Journal of Information Systems Education

Volume 29
Issue 2
Spring 2018

Fostering Cooperative Learning with Scrum in a Semi-Capstone Systems Analysis and Design Course

Alejandra Magana, Ying Ying Seah, and Paul Thomas

Recommended Citation: Magana, A., Seah Y. Y., & Thomas, P. (2018). Fostering Cooperative Learning with Scrum in a Semi-Capstone Systems Analysis and Design Course. *Journal of Information Systems Education*, 29(2), pp. 75-92.

Article Link: http://jise.org/Volume29/n2/JISEv29n2p75.html

Initial Submission:	16 August 2017
Accepted:	30 January 2018
Abstract Posted Online:	21 March 2018
Published:	13 June 2018

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

Fostering Cooperative Learning with Scrum in a Semi-Capstone Systems Analysis and Design Course

Alejandra J. Magana
Ying Ying Seah
Paul Thomas
Computer and Information Technology
Purdue University
West Lafayette, IN 47907, USA
admagana@purdue.edu, yseah@purdue.edu**,** pjosekut@purdue.edu

ABSTRACT

Agile methods such as Scrum that emphasize technical, communication, and teamwork skills have been practiced by IT professionals to effectively deliver software products of good quality. The same methods combined with pedagogies of engagement can potentially be used in the setting of higher education to promote effective group learning in software development classrooms. Therefore, the purpose of this study is to integrate both Scrum and cooperative learning guidelines into a systems analysis and design classroom to promote the skills of teamwork, communication, and problem-solving while learning systems analysis and design methods. This integration was implemented in a sophomore, semi-capstone design course where students were engaged in collaborative classroom activities. Two different approaches – overlapped approach and delayed approach – were used in two different semesters for this implementation. Based on the analysis of student performance in the course, student reflections on their team performance, and student overall perceptions of the teaching approach, this study suggests that the integration of cooperative learning and Scrum serves as guidance for students to effectively analyze and design software solutions, as well as to reflect on their team performance and learning process. In addition, a delayed approach for Scrum implementation appears to effectively support student learning by providing better and earlier feedback.

Keywords: Cooperative learning, Scrum, Systems analysis & design, Student performance

1. INTRODUCTION

It has been widely acknowledged that professionals in Information Technology (IT) are required to possess knowledge and skills that allow them to solve organizational problems by working with technically-oriented peers, as well as by interacting with end users involved in the functional areas of the organization (Koong, Liu, and Liu, 2002; Sivitanides et al., 1995). This necessitates IT professionals to possess technical knowledge as well as soft skills such as communication, problem-solving, and teamwork (Bailey and Stefaniak, 1999). In addition, writing and presentation delivery, along with interpersonal and management skills, are critical for success in the IT profession (Wilkins and Noll, 2000). Such a combination of skills has been identified as relevant by industry experts and academicians alike (Aasheim, Li, and Williams, 2009). As a response to these needs, bodies of program accreditation such as ABET (2016) have now identified not only the required knowledge and skills that information technology graduates should exhibit, but also the attitudes and behaviors needed to confront complex problems. For instance, ABET (2016) considers the criteria "an ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs" as equally important as "an ability to function effectively on teams to accomplish a common goal." Accordingly, IT programs need to identify ways in which students can effectively be exposed to this broad range of skills sooner and more often throughout their undergraduate programs of study.

One way in which industry IT professionals can effectively combine such technical, communication, and teamwork skills for the delivery of quality software products is facilitated by Agile methods such as Scrum. Scrum (Takeuchi and Nonaka, 1986) is composed of a set of guidelines for implementing Agile project management with the goal of creating tested and usable results within weeks. Originally introduced in the context of Japanese manufacturing by Takeuchi and Nonaka in 1986, Scrum as an Agile framework for software development was introduced in the mid-to-late 1990s (Schwaber and Beedle, 2002). Variations or extensions of Scrum, such as Scrum/XP Hybrid, have appeared since then. However, a recent survey with thousands of companies responding revealed that the most commonly used Agile methodology is Scrum, with 58% of companies using it, with 10% of companies using Scrum/XP Hybrid (VERSIONONE.COM, 2017). Schwaber and Beedle

(2002) also reported Scrum to be the most prevalent Agile methodology in the IT industry.

On the other hand, in higher education settings, technical knowledge, communication, problem-solving, and teamwork skills can be effectively combined through classroom-based pedagogies of engagement (Smith et al., 2005). Pedagogies of engagement simultaneously promote teamwork and students' involvement in their own learning (Smith et al., 2005). A means for implementing pedagogies of engagement that promote small-group learning is cooperative learning (Smith et al., 2005). This study combines both Scrum and cooperative learning guidelines (Johnson, Johnson, and Smith, 1998b) with the goal of promoting teamwork, communication, and problem-solving skills, with the learning of systems analysis and design methods. We argue that bringing together best practices for managing software development projects from industry with best practices for meaningful group learning from academia can result in synergistic ways for implementing teamwork learning approaches in large classrooms. This paper, therefore, describes the design elements of a sophomore course in the department of Computer and Information Technology at Purdue University, CNIT 280 Systems Analysis and Design Methods, which aims at delivering authentic learning experiences for undergraduate students early in their academic careers.

We situate CNIT 280 as a semi-capstone design course where students are expected to demonstrate and integrate knowledge and skills learned in this specific course with those acquired from previous courses along with IT workplace techniques and frameworks for documenting and managing software projects. The guiding research questions for this study are: 1) What are students' levels of achievement on a systems analysis and design course that integrates learning and Agile methods through a semester-long project? 2) What are students' team reflections on their learning and performance as a team working on a semester-long project facilitated with Agile methods? and 3) What are students' perceptions of a systems analysis and design course that integrates cooperative learning and Agile methods through a semester-long project?

2. AGILE APPROACHES TO TEACHING AND LEARNING IN SYSTEMS DEVELOPMENT COURSES

The use of Agile methodologies, in general, and Scrