Journal of Information Systems Education

Volume 30
Issue 3
Summer 2019

Teaching Tip
Implementing Scrum Wholesale in the Classroom

Corey Baham

Recommended Citation: Baham, C. (2019). Teaching Tip: Implementing Scrum Wholesale in the Classroom. *Journal of Information Systems Education,* 30(3), 141-159.

Article Link: http://jise.org/Volume30/n3/JISEv30n3p141.html

Initial Submission:	30 September 2018
Accepted:	22 January 2019
Abstract Posted Online:	5 June 2019
Published:	12 September 2019

ISSN: 2574-3872 (Online) 1055-3096 (Print)

Teaching Tip
Implementing Scrum Wholesale in the Classroom

Corey Baham
Department of Management Science and Information Systems
Oklahoma State University
Stillwater, OK 74074, USA
corey.baham@okstate.edu

ABSTRACT

As the most widely used agile software development method, Scrum has become a mainstay in many organizations that develop software. Despite Scrum's popularity, several studies examine Scrum implementations that include some parts of the methodology and exclude others. This paper describes how Scrum has been incorporated into the classroom wholesale and highlights important considerations when using Scrum for student software development projects. Students having little to no knowledge of Scrum were able to gain confidence in using the method in a real-world setting. The paper discusses the use of a hands-on Scrum project as a pedagogical tool for teaching the Scrum methodology and software development life cycle principles. Quantitative and qualitative data were collected to understand student experiences with a wholesale Scrum implementation in the classroom. The paper concludes with data analysis and recommendations for implementing Scrum in future projects.

Keywords: Agile, Scrum, Collaboration, IS education

1. INTRODUCTION

The widespread use of the Internet and the emergence of object-oriented programming have led to unprecedented changes in the software development industry. Seeking competitive advantages in a more globally connected economy, firms sought increases in software production speed, efficiency, and agility. In response, software development practitioners grappled with ways to develop faster, more agile processes to produce more frequent iterations of working software. During the 1990s, a number of agile software development (ASD) methods were created, and this shift in the software development practice was further solidified in 2001 with the advent of the *Manifesto for Agile Software Development* (Beck et al., 2001). As ASD methods have become a mainstay in the business world, ASD should be taught not only to computer science students but also to business students, such as management information systems (MIS) majors, in order to inform them of the current software development landscape in organizations. Despite the popularity of ASD methods in industry and the increased attention from Information Systems recruiters and executives, current Systems Analysis and Design (SAD) textbooks provide limited knowledge on how to implement ASD methods. Hands-on software development projects, which are appropriately scoped, can help motivate MIS students to explore both the social and technical concepts pertaining to ASD. On this basis, the author assigned MIS students a course project to develop a web application using Scrum, the most widely used agile method (West et al., 2010; Version One, 2018). This project is simple enough for MIS students to implement, as it requires moderate programming skills already learned in other courses. The project lets the students explore, in some depth, the combination of social and technical aspects of software building involved in ASD. In accordance with guidelines for teaching tips, this paper contributes to the literature as wisdom-of-practice scholarship (Weimer, 2006) by detailing how Scrum has been implemented in the classroom, providing empirical results of multiple wholesale Scrum implementations, and providing pedagogical recommendations for future implementations. The term "wholesale" refers to an implementation of Scrum that utilizes all (as opposed to parts) of the Scrum process components and roles, which we discuss in the next section. The solutions described herein are replicable, grounded in theory and best practices, and recommended based upon actual experiences. The rest of this paper is organized as follows.

Section 2 provides an overview of Scrum. Section 3 describes a doable project for business students. Section 4 discusses how to initiate and implement Scrum into the classroom. This is followed by a discussion of the pedagogical approach and a summary of student feedback in Section 5. The paper concludes in Section 6 by reviewing important aspects of Scrum projects.

2. AN OVERVIEW OF SCRUM

Scrum, which gets its name from the game of rugby, was formalized into a method for building software using a holistic, team-based approach (Takeuchi and Nonaka, 1986; Schwaber, 1995). Scrum focuses on defining process components and roles (Holvitie, Leppänen, and Hyrynsalmi, 2014) as shown in Table 1, but leaves the practicalities open for choice (Abrahamsson et al., 2002).

Process Components	**Definition**
Daily Meetings	A meeting when the Scrum team shows what they accomplished during the Sprint.
Iteration Backlog	A list of the Product Backlog items the team commits to delivering plus the list of tasks necessary to delivering those Product Backlog items.
Product Backlog	A prioritized list of desired product functionality.
Sprints	The intervals into which the development process is divided.
Sprint Planning Meetings	A meeting where the Product Owner describes the highest priority features.
Sprint Reviews	A meeting when the Scrum team shows what they accomplished during the Sprint. Typically, this takes the form of a demo of the new features.
Sprint Retrospectives	A brief, dedicated period of time set aside at the end of each Sprint to deliberately reflect on how the team is doing and to find ways to improve.
Roles	**Description**
The Development Team	Professionals who do the work of delivering a potentially releasable increment of "done" product at the end of each Sprint.
Product Owner	A person who is responsible for maximizing the value of the product and the work of the Development Team.
Scrum Master	A person who is responsible for ensuring Scrum is understood and enacted.

Table 1. Scrum Processes and Roles (Cohn, 2010; Sutherland and Schwaber, 2016)

The Scrum framework created by Sutherland and Schwaber (2016) describes the interaction between the Scrum team and its customer. It consists of the roles, ceremonies and artifacts (i.e., process components), and guidelines that serve a specific purpose as shown in Table 1. Additional details concerning what Scrum entails can be found in the references provided (Schwaber, 1995; Abrahamsson et al., 2002; Rubin, 2012; Sutherland and Schwaber, 2016). As a recent study highlights (May, York, and Lending, 2016), Systems Analysis and Design textbooks contain little content on ASD methods, often containing only enough information to introduce students to ASD concepts and games such as planning poker. Although useful, most of the education literature on Scrum lacks the software development context (Pope-Ruark, 2012), a holistic treatment of the Scrum methodology (i.e., focuses on a few ceremonies within Scrum) (Yue et al., 2009; May, York, and Lending, 2016), or important details concerning how Scrum can be implemented in the classroom (Cleland and Mann 2003; Jiménez and Cliburn 2016).

Early ASD papers in the education literature emphasize the importance of incorporating agile approaches into Systems Analysis and Design courses (Batra and Satzinger, 2006) and employ adoption assessments to determine the viability of specific agile methods for a given project (McAvoy and Sammon, 2005). More recent work describes agile implementations and student feedback in various courses (Lang, 2016; Weber, 2016), including several capstone projects (Mahnic, 2012; Hoskey and Hoskey, 2016). This literature emphasizes the importance of understanding the differences between ASD projects in the classroom setting compared to industry, including student versus professional expertise, academic calendars and student schedules versus a 40-hour work week, and feedback from professionals versus instructors and teaching assistants. Wagh (2012) highlights the need for a lightweight, adaptive approach to teach ASD that can be tailored to the limited time and resources available in an academic setting. The results suggest that by focusing on small achievable work in Sprints, students were able to deliver more complete and cohesive features as opposed to more splintered development work produced using traditional methods. Baird and Riggins (2012) combined traditional planning with three-week, Scrum-based Sprints. Their results suggest that students preferred a hybrid approach to an agile only approach for constructing software prototypes. Additionally, the study points out the student satisfaction scores could have been higher with more customer involvement. Masood, Hoda, and Blincoe (2018) describe the effectiveness of specific adaptations to agile practices in a university context. Among their recommendations are conducting ceremonies face-to-face when possible, supporting teams with experienced tutors and upfront training, and using online tools to simplify team communication.

This study builds on the extant literature in the following ways. First, the Scrum projects in this study are fully functional applications rather than prototypes. Student projects were made up of seven, approximately one-week Sprints. Each week students are expected to deliver a minimum viable product which fulfills the basic user requirements. The software was expected to be deployed after the last Sprint was completed. Second, the projects followed Scrum very closely including all its roles and ceremonies. Modifications were limited to those that adapted Scrum to the classroom without compromising its core tenets such as performing Scrum Meetings on class days instead of on a daily basis. In comparison to previous work (Baird and Riggins, 2012), we did not employ a Scrum/waterfall hybrid approach, exclude ceremonies like Sprint Reviews or Sprint Retrospectives, or use blueprint-style planning (Faludi, 1973). Although some empirical studies note that many organizations heavily modify ASD methods in practice (Fitzgerald, Hartnett, and Conboy, 2006; Maruping, Venkatesh, and Agarwal, 2009; Ramasubbu, Bharadwaj, and Tayi, 2015), prior research and industry case studies note that some organizations adhere to ASD methods wholesale (Overhage and Schlauderer, 2012; Case Studies,

2017; Scrum Case Studies, 2017). Thus, we implemented Scrum wholesale to provide students with an example of how all the roles and ceremonies work. Third, the student projects described herein mandate the use of a Product Owner, which is a fundamental role in Scrum, yet excluded from some implementations of Scrum in the classroom. In most cases, our Product Owners were local business owners who agreed to have students build a web application for their businesses in exchange for their full participation in the Product Owner role. In fewer cases, a pseudo Product Owner was used in the form of the instructor or a person with experiences related to a given project. In comparison to previous work (Mahnic, 2012; Wagh, 2012; Jiménez and Cliburn, 2016), in no cases were Product Owners fellow students who were enrolled in the same course.

In this paper, we draw upon the personal experience of teaching Scrum in Systems Analysis and Design courses to provide guidelines on initiating and implementing Scrum for teachers. The guidelines herein come from overseeing over 50 hands-on Scrum projects. It should be said that implementing Scrum wholesale is not always a "nice and neat" process. Instead, the process can be somewhat messy, requiring flexibility in adapting, while not removing the core tenets of Scrum during the execution of each unique software project. Despite these challenges, in the next two sections, we present a way to scope, initiate, and implement Scrum projects based on classroom experiences.

3. THE ASSIGNMENT: A DOABLE SCRUM PROJECT

3.1 Course Overview

Prior to taking SAD, students were required to complete the following prerequisite courses:

- Introduction to Object-Oriented Programming
- Database Management
- Web Development

In rare circumstances, the Web Development course could be taken concurrently with SAD. Students are expected to apply the concepts learned in the prerequisite courses in SAD where they are asked to complete a working piece of software that integrates with a database.

The 17-week SAD course is laid out as follows: During Weeks 1-5, using the Valacich and George (2017) textbook, an overview of the fundamental systems development life cycle (SDLC) phases are provided, namely: planning, analysis, design, implementation, and maintenance. The introductory chapter discusses the traditional waterfall approach and briefly describes ASD methods. Students are tested on basic SDLC concepts. Next, project teams are assembled and given some training in Scrum over Weeks 6-7. Once topics are approved, they begin applying planning concepts toward their project as shown in Appendix A. A total of seven Sprints are completed from Weeks 8-16 (nine weeks) with two Sprints spanning two weeks due to semester breaks. The final week of the course is reserved for student presentations.

A hands-on software development project was chosen over the further teaching of SDLC terminology and concepts from the textbook. The rationale is that a hands-on project would help make the concepts more concrete and provide students with a project to add to their personal portfolios. In addition, other than the SDLC phases, the course textbook contains several topics that are covered in other courses such as Database Management. An ASD approach was chosen over waterfall because (1) there has been a wide adoption of ASD methods in practice, (2) the value of learning an ASD approach has been lauded by many of the department's industry partners, and (3) students are exposed to more waterfall style approaches in other classes (e.g., project management, web programming, etc.). While teaching SAD, we wanted to have students gain experience applying an ASD method to create a unique software solution to solve a real-world problem. Thus, a hands-on project allows students to experience both the process components and roles involved in Scrum.

3.2 Project Requirements

Students were given the choice between a portfolio of class projects pre-selected by the instructor and proposing their own unique project. All projects were required to meet the minimum requirements of:

- Creating an application (web or mobile) that integrates with a database.
- Securing the involvement of a Product Owner who agreed to meet weekly (or at least bi-weekly) to provide feedback for the software team.
- Employing modern coding and design principles as expressed by the instructor.

The portfolio of projects available required only limited support and moderate technical knowledge, which students should have from previous courses (e.g., web programming, database, etc.). Students were encouraged to build web applications using Microsoft Visual Studio, which they used in an introductory programming course, as they were familiar with many of the features in Microsoft's technology stack. Therefore, setting up the student project required simply maintaining the standard Web accounts that are available to all MIS students. This project did not require students to learn many new skills, but instead it challenged students to integrate the technical skills they learned in separate courses.

3.3 Example of a Student Project

Students developed web applications that could receive information through web forms and store it in a database. In many cases, the information stored in the database could be recalled in a way that is beneficial for customers. For example, one group developed a web application that tracked the location and displayed pertinent information about local food trucks, such as hours of operation and menu items. Food truck owners could create an account and enter information about their food truck. Additionally, food truck owners could make their location visible to application users during their hours of operation. Once food truck information is entered, it is recorded in the database and application users can search for food trucks based on attributes such as food type and area. Other projects included storefront web applications for clothing shops, local diners, and audio/video equipment vendors. These web applications place a strong emphasis

front-end design concepts such as the use of mobile friendly, responsive frameworks, easy to navigate layouts, and an appropriate use of colors and spacing.

3.4 Team Assignments

Project teams were constructed strategically to replicate organizations where managers arrange individuals into teams as opposed to allowing students to self-select their team members (Masood, Hoda, and Blincoe, 2018). Students were asked to complete a questionnaire detailing their technical experience and level of expertise across the following areas (1 to 5 rating where 1 = little to no knowledge and 5 = expert knowledge):

- Database
- IT Infrastructure
- Front-end design and coding
- Back-end coding

Students were then mixed and matched in small groups of 3-4 based on their skill levels. All these skills were covered in previous courses apart from a few instances where students were given permission to take some courses concurrently with SAD. Moreover, we aimed to create an average score of 12 or higher in groups of 4 with at least one member with a score of 4 or higher in back-end coding. Appendix B shows how students were mixed and matched for the team assignments.

3.5 Notable Constraints

Although the project is doable, the semester timeline, student skill level, and Product Owner availability are constraints that must be accounted for. The project should be able to be completed within one semester. Thus, the project scope should be managed so that the core functionality is completed before adding extra "bells and whistles." Additionally, the project should be moderately challenging for the average student. In the next section, we present a two-phase approach to executing Scrum in the classroom within the aforementioned constraints.

4. INITIATING AND EXECUTING SCRUM

Drawing on the extant literature, we note that training facilitates method knowledge and ongoing coaching helps to deepen knowledge long-term (Senapathi and Srinivasan, 2014). Additionally, more customer involvement in ASD projects aids in knowledge sharing and understanding user requirements. Thus, the Product Owner plays a pivotal role in Scrum projects. Working with a Product Owner provides business students the opportunity to build software in a way that incorporates continuous feedback from their clients. This literature helped us to develop a theory base for adapting Scrum to the classroom setting using a two-phase solution. Figure 1 shows the initiate and execute phases used to introduce and guide students using Scrum, most for the first time. The initiate phase to Scrum is meant to help students get started with Scrum and better understand their user requirements before building software.

Initiate	• Scrum Training
	• Project Planning
	  ○ Conduct a User Story Workshop
	  ○ Develop Product Backlog

Execute	• Method Adaptations
	  ○ Scrum Roles
	  ○ Scrum Process Components
	  ○ Class Schedule
	  ○ Documentation

Figure 1. Phases of Scrum Projects

4.1 Scrum Training

ASD surveys consistently point to the importance of adequate training for ASD teams (Version One, 2018). In line with this recommendation, instructors should take time to introduce the Scrum framework, its roles and ceremonies, and Scrum's relation to the *Manifesto for Agile Software Development*. Given that most SAD textbooks discuss the fundamental phases of the SDLC, providing a historical backdrop of ASD methods may help students understand why these methods were developed. Additionally, providing examples from practice could help students to contextualize an organization's desire to adopt such methods as opposed to simply discussing the nuts and bolts of the Scrum framework. Similar to industry Scrum training, the foundational Scrum concepts can be taught over a few class sessions. We recommend dedicating a few hours to Scrum training upfront which can include many of the basic concepts covered in books and practitioner literature. In our experience, two one-and-a-half-hour class periods were used for training. A slide deck was provided to students which described all the Scrum roles and ceremonies previously discussed. These concepts should be reviewed during the project as well to ensure method discipline. Short (5-10 minute) training sessions at the end of Day 1 (see Appendix A) of each week were used to strengthen Scrum execution, address bad habits (e.g., not standing during Scrum Meetings) and frequently asked questions, and introduce new concepts (e.g., estimating user stories, burn down charts). Topics covered in these short training sessions included:

- Improving Scrum Meeting effectiveness
- Improving your effectiveness as a Scrum Master
- Scrum in the classroom vs. Scrum in industry
- Relative estimation
- Information Radiators – Part 1: Burn down charts
- Information Radiators – Part 2: Task boards

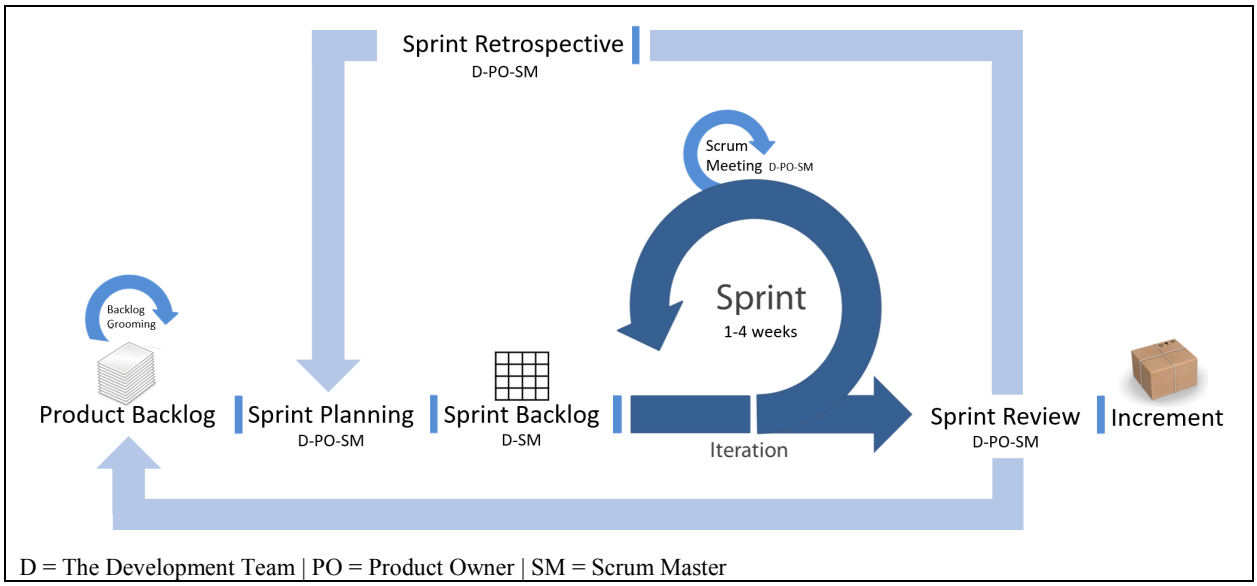


Figure 2. The Scrum Workflow

4.2 Project Planning

Conduct a user story workshop to develop the Product Backlog and kick off the project. Once the project teams have been assigned and the minimum project requirements have been explained, teams can begin project planning. Students should be encouraged to use planning techniques from prior courses such as developing a project scope statement. We used the concept of Sprint 0 to kick off the project. Sprint 0 is a time-boxed iteration of project planning. Once the project is outlined at a high level, a user story workshop (Cohn, 2010) can be conducted with each team and their Product Owner. Students complete the workshop in three steps. First, each team member generates as many user stories (or short, simple descriptions of a feature told from the perspective of the person who desires the new capability) as possible apart from the influence of the other team members. Second, the initial set of user stories is organized collectively according to the user roles (or similar likeness). Here similar stories can be consolidated into one. Third, the team attempts to prioritize the stories according to their customer's needs. The Product Owner has the final say on the priority of the user stories. The finalized list is deemed the original Product Backlog. With the Product Backlog in hand, the team is now ready to do Sprint Planning and start their first Sprint. At this stage of the project, common relative estimating techniques (e.g., story points, ideal days, etc.) were not enforced.

Instead, students focused on scoping each story so that it was small enough for one person to complete alone within a "reasonable amount of time" (i.e., a one-week Sprint). Teams were able to add the number of user stories to the Product Backlog that they agreed on without the pressure of estimating user story sizes "exactly right." Even though relative estimating techniques were introduced in a training session during Sprint 4, teams were often able to discover their capacity through trial and error in the first few Sprints. For instance, teams were asked to provide a percentage of the Sprint Goal completed and an explanation of that percentage at the end of each Sprint. During the Sprint Retrospective, teams reflected upon the number of actual user stories completed versus the total estimated. Teams discussed these differences to improve future estimates. Overall, the approach described above was effective for cognitively simplifying estimating, which is often difficult for students to grasp especially when trying to learn all the core aspects of a new method.

