feature of JIRA is the burndown chart feature. This chart, as shown in Figure 3, plots the productivity of the team during each Scrum sprint. It shows the task hours that have not

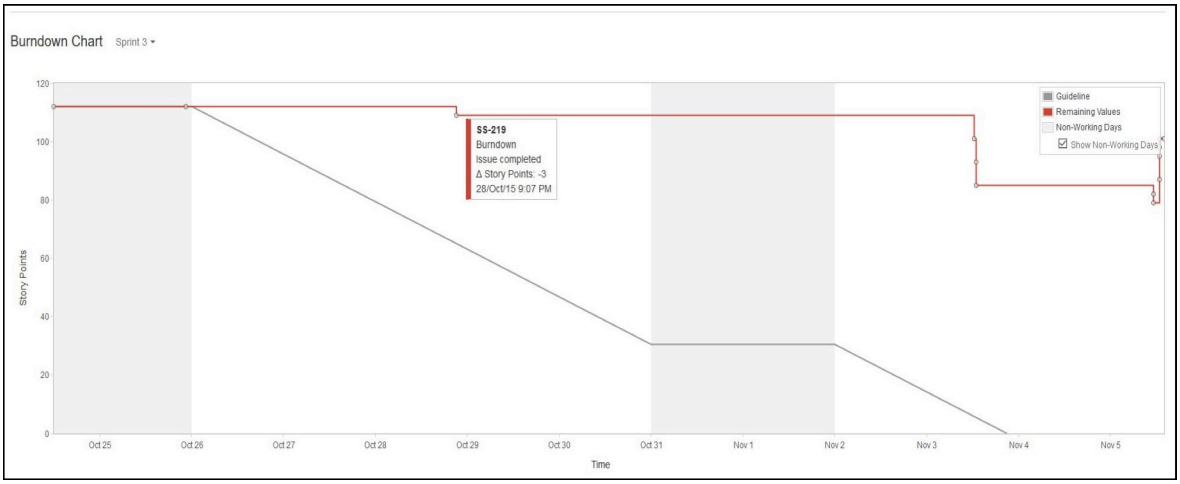

Figure 3: Burndown chart showing task completion progress.

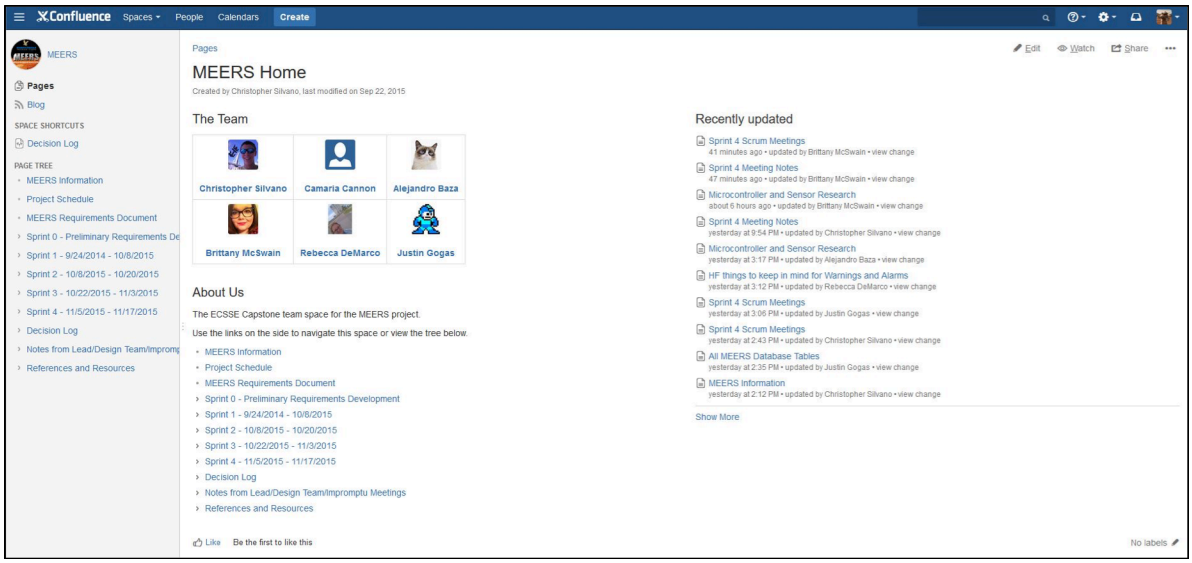

Figure 4: A team Space in Confluence showing the team members, Space activity, and file list.

been completed yet and graphs them against the time period in which the team decided those tasks needed to be done.

In Confluence, team members create project Spaces which is an area dedicated to that specific team where they can share articles, create and collaborate on documents, and host their files. Spaces are a great way to keep the team organized by creating catalogs of documents. Another benefit of Confluence is that any document in the Space can be found by keywords or document text. As teams build their Spaces, the Confluence page slowly grows into a wiki page about the project with links to articles, documents, JIRA tasks, Bamboo test reports, and anything else relevant to the team. As seen in Figure 4, a team space is fully customizable and fully integrated with all of the other Atlassian tools. Confluence is the most used tools and acts as an information hub for the teams.

The other two Atlassian tools, Bitbucket and Bamboo, serve as supplementary tools to JIRA and Conflunece. Bitbucket is used as the primary Git version control system, which allows the team members to track file changes while being able to revert back to any previews version of that file. Each change is tracked and labeled providing the team members a clear understanding of the changes being made and a visual representation of these changes. The reversion feature is useful when a change commit introduces a bug that cannot be easily debugged.

Each commit done to a team's repository correlates to a task in JIRA. Usually these tasks are additions to the product or fixes requested by the product owner. In order to ensure that each commit does not compromise the current build of the product, tests are performed at the time of commit. If the current commit passes these tests, it is approved and introduced into the current build of the product. This entire process is done by Bamboo. Bamboo works concurrently with Bitbucket and performs team-designed tests at the time of each commit. Bamboo also gives the team the option to deploy different builds of their product.

4.3 Process Implementation and Assessment

The fall 2015 semester capstone design course is divided into two phases, the team forming and preparation phase and the sprint phase consisting of four two-week sprints.

During the team-forming and preparation phase, the faculty and staff members of ERAU first present the students with a number of possible projects. After the presentations, students are instructed to form teams to address one of these projects. Team formation involves self-selection, some coaching from course instructors, as well as adjustments according to the requirements of the projects. For conventional Scrum teams, the product owners are responsible for generating the first batch of product features, but for our course, students were tasked to do this to provide a requirements elicitation experience preceding design to meet the course's learning outcomes. The student teams are instructed to make detailed plans to interview the product owners. The interview questions include questions of the vision and scope of the projects, needed skill sets, time lines and forms of deliverables, etc. Students record this information in JIRA and Confluence to form the initial product backlog.

During the sprint phase, students perform the standard Scrum sprints. A sprint starts with sprint planning and ends with a sprint retrospective.

1. *Grooming.* The student team and the product owner meet regularly to groom the product backlog, which includes creating new features of the project, breaking down the big features into a set of smaller features that are doable in one sprint, and prioritizing the features according to their importance to the product owner.
2. *Sprint planning.* Each team, including all of the team's members and the faculty/staff product owner, meet together to determine a subset of the product backlog items they believe they can and should complete during the sprint. The development team breaks down each big feature into a set of detailed and implementable tasks, and put them into the sprint backlog. Each task should be accompanied by an estimate of time to be used and a plan to design, build, integrate, and test the task's work during the sprint. For our capstone design teams, to address the personal responsibility and fairness issue, each task

Table 1. Distribution of points of individual assessment.

Deliverable	**Points Assigned**
Attendance (mandatory)	10
Scrum Sprint Performance	10
Faculty Performance Evaluation	10-15
Total	**30-35**

Table 2. Distribution of points of team assessment.

Deli**verable**	**Points Assigned**
Final Requirements Document	10
Design Documentation and Manual	15
Prototype	20
Prototype Design Review (presentation)	10
Scrum tool and process usage (version control, continuous integration, sprint planning, reflection, etc.)	15
Total	**70**

is assigned to team members during the planning phase. This is supported well by the Atlassian tools.

3. *Daily Scrum.* For a conventional Scrum team, there is a daily Scrum, every day at the beginning of the day for about 15 minutes or less, for face-to-face team communication. This is also called a daily stand-up. Yet, for our capstone design team, the teams use Confluence to perform such a "daily" Scrum.
4. *Sprint review.* A sprint review is a meeting to inspect and adapt the product, which has two parts. The first one is the meeting between the team members and the product owner, which focus on identifying the "done" features of the sprint according to previously agreed-upon acceptance criteria for the feature. The second is a class demo for the entire capstone design class and the interested faculty/staff members.
5. *Sprint retrospective.* A sprint retrospective is a meeting between the team members to inspect and adapt the process, which usually occurs after the sprint review and before the next sprint planning. Its focus is on the continuous process improvement necessary to help a good Scrum team become better. At the end of a retrospective, the team should have identified and committed to a practical number of process improvement actions.

For the course, students are assessed individually, as shown in Table 1, and collaboratively, as shown in Table 2. Individual student performance on doing project work is measured using the Atlassian tool's tracking capabilities. Team performance is assessed both in terms of the product as well as the adherence to the Scrum methodology and Atlassian tool usage.

5. Results and Lessons Learned

This section presents the team's analysis of the adoption of the Scrum methodology applied with the Atlassian tool set for the course. First, student attitudes and comments captured from

Table 3. Student survey questions and results.

Survey Question	Student Response Count (N=27)					Avg. Score	Std. Dev.
	Strongly Disagree (1)	Disagree (2)	Neutral (3)	Agree (4)	Strongly Agree (5)		
I understand the Scrum methodology and its application to my capstone design project.	0	0	2	11	14	4.444	0.641
I understand how to use JIRA.	0	1	4	13	9	4.111	0.801
I understand how to use Confluence.	0	0	1	15	11	4.370	0.565
JIRA / JIRA Agile assisted my team in backlog planning, sprint planning, and progress monitoring.	0	1	5	10	11	4.148	0.864
I feel that learning the Atlassian tools improves my understanding of the Scrum methodology.	0	3	6	13	5	3.741	0.903
JIRA / JIRA Agile helped our team engage with our product owner.	2	7	6	7	5	3.222	1.251
Confluence assisted my team with maintaining requirements and design documentation.	0	0	7	8	12	4.185	0.834
Using the Atlassian tools is a valuable experience.	0	4	4	7	12	4.000	1.109

anonymous end-of-term course surveys for the fall term are presented. Then, the results of a survey of faculty/staff product owners are presented and discussed. Finally, additional lessons learned from the instructors and teaching assistants are discussed.

5.1 Student Course Evaluations

Student feedback and attitudes toward Scrum and the use of Atlassian tools is an important indicator of the viability of this approach and its acceptance among students. The survey uses statements that are then rated based upon a Likert scale from Strongly Disagree (1) to Strongly Agree (5). The questions and results are shown in Table 3.

The survey results demonstrated an overall positive reception by students of the methodologies and tools incorporated into the classroom. The lowest score received involved the ability to use the Atlassian tools to engage with the product owner / customer. This issue will need to be remediated by providing further training to the faculty product owners. Students also seemed to like the Atlassian tools, but a number did not feel that they provided an additional benefit to the Scrum methodology.

The following are a sampling of noteworthy free-form comments from the survey (most are direct quotes with a few paraphrased due to length). Things they liked best about the Atlassian tools:

- General:

    - Keeps everything organized and easy to see who is working on what.
    - I liked the inclusion and flow of having all of these tools in one place.
    - Makes interaction with teammates easier.
    - Gives capstone a more professional feel than notecards and a corkboard since these types of products are used in industry.
    - I have already had interviewers comment on my knowledge of these tools.
    - Very cool about being able to link JIRA stories/tasks to Confluence pages and Bitbucket branches.
- JIRA Specific
    - The scrum board is kinda cool. There are lots of features it is lacking that we had noticed in the beginning but we found ways around it.
    - It helped to track our progress and plan for future tasks.
- Confluence Specific
    - Useful for keeping all documents organized in a central location and made it easily viewable to the instructors.
    - All documents/updates/tasks were there for the whole team to see.
- Bitbucket Specific
    - Bitbucket is fantastic for code management.