The execute phase describes the application of Scrum in building software incrementally. Here, the team should gain practical knowledge of how the roles and responsibilities of each person fit within the Scrum workflow as shown in Figure 2. Increments of a potentially shippable product were determined each week during the Sprint Planning meeting in consultation with the Product Owner. At the end of each Sprint, teams were expected to deliver a product increment in accordance with its stated Sprint Goal, which was described by user stories in the Sprint Backlog. This product increment was expected to be an extension of the previous increment. User stories that were not completed during a Sprint were often given priority in the next Sprint. In these cases, during Sprint Planning, teams considered new user stories after accounting or the portion of the unfinished user stories from the previous Sprint. Overall, teams completed user stories in the order specified by the Product Owner.

4.3 Method Adaptations

Adaptations to traditional Scrum roles and process components were necessary to fit the classroom setting (Masood, Hoda, and Blincoe, 2018). Beginning with the Scrum roles, we describe the adaptations employed in our implementation of Scrum.

4.3.1 Scrum roles. Since the Scrum Master role was new to most students, the role rotated to a different team member every Sprint (Hoskey and Hoskey, 2016). This provided an opportunity for each development team member to gain experience in the role of Scrum Master and encouraged members to contribute to the technical requirements of the

project. Rotating the Scrum Master role also prevented one member from adopting a non-technical role for the duration of the project. Other business courses such as Project Management offered students the opportunity to oversee technical projects without performing hands-on technical work. Each week a given Scrum Master would lead the Scrum Meeting by asking the following questions of each team member (Cohn, 2010):

- What have you done since the last time we met?
- What will you do today?
- Are there any impediments in your way?

As a follow up, the Scrum Master would initiate actions that helped the team to remove impediments stated during the Scrum Meeting.

As previously mentioned, each team was responsible for securing the commitment of its Product Owner. For teams that built applications for existing businesses, business owners or company stakeholders were a natural fit for the Product Owner role. For other teams, identifying a Product Owner was not so obvious and required creative thinking. For instance, the team that created the Food Truck application found a person outside of the students in the course who was considered a local food expert, had eaten at numerous food trucks, and agreed to meet with the team once a week during the Scrum Meeting and Sprint Review ceremonies. This enabled the team to simulate the Product Owner role.

In other cases, the instructor served as a pseudo-Product Owner if he/she had relevant expertise with the product. A number of strategies were employed for Product Owners that did not communicate adequately or provide timely feedback. Product Owners were asked to give their verbal consent to actively participate for the entire nine-week (seven Sprint) duration of the project. Students were challenged to show initiative in following up with Product Owners both digitally and in person if necessary. Any Product Owner that did not communicate to the team within the first two Sprints was replaced. In these cases, students explored a better fit for the Product Owner role within the organization before moving to a different organization. Any of the new Product Owners that did not provide consistent feedback would have been replaced by the instructor or graduate assistant. Fortunately, no teams had to switch Product Owners more than once. In addition to the creativity needed in selecting a Product Owner, added flexibility was required to work with some Product Owners' schedules. Teams developed workarounds for Product Owners that could not meet during the designated time for Scrum Meetings and Sprint Reviews. Video conference applications were particularly useful in overcoming spatial limitations.

Regarding the Development Team, each team member took responsibility over at least one aspect of the project: database, IT Infrastructure, front-end design and coding, or back-end coding. Groups with three members had one team member occupy two roles. In the spirit of collaborative Scrum teams, team members were encouraged to assist in multiple areas as needed. However, each team member was required to ensure that their assigned area was completed. If team members completed the work in their assigned area early, the team and instructor came together to determine what portion of the project could use additional attention.

4.3.2 Class schedule. Figure 3 displays a one-week Sprint schedule that each of the teams followed during the class project. The project lasted roughly nine weeks (seven Sprints) of the semester with students attending two 1.5-hour classes per week. Figure 3 is reflective of a two-day per week (i.e., Tuesday / Thursday) course schedule, but can be altered to fit different teaching schedules such as moving Sprint Planning to the third day of a three-day work week. In our example, a Sprint begins with Sprint Planning and ends with the Sprint Retrospective. On Day 1, teams are expected to complete their Scrum Meeting and then work on their projects. Instructors are encouraged to use this time to visit with each group and examine their progress and answer project related questions. Although the time provided during class was enough to complete much of the project, most students mentioned that they met outside of class periodically. Lastly, we used the last 5-10 minutes of class for training or to address frequently asked questions.

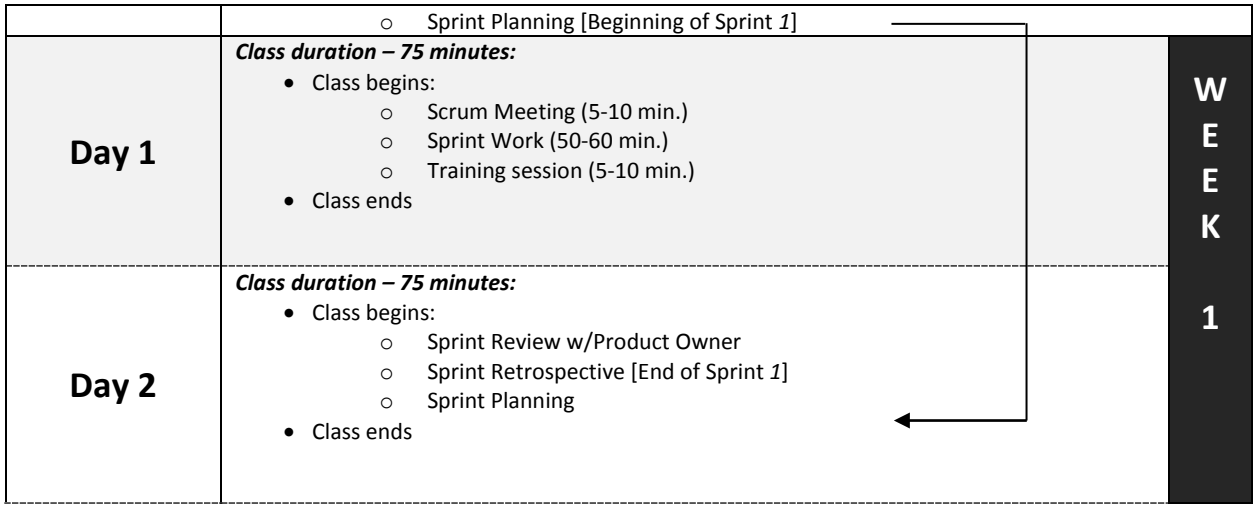


Figure 3. Sprint Schedule

On Day 2, teams are expected to complete their Sprint Review by demonstrating the work completed on their application to their Product Owner. After the Sprint Review, the team is to close out the Sprint with the Sprint Retrospective. A common framework for conducting the Sprint Retrospective is to use the Start doing – Stop doing – Continue doing framework. Here team members reflect on the execution of the last Sprint, highlighting the positives and negatives individually before discussing them together. Day 2 commences with the Sprint Planning Meeting where teams engage in Product Backlog Grooming with the Product Owner before producing the next Sprint Backlog. If the Product Owner is not available, teams may send the proposed revisions of the Product Backlog to the Product Owner for feedback. Again, workarounds may be implemented as necessary.