Things they liked least about the Atlassian tools:

- Usability
    - Usability is not straightforward.
    - Learning curve especially for the first couple of sprints.
    - Some actions were complex and non-intuitive for moving content in confluence.
    - Overcomplicated. Could easily have been done with a physical Scrum board.
    - Confusing until you understand how Scrum works.
- Availability
    - Downtime (frequently cited issue).
    - Tools had a fair amount of problems with staying online and maintaining passwords (had to be reset often).
- Inability to track effort for sub-tasks
    - JIRA did not track story points (*time estimates*) at the sub-task level making planning difficult.
    - JIRA only updated burndown chart for progress tracking if entire story was completed and not if any sub-tasks were completed for a story.
- Not sure if the process is being correctly followed in JIRA. Real-world examples on exactly how some industry folks use JIRA to document their stories would be very helpful to point out what we are doing wrong.

5.2 Faculty Product Owner Survey

Faculty members acting as product owners were surveyed to determine their attitudes toward Scrum and the Atlassian tools. The survey attempted to determine how well they felt integrated into the development process and whether or not the tools facilitated that relationship. The

Table 4. Faculty product owner survey questions and results.

Survey Question	Faculty Response Count (N=4)					Avg. Score	Std. Dev.
	Strongly Disagree (1)	Disagree (2)	Neutral (3)	Agree (4)	Strongly Agree (5)		
I understood my role as product owner within the Scrum process.	0	0	0	2	2	4.50	0.58
The development team coordinated backlog groom and planning with me.	0	1	1	1	1	3.50	1.29
I utilized the Atlassian tools to view, groom, and prioritize the product backlog.	1	0	2	0	1	2.33	1.15
The Atlassian tools simplified the process of viewing, grooming, and prioritizing the product backlog.	0	0	2	1	0	3.33	0.58
I would recommend students follow the Scrum methodology for other design courses.	0	0	0	2	2	4.50	0.58
I would recommend students use the Atlassian tools for other design courses.	0	0	1	0	2	4.33	1.15

survey uses statements that are then rated based upon a Likert scale from Strongly Disagree (1) to Strongly Agree (5). The questions and results are shown in Table 4.

The survey results demonstrated that the faculty product owners have a good understanding of the role of product owner within the Scrum process and tend to coordinate with the team to groom the backlog and plan the sprints. Yet, many of them did not use the tools and as a result did not see the value of the tools in maintaining and grooming the product backlog. Despite the relatively low score for the usage of the tools, they deem the Scrum methodology valuable for other design courses, and they would also recommend the Atlanssian tools. Note that the faculty survey is not as indicative as the student survey due to a couple of reasons: (a) the number of sample for faculty product owner is very small; (b) faculty members did not have the outside pressure or motivation to use the tools; (c) they have a highly diverse level of tool usage from not at all to frequent; and (d) the faculty and staff product owners came from a variety of backgrounds including electrical engineering, mechanical engineering, computer science, and human factors.

Please note that one of the product owners surveyed is one of the capstone course's instructors, who additionally took on the role of product owner. This course was their first opportunity to use the Atlassian tools, but they had an existing background in using Agile methodologies for student teams. Despite best efforts to remain objective when reviewing the Atlassian tools and methodologies discussed, a personal bias may exist.

5.3 Other Observations

A number of ad hoc observations from the authors and colleagues are captured in this sub-section.

Technical environment challenges: When choosing to utilize a tool such as the Atlassian tools, which is both self-hosted and Internet-based, there are some significant challenges in setting up the tools' infrastructure. Both a graduate research assistant and an IT professional are required to configure the cloud server, install and configure the software, and provide technical support. Data backups must be run routinely to avoid the risk of lost data critical to student work.

During our fall term, the server went down for a two-and-a-half day period due to insufficient storage allocated to the leased cloud server, which required the migration to a larger storage solution. The downtime had a significant impact on the students' ability to function with the Scrum process as of their work, their backlogs, and design documentation stored in confluence was unavailable. Without the assistance of both the teaching assistant and IT support professional, the impact could have been significantly worse. A second outage occurred close to the end-of-term. While the issue was quickly resolved, each outage resulted in a negative sentiment toward the toolset, and did create a vocal minority that spoke negatively of the adoption of the toolset.

Nuance in Agile Methodologies: A common issue experienced by the course instructors is students becoming frustrated or confused when encountering an issue (e.g. user story or customer request) that did not align with the prescriptive discussions of Scrum found in books and online. With Agile, there is a need to work within the framework, but to have flexibility to allow it to work across a multitude of situations. Students do not have the maturity to know when to treat the methodology as a rule versus when to treat it as a guideline / best practice. A more experienced engineer typically has a better understanding of this nuance from past experiences in working with different design standards and best practices in industry.