4.3.3 Documentation. Requiring the completion of relevant documentation is a helpful way to keep team members accountable throughout the project. In the example project, teams were required to document their activities using video or audio recordings, word processing forms, or both. Scrum Meetings and Sprint Reviews were video recorded and uploaded to a course management system. This provided accountability for monitoring team member attendance and the progress of the application. Teams were required to complete the appropriate word processing forms for the Scrum Meeting, Sprint Review, Sprint Retrospective, and Sprint Planning ceremonies. These forms solicited basic information such as the date and names of participants as well as team issues and screen shots of the application. Interested readers may contact the author for copies of these forms. The forms were compiled into a binder that organized each set of Scrum ceremonies by Sprint. Overall, these adaptations helped students to execute Scrum relatively wholesale in a classroom environment.

5. PEDAGOGY AND STUDENT FEEDBACK

In this section, we provide pedagogical details concerning the use and effectiveness of Scrum projects as a mechanism to teach Scrum concepts. The course project moves those concepts beyond abstract concepts to a deeper understanding. While earlier sections describe how the project is assigned to and executed by business students, this section presents the use of this project as a pedagogical tool for teaching Scrum concepts.

5.1 Project as Pedagogical Tool

To stimulate a deeper understanding, students should include in their project report a discussion on Scrum concepts they learned by analyzing their projects. Those concepts include the following:

- Discuss three Scrum values (transparency, inspection, and adaptation). Understand the importance of the Scrum values as they relate to building software as a team.
- Discuss differences between traditional roles of Project Manager and Business Analyst and Scrum roles of Scrum Master and Product Owner. Explain three distinct roles and responsibilities that Scrum Masters and Product Owners possess.
- Discuss at least three things that were learned during the development of the application using Scrum.
- Discuss the actual scope of work that was completed versus that which was originally captured in the original Product Backlog. Detail conditions that emerge that impacted scope changes.
- Discuss what influenced the team's selection of communication mediums for video conferencing, chatting, etc. Explain the pros and cons of these mediums.
- Discuss next steps. What might future iterations of the application look like?

5.2 Student Feedback

Both quantitative and qualitative data were collected throughout the course to assess the effectiveness of using Scrum in the classroom. All 41 students from 2 sections of the course were invited to complete the survey anonymously (see Appendix C). Thirty-five of the 38 students who began the survey completed it (92% response rate). For the quantitative questions, we asked students questions associated with each of the research measures which were answered using a five-point Likert scale ranging from 1-Strongly Disagree to 5-Strongly Agree as shown in Table 2. A mean of 4 or above suggests that, on average, students at least "Somewhat Agree" with the statement. A mean of 2 or below suggests that, on average, students "Somewhat Disagree" with the stsatement. A mean of 3 is a neutral response ("3-Neither Agree nor Disagree"). These questions measured prior Scrum knowledge, current Scrum knowledge, perceived team Scrum knowledge, and perceived team execution quality across each of the Scrum ceremonies.

Variables	Mean	Std. Error
Prior Scrum knowledge		
Q1	1.486	0.180
Current Scrum knowledge		
Q2 Scrum Meeting	4.800	0.069
Q3 Sprint Planning	4.543	0.085
Q4 Sprint Review	4.629	0.092
Q5 Sprint Retrospective	4.343	0.108
Q6 Product Backlog	4.257	0.118
Q7 Sprints	4.714	0.077
Perceived team Scrum knowledge		
Q8 Scrum Meeting	4.629	0.101
Q9 Sprint Planning	4.543	0.095
Q10 Sprint Review	4.457	0.103
Q11 Sprint Retrospective	4.286	0.113
Q12 Product Backlog	4.143	0.137
Q13 Sprints	4.629	0.092
Overall knowledge of Scrum		
Q14	4.643	0.086
Perceived team execution		
Q15 Scrum Meeting	4.429	0.118
Q16 Sprint Planning	4.514	0.111
Q17 Sprint Review	4.371	0.124
Q18 Sprint Retrospective	4.200	0.122
Q19 Product Backlog	4.029	0.151
Q20 Sprints	4.343	0.136
Future Outlook – *I feel:*		
Q21 …that doing Scrum enhanced my knowledge of Scrum	4.800	0.090
Q22 …comfortable doing Scrum at a future job	4.457	0.095
Q23 …that doing Scrum enhanced my knowledge of SDLC principles	4.286	0.127
Q24 …comfortable in a systems analyst position	4.214	0.122

Table 2. Mean Scores

5.2.1 Prior knowledge. During the first day of class, students were asked about their knowledge and experience of Scrum principles. Only one student said they were "very knowledgeable" of Scrum. Additionally, three students said that they were "somewhat knowledgeable" of Scrum. However, upon further investigation, the student that responded as "very knowledgeable" of Scrum actually used a Scrum variation that excluded practices such as Sprint Retrospectives and time-boxed Scrum Meetings. This student later indicated that doing Scrum more holistically enhanced their knowledge of Scrum. Overall, 88% percent of students indicated that they were "not very knowledgeable" or "not knowledgeable [of Scrum] at all" at the start of the class.

5.2.2 Current knowledge. Near the end of the project, students were surveyed concerning their experience with using Scrum. In summary, the majority of responses averaged 4 ("somewhat agree") or higher on all questions. Questions Q2-Q7 assessed respondent's perceived knowledge of each Scrum ceremony ("Currently, I have an adequate knowledge of ___"). Questions Q8-Q13 assessed respondent's perceptions of their team's knowledge of each Scrum ceremony ("My team has an adequate knowledge of ___"), while question Q14 assessed respondent's overall knowledge of Scrum ("Overall, I am (now) knowledgeable of Scrum principles and practices"). Questions Q15-Q20 assessed respondent's perceptions of their team's level of execution of each Scrum ceremony ("My team executed ___ as designed").

Questions related to individual and team knowledge of Scrum Meetings had some of the highest averages of 4.800 and 4.629, while questions related to Product Backlog Grooming had some of the lowest averages 4.257 and 4.143, respectively. One possible explanation for this is that teams were asked to conduct Scrum Meetings, Sprint Planning, and Sprint Reviews upon arriving to class and Scrum Meetings were done in a section of the classroom where the instructor could see and interact with the team. Despite Scrum Meetings being executed only on Day 1, students reported a high level of understanding and execution of the ceremony and met outside of class to work collaboratively on their projects. Thus, most of the transparency afforded by "daily" Scrum Meetings was attained with the modifications presented. Sprint Planning, Sprint Review, and Sprint Retrospective ceremonies required teams to complete formal documentation each time, while teams could conduct and document Product Backlog Grooming more informally. Some attrition was observed in some teams' ability to conduct the Sprint Retrospective as instructed. The execution of this ceremony, which had a mean score of 4.200, took place after the Sprint review, a major milestone. Team members were asked to reflect individually before discussing the quality of the past Sprint's effort collectively. Some teams resorted to "just getting the documentation done" instead of taking adequate time to reflect. Product Backlog Grooming ceremonies were less visible, required more initiative to complete, and occurred less frequently than the other Scrum ceremonies. This is also evidenced by the lower averages for team Product Backlog Grooming execution.

Questions Q21-Q24 asked students to indicate their comfort level with Scrum and software development life cycle (SDLC) concepts (i.e., planning, analysis, design, and implementation) moving forward. These questions also solicited qualitative data by asking students to explain their answers in addition to providing 1 to 5 ratings, as follows:

21. I feel that doing Scrum enhanced my knowledge of Scrum. Explain: ___
22. I feel comfortable doing Scrum at a future job. Explain: ___

23. I feel that doing Scrum enhanced my knowledge of SDLC principles. Explain: ___
24. I would feel comfortable in a systems analyst position. Explain: ___

The open-ended responses allowed us to gain deeper insights about students' level of comfort with Scrum and SDLC principles moving forward. For Q21-Q24, 86% percent of students "strongly agreed" that doing Scrum enhanced their knowledge of Scrum and another 9% "somewhat agreed." Only 2 of 35 students were neutral. First, student responses frequently mentioned how doing Scrum helped them understand how the methodology worked on a software development project. Below are a few examples:

> *Before Scrum and actively using it, I had no idea how it worked or was. Actively using Scrum is the best way to learn.*

> *I had no knowledge of Scrum before taking this course. By participating in a Scrum Project, I was able to apply the theory in meaningful ways.*

> *It is a lot easier to learn something by doing it rather than reading about it in a text book. I liked that this class was able to bring it to life, so that we are more prepared in the workforce.*

Second, 97% of students felt at least moderately comfortable doing Scrum at a future job. Explanations indicated that the most confident students felt ready to do Scrum at a future job immediately while others, though confident, felt they might need a refresher:

> *Having experience with [Scrum] now gives me confidence using it again in the future.*

> *It might take a little refreshing, but I would be comfortable for sure.*

> *Overall, I would be fine doing Scrum in an actual professional setting.*

These results suggest that teaching Scrum even to students with experience doing Scrum in an internship is useful as corporations often exclude certain aspects of Scrum as they tailor it to their needs. By learning Scrum wholesale, students gain a more comprehensive understanding of Scrum including the aspects that an organization may exclude.

Third, most students (89%) felt that doing Scrum enhanced their knowledge of SDLC principles at least moderately. While some students said that "doing Scrum enhanced [their] knowledge [of SDLC principles] tremendously" others saw Scrum as less of an enhancement and more of a "direct application" of their SDLC knowledge. Upon reflection, a greater effort should be made to help students understand how SDLC principles are applied in ASD methods like Scrum. These linkages could have been made more conspicuous by juxtaposing SDLC principles with the Scrum framework during the introduction to Scrum. This might have helped one of the four students who was neutral. One of them said,

> *Yes [Scrum enhanced their knowledge of SDLC principles], but it's still hard to conceptualize Scrum and SDLC together due to the nature of both being so contrasting.*

Fourth, most students (91%) indicated that they would feel at least moderately comfortable in a systems analyst position. Of these, the majority at 57% said they were "somewhat comfortable" and 34% said they were "very comfortable." Those students that were "very comfortable" expressed that being a systems analyst was their "dream job" and that the project helped enhance their knowledge of software development. Some comments were as follows:

> *After this course, I feel like I know the requirements to be able to perform well in a [systems analyst] position.*

> *Yes, I would [feel confident in a systems analyst position] due to this project enhancing my knowledge of software development.*

We also observed relatively high scores from those who either had a past or upcoming internship related to software development (e.g., business analyst, systems analyst, etc.). Many universities support the idea of students getting internships, oftentimes to gain meaningful work experience that may lead to future full-time employment. Additionally, many organizations employ internships as a key evaluator of prospective student talent. The findings in this study suggest that not only is the acquisition of an internship beneficial for a student's future industry career, but also their academic career.

Similar to the previous question, a greater effort should be made to help students to feel confident about their ability to function in the role of a systems analyst by understanding how the skills required to complete the software project translate to the systems analyst role. Although students performed a few requirements gathering exercises and ultimately developed a working piece of software, some students felt that they needed more experience doing systems analysis work before they felt "very comfortable" in the system analyst role. Below are two student responses that express this sentiment:

> *I have a good understanding of the role but would like to have more experience.*

> *After more trainings… I would feel comfortable in a systems analyst position.*

In summary, this project stimulated discussions about ASD concepts of which students may not have otherwise had a clear understanding. In addition to the text presented in this section, the main points reported by students are summarized below. They have said this project:

- Helped them gain hands-on experience using Scrum, which enhanced their knowledge of both Scrum and SDLC principles.
- Helped them feel confident about both participating in an ASD project at a job and working as a systems analyst.
- Helped them understand the advantages and disadvantages of ASD versus traditional methods.
- Helped them understand how important communication is in meeting changing customer requirements.
- Helped them understand how to approach IT projects.

5.3 Study Limitations and Future Research

As with all studies, this one has limitations. First, this study is limited to a single professor and a single institution. Although these factors limit generalizability, the findings in this study confirm findings in previous studies on the effectiveness of using Scrum in the classroom to teach software development principles. Future work could look at wholesale Scrum implementations across multiple student populations and elaborate on their differences. Second, the nine-week (seven Sprints) Sprint schedule presented here assumes that the scope of the projects selected are of doable size and in line with student expertise from prior courses. In this study, software projects were typically company websites of less than 10 pages which contained basic information, pictures, and a web form that connects to a database. Since these projects tended to be small in scope, future research could examine Scrum implementations in larger projects that span multiple semesters and student teams. Third, this research focused on equipping educators with implementing Scrum wholesale, which we specify as using all the core process components and roles as previously defined. Future research could explore more specific and complementary aspects of Scrum such as estimation techniques, task boards, burndown charts, and software tools in greater depth. Similarly, specific attention could be given to Scrum roles. For example, the Scrum Master role was rotated among team members each Sprint, so students did not get consistent experience in the role. As a result, they seemed to be more keenly aware of their technical challenges (e.g,. connecting the database, setting up the code repository, formatting the front-end using CSS, coding web forms, etc.) rather than challenges related to the Scrum Master role. We also found that a detailed analysis of the tips, feedback, and lessons learned from working with Product Owners to merit a separate paper.

Given that implementing Scrum wholesale for student projects was shown to be effective, we reiterate that future projects should take care to make the linkages between Scrum and SDLC principles more salient. Additionally, once learned, Scrum ceremonies should be observed so that students complete them rigorously and do not resort to "going through the motions." Similarly, Product Owners should be chosen carefully so that students can benefit from working with a person who is invested in the project and willing to provide timely feedback. Moving forward, we recommend using a formal agreement for Product Owners so that they understand the expectations and responsibilities of the role. Additionally, providing training to Product Owners either through video or face-to-face instruction could potentially help. This will likely minimize cases where Product Owners fail to provide adequate feedback.