Product Owner Participation: Despite providing guidance to the product owners on how to use Atlassian tools at the start of term, a number of product owners remained hands-off with using the Atlassian tools, and in particular JIRA. This was reflected in both ad hoc students' input as well as the faculty survey shown above. It is recommended that a more hands-on approach should be taken for future projects to provide additional mentorship and training to the product owners on Scrum and the Atlassian tools for the first two sprints to ensure better understanding and tool buy-in.

6. Conclusions

The adoption of industry-grade tools to support cross-disciplinary design teams to implement their projects following the Scrum agile methodology has proven to be valuable to the author's capstone design course. It, however, has not been without some technological challenges. The free academic license for the Atlassian tools required the establishment of a complex technical environment, which on the whole is cheaper than a commercial license, but still has associated costs and risks including the need for individuals with IT experience to maintain the servers and address outages quickly when they occur.

For the teams that used the Atlassian tools and had a product owner that also adopted the use of the tools, the adherence to the Scrum agile methodology was significantly higher. Future academic years will need to focus on improved training for faculty and staff acting as product owners and additional coaching during the first few Scrum sprints. IT issues should be resolved

after this first year's experience; however, having IT support available and on-call to resolve technical issues is important as prolonged downtime has a significant negative impact on student attitude toward the tools.

References

1 Winston W. Royce. Managing the development of large software systems: Concepts and techniques. In WESCON Technical Papers, 1970. Reprinted in Proceedings of the Ninth International Conference on Software Engineering, 1987, pp. 328–338.

2 Watts Humphrey. Team Software Process. Software Engineering Institute, Technical Report CMU/SEI-2000-TR-023, 2000, online at: http://www.sei.cmu.edu/reports/00tr023.pdf.

3 Allistair Cockburn. Agile Methodologies: The Cooperative Game, Addison-Wesley, 2nd edition, 2006.

4 Nabil Mohammed Ali Munassar and A. Govardhan. "A Comparison Between Five Models of Software Engineering." IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010.

5 Jeff Sutherland. Scrum: The Art of Doing Twice the Work in Half the Time. Penguin Books: New York, 2014.

6 Atlassian. "Software Development and Collaboration Tools". Online at: www.atlassian.com, accessed 11 November 2015.

7 US Department of Energy. "Advanced Vehicle Technology Competition: EcoCar3". Online at: ecocar3.org, accessed 11 November 2015.

8 Federal Aviation Administration. "JO 7470.1A - Distance Measuring Equipment (DME)/DME Infrastructure Evaluation for Area Navigation (RNAV)." Online at: https://www.faa.gov/regulations_policies/orders_notices/index.cfm/go/document.information/documentID/15572, accessed 12 November 2015.

9 Beck et al., "Manifesto for Agile Software Development," Online at: www.agilemanifesto.org, accessed 11 November 2015.

10 Allistair Cockburn. Crystal Clear: A Human-Powered Methodology for Small Teams. Addison Wesley, 2004.

11 Kenneth Ruben. Essential Scrum: A Practical Guide to the Most Popular Agile Process. Addison Wesley, 2012.12 Cyril Coupal and Kelvin Boechler, "Introducing agile into a software development Capstone project," in 2005 Proceedings of Agile Conference, pp.289-297, 24-29 July 2005.

13 Deshinta A. Dewi and Mohana Muniandy, "The agility of agile methodology for teaching and learning activities," in 2014 8th Malaysian Software Engineering Conference (MySEC), pp.255-259, 23-24 Sept. 2014.

14 Luciano Pinto; Ricardo Rosa; Cristiane Pacheco; Christophe Xavier; Raimundo Barreto; Vicente Lucena; Marcus Caxias; Carlos M. Figueiredo, "On the use of Scrum for the management of practical projects in graduate courses," in 39th IEEE Frontiers in Education Conference, 2009. FIE '09, pp.1-6, 18-21 Oct. 2009.

15 Viljan Mahnic, "A Capstone Course on Agile Software Development Using Scrum," in IEEE Transactions on Education, vol.55, no.1, pp.99-106, Feb. 2012.

16 Dean Knudson and Alex 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), pp.444-448, 22-24 May 2011.

17 R. F. Gamble and M. L. Hale, "Assessing individual performance in Agile undergraduate software engineering teams," in 2013 IEEE Frontiers in Education Conference, pp.1678-1684, 23-26 Oct. 2013.

18 Brian Nejmeh and D. Scott Weaver, "Leveraging Scrum principles in collaborative, inter-disciplinary service-learning project courses," in 2014 IEEE Frontiers in Education Conference (FIE), pp.1-6, 22-25 Oct. 2014.

19 Atlassian. "JIRA Software-Issue and Project Tracking for Software Teams." Online at: https://www.atlassian.com/software/jira/, accessed 12 November 2015.

20 Atlassian. "Confluence – Team Collaboration Software." Online at: https://www.atlassian.com/software/confluence/, accessed 12 November 2015.

21 Atlassian. "Bitbucket – the Git solution for professional teams." Online at: https://www.atlassian.com/software/bitbucket, accessed 12 November 2015.

22 Git. "Git --fast-version-control." Online at: https://git-scm.com/, accessed 12 November 2015.

23 Atlassian. "Continuous Integration & Build Management – Bamboo." Online at: https://www.atlassian.com/software/bamboo/, accessed 12 November 2015.
24 Microsoft. "Microsoft Azure: Cloud Computing Platform & Services." Online: https://azure.microsoft.com/, accessed 12 November 2015.
25 CentOS. "CentOS Project." Online at: https://www.centos.org/, accessed 12 November 2015.
26 Apache. "The Apache HTTP Server Project." Online at: https://httpd.apache.org/, accessed 12 November 2015.
27 Apache. "Apache Tomcat." Online at: http://tomcat.apache.org/, accessed 12 November 2015.
28 Mulesoft. "Meet Apache Tomcat." Online at: https://www.mulesoft.com/tcat/tomcat-catalina, accessed 12 November 2015.

Biographic Sketches:

Kurt Pedrosa

Kurt Pedrosa is a Master's student at Embry-Riddle Aeronautical University in Daytona Beach, FL. He is currently working on a Rockwell Collins funded project to develop new alternative methods to sense and avoid aircraft. Most of his work is focused on developing new receiver architectures on software-defined radios in order to detect high frequency signals. His focus is on FPGA image development and implementing algorithm for detecting aircraft. He is a Graduate Teaching Assistant for the Capstone senior design class. In his off time you can usually find him in the RF laboratory trying to listen to the ISS communication channels using software defined radio.

Richard Tubbesing

Richard G. Tubbesing IV was a Master's student at Embry-Riddle Aeronautical University in Daytona Beach, FL. He received his Master's Degree in Electrical and Computer Engineering in December of 2015 and began to work at Rockwell Collins after receiving his degree. His graduate project involves using FPGAs to decode digital TV signals in real time for use in a passive radar application. Additionally, he was one of the two Graduate Teaching Assistants for the capstone senior design class in the ECSSE department in the fall 2015 semester. For fun, he is an automobile enthusiast and likes to work on his '91 Toyota MR2 in his spare time.

Richard S. Stansbury

Dr. Richard S. Stansbury is an associate professor of computer engineering and computer science at Embry-Riddle Aeronautical University in Daytona Beach, FL. He is actively involved in unmanned aircraft system (UAS) research and is the university PI for the ASSURE FAA Center of Excellence for Unmanned Aircraft Systems. He has supported FAA's Office of Commercial Space Transportation in the development of an ADS-B prototype for reusable suborbital launch vehicles, and is the ERAU representative for the FAA Center of Excellence for Commercial Space Operations. He teaches capstone senior design for his department and is the program coordinator for the Master of Science in Unmanned and Autonomous Systems Engineering.

Jianhua Liu

Dr. Jianhua Liu is an associate professor of electrical engineering at Embry-Riddle Aeronautical University in Daytona Beach, FL. He has expertise in digital signal processing, signal processing

for wireless communications, image processing, applications of image processing in in unmanned aircraft system (UAS) applications. He teaches capstone design, digital signal processing, and image processing courses and is currently the program coordinator for the Master of Science in Electrical and Computer Engineering.