Perhaps the most difficult part of the projects was successfully implementing them by the business. The adoption rate (<5%) was poor for several reasons. First, a formal process for transitioning class projects to live web applications was lacking. Second, students lack motivation to continue with the projects after their course requirements ended. At the semester's end, student schedules change and they rarely have available time to meet each week. Third, many Product Owners were hesitant to deploy their web applications because of their lack of knowledge concerning the web hosting process, unwillingness to cover the maintenance costs, and in a dearth of cases, lack of satisfaction with the design. Future research should formalize a process that transitions class projects to live web applications before the course ends. Both Product Owner and student or instructor commitments are needed to complete this task. Other considerations for future research include examining implementations for ASD methodologies other than Scrum (e.g., Kanban) and applications for ASD beyond developing web applications (Baham et al., 2017).

6. CONCLUSION

This study contributes to the literature by detailing how Scrum has been implemented in the classroom wholesale, providing empirical results of multiple Scrum implementations, and providing pedagogical recommendations for future implementations of Scrum in the classroom. In relation to our results, we summarize our recommendations as follows:

- Require Students to Deliver Actual Software
- Require Students to Acquire a Product Owner
- Provide Scrum Training
- Carefully Adapt the Method
- Use Documentation to Monitor Method Discipline

As Cao et al. (2002) note, the pedagogical method of choice tends to be driven by such factors as the educational background of students and the objectives of their academic program. Not only is understanding how to develop a simple software program expected as a software developer, but understanding the importance of working in a dynamic team environment is critical to adding value to a group project. In working on this project, students were asked to integrate previous knowledge and add new knowledge as they encountered new challenges. Additionally, many business students may eventually lead software projects. Thus, the social and technical skills promoted here can help them to have better communication with IT professionals. Students searched the Internet for solutions to complex problems, which were not often found in a textbook. In some cases, being unable to find "one-size-fits-all" solutions led to complaining. Therefore, instructors should make sure that students limit the scope of their projects to that which matches their expected skillset in a given academic program. Overall, the findings of this study suggest a number of benefits for implementing Scrum wholesale in the classroom.

7. REFERENCES

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). *Agile Software Development Methods: Review and Analysis*. VTT Publication 478, Espoo, Finland.

Baham, C., Hirschheim, R., Calderon, A. A., & Kisekka, V. (2017). An Agile Methodology for the Disaster Recovery of Information Systems under Catastrophic Scenarios. *Journal of Management Information Systems*, 34(3), 633-663.

Baird, A. & Riggins, F. J. (2012). Planning and Sprinting: Use of a Hybrid Project Management Methodology within a CIS Capstone Course. *Journal of Information Systems Education*, 23(3), 243-258.

Batra, D. & Satzinger, J. W. (2006). Contemporary Approaches and Techniques for the Systems Analyst. *Journal of Information Systems Education*, 17(3), 257-266.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). Manifesto for Agile Software Development. Retrieved March 9, 2018, from https://agilemanifesto.org/.

Case Studies. (2017). Retrieved March 9, 2018, from http://www.scaledagileframework.com/case-studies/.

Cao, Q., Davis, J. S., Bai, X., & Katter, O. E. (2002). Using ASP-Based Message Encryption Project to Teach Information Security Concepts. *Journal of Information Systems Education*, 13(3), 183-188.

Cleland, S. & Mann, S. (2003). Agility in the Classroom: Using Agile Development Methods to Foster Team Work and Adaptability amongst Undergraduate Programmers. *16th Annual NACCQ*.

Cohn, M. (2010). *Succeeding with Agile: Software Development using Scrum*. Pearson Education.

Faludi, A. (1973). *Planning Theory, Urban and Regional Planning Series (1st edition)*. Oxford: Pergamon Press.

Fitzgerald, B., Hartnett, G., & Conboy, K. (2006). Customising Agile Methods to Software Practices at Intel Shannon. *European Journal of Information Systems*, 15(2), 200-213.

Holvitie, J., Leppänen, V., & Hyrynsalmi, S. (2014). Technical Debt and the Effect of Agile Software Development Practices on It - An Industry Practitioner Survey. *Managing Technical Debt (MTD), 2014 Sixth International Workshop on* (pp. 35-42). IEEE.

Hoskey, C. & Hoskey, A. (2016) Cultivating Sprightly Students: Using Agile Development in an Information Systems Capstone Course. *Information Systems Education Conference.* Pittsburgh, PA.

Jiménez, O. & Cliburn, D. (2016). Scrum in the Undergraduate Computer Science Curriculum. *Journal of Computing Sciences in Colleges*, 31(4), 108-114.

Lang, G. (2016). Agile Learning: Sprinting through the Semester. *EDSIG Conference*. Las Vegas, NV.

Mahnic, V. (2012). A Capstone Course on Agile Software Development using Scrum. *IEEE Transactions on Education,* 55(1), 99-106.

Maruping, L. M., Venkatesh, V., & Agarwal, R. (2009). A Control Theory Perspective on Agile Methodology Use and Changing User Requirements. *Information Systems Research*, 20(3), 377-399.

Masood, Z., Hoda, R., & Blincoe, K. (2018). Adapting Agile Practices in University Contexts. *Journal of Systems and Software*, 144, 501-510.

May, J., York, J., & Lending, D. (2016). Play Ball: Bringing Scrum into the Classroom. *Journal of Information Systems Education*, 27(2), 87-92.

McAvoy, J. & Sammon, D. (2005). Agile Methodology Adoption Decisions: An Innovative Approach to Teaching and Learning. *Journal of Information Systems Education*, 16(4), 409-420.

Overhage, S. & Schlauderer, S. (2012). Investigating the Long-Term Acceptance of Agile Methods: An Empirical Study of Developer Perceptions in Scrum Projects. *45th Hawaii International Conference on System Science (HICSS)*, 5452-5461.

Pope-Ruark, R. (2012). We Scrum Every Day: Using Scrum Project Management Framework for Group Projects. *College teaching*, 60(4), 164-169.

Ramasubbu, N., Bharadwaj, A., & Tayi, G. K. (2015). Software Process Diversity: Conceptualization, Measurement, and Analysis of Impact on Project Performance. *Management Information Systems Quarterly*, 39(4), 787-807.

Rubin, K. (2012). *Essential Scrum: A Practical Guide to the Most Popular Agile Process.* Addison-Wesley Signature Series.

Schwaber, K. (1995). Scrum Development Process. *OOPSLA'95 Workshop on Business Object Design and Implementation.* Austin, TX.

Scrum Case Studies. (2017). Retrieved March 11, 2018, from http://www.scrumcasestudies.com/.

Senapathi, M. & Srinivasan, A. (2014). An Empirical Investigation of the Factors Affecting Agile Usage. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. ACM.

Sutherland, J. & Schwaber, K. (2016). The Scrum Guide. *The Definitive Guide to Scrum: The Rules of the Game. scrum.org*.

Takeuchi, H. & Nonaka, I. (1986). The New Product Development Game. *Harvard Business Review,* 64(1), 137-146.

Valacich, J. & George, J. (2017). *Modern Systems Analysis and Design*. Pearson Education.

Version One. (2018). *12th Annual State of Agile Report*. Retrieved June 15, 2018, from https://www.stateofagile.com/#ufh-i-423641583-12th-annual-state-of-agile-report/473508.

Wagh, R. (2012). Using Scrum for Software Engineering Class Projects. *AGILE India*, 68-71, IEEE.

Weber, E. (2016). Performance Learning of Agile Methodology Using Paired Courses of Systems Analysis and Design and Web / Mobile Programming. *EDSIG Conference.* Las Vegas, NV.

West, D., Grant, T., Gerush, M., & D'silva, D. (2010). Agile Development: Mainstream Adoption has Changed Agility. *Forrester Research*, 2(1), 41.

Weimer, M. (2006). *Enhancing Scholarly Work on Teaching and Learning: Professional Literature that Makes a Difference*. Indianapolis, IN: Jossey-Bass.

Yue, K., De Silva, D., Kim, D., Aktepe, M., Nagle, S., Boerger, C., Jain, A., & Verma, S. (2009).Building Real World Domain-Specific Social Network Websites as a Capstone Project. *Journal of Information Systems Education*, 20(1), 67-76.

AUTHOR BIOGRAPHY

Corey Baham is an assistant professor of Management Science and Information Systems at the Spears School of Business at Oklahoma State University. He completed his Ph.D. in Information Systems and Decision Sciences from Louisiana State University. His current research focuses on agility in IS development, systems recovery, and firm dexterity. His work has been published in the *Journal of Management Information Systems, Communications of the Association of Information Systems*, and major IS conference proceedings.

Appendix A: Sample Class Schedule

Day 1	Group Project Overview Introduction of the Scrum Framework and Team Roles	
Day 2	Introduction of the Scrum Framework and Team Roles (continued) ***Sprint 0: Developing the Product Backlog*** • Preliminary Planning ○ Baseline Project Plan	

Day 1	○ Project Scope Statement
	• Begin User Story Workshop
	  ○ Brainstorm
	  ○ Map according to roles	

Day 2	(Complete User Story Workshop w/Product Owner) ○ Create Product Backlog *End **Sprint 0***	

	Transition to Scrum Workflow
	• Scrum Workflow
	  ○ Sprint Planning [Beginning of Sprint *1*]
	  ○ Sprint Work	

Day 1	***Class duration – 75 minutes:***
	• Class begins:
	  ○ Scrum Meeting (5-10 min.)
	  ○ Sprint Work (50-60 min.)
	  ○ Training session (5-10 min.)
	• Class ends
		WEEK 1

Day 2	***Class duration – 75 minutes:***
	• Class begins:
	  ○ Sprint Review w/Product Owner
	  ○ Sprint Retrospective [End of Sprint *1*]
	  ○ Sprint Planning [Begin Sprint *2*]
	• Class ends	

Day 1	***Class duration – 75 minutes:***
	• Class begins:
	  ○ Scrum Meeting (5-10 min.)
	  ○ Sprint Work (50-60 min.)
	  ○ Training session (5-10 min.)
	• Class ends
		WEEK 2

Day 2	***Class duration – 75 minutes:***
	• Class begins:
	  ○ Sprint Review w/Product Owner
	  ○ Sprint Retrospective [End of Sprint *2*]
	  ○ Sprint Planning [Begin Sprint *3*]
	• Class ends	

Table-A: Sample Class Schedule

Appendix B: Team Assignments

Class 1					
Group 1	Front end coding:	Back end coding:	Database design:	Systems infrastructure:	Avg. Col. Score
Student 1	3	4	4	2	
Student 2	4	2	3	3	
Student 3	2	4	3	5	
Student 4	4	3	3	2	
Column Total	13	13	13	12	
					12.75
Group 2					
Student 5	3	4	5	5	
Student 6	4	3	2	2	
Student 7	4	3	4	3	
Student 8	3	3	4	2	
Column Total	14	13	15	12	
					13.5
Group 3					
Student 9	4	4	4	3	
Student 10	3	3	3	3	
Student 11	3	2	3	3	
Student 12	3	3	4	4	
Column Total	13	12	14	13	
					13
Group 4					
Student 13	2	2	3	2	
Student 14	4	4	4	3	
Student 15	3	4	4	3	
Student 16	2	4	3	3	
Column Total	11	14	14	11	
					12.5

Class 2					
Group 5	Front end coding:	Back end coding:	Database design:	Systems infrastructure:	Avg. Col. Score
Student 17	4	3	3	2	
Student 18	3	3	3	2	
Student 19	3	3	4	2	
Student 20	4	4	4	3	
Column Total	14	13	14	9	
					12.5
Group 6					
Student 21	3	4	3	4	
Student 22	4	2	4	3	
Student 23	3	2	4	2	
Student 24	3	3	3	2	
Column Total	13	11	14	11	
					12.25
Group 7					
Student 25	2	3	4	2	
Student 26	3	3	4	2	
Student 27	4	4	4	3	
Student 28	3	3	2	2	
Column Total	12	13	14	9	
					12
Group 8					
Student 29	4	4	4	3	
Student 30	4	4	4	3	
Student 31	3	2	3	2	
Student 32	4	3	3	2	
Column Total	15	13	14	10	
					13
Group 9					
Student 33	3	3	4	2	
Student 34	3	3	3	3	
Student 35	4	4	4	3	
Column Total	10	10	11	8	
					9.75

Appendix C: Scrum Knowledge Assessment

Instructions: Please rate each question and its components from A to E as follows:

A = Strongly disagree

B = Somewhat disagree

C = Neutral

D = Somewhat agree

E = Strongly agree

1. Prior to this course, I was ____ of Scrum principles and practices.

A	B	C	D	E
Not knowledgeable at all	Not very knowledgeable	Neutral	Somewhat knowledgeable	Very knowledgeable

Questions 2-7: Currently, I have an adequate knowledge of ____.

2. Scrum Meeting

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

3. Sprint Planning

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

4. Sprint Review

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

5. Sprint Retrospective

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

6. Product Backlog Grooming

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

7. Sprints

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

Questions 8-13: My team has an adequate knowledge of _____.

8. Scrum Meeting

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

9. Sprint Planning

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

10. Sprint Review

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

11. Sprint Retrospective

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

12. Product Backlog Grooming

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

13. Sprints

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

14. Overall, I am (now) knowledgeable of Scrum principles and practices.

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

Questions 15-20: My team executed _____ as designed.

15. Scrum Meeting

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

16. Sprint Planning

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

17. Sprint Review

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

18. Sprint Retrospective

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

19. Product Backlog Grooming

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

20. Sprints

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

21. I feel that *doing* Scrum enhanced my knowledge of Scrum.

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

22. I feel comfortable doing Scrum at a future job.

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

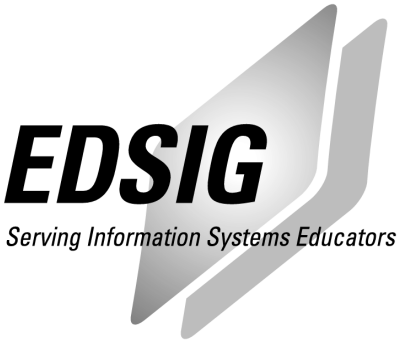
23. I feel that doing Scrum enhanced my knowledge of SDLC principles.

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

24. I would feel comfortable in a systems analyst position.

A	B	C	D	E
Strongly disagree	Somewhat disagree	Neutral	Somewhat agree	Strongly agree

Information Systems & Computing
Academic Professionals



STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 2574-3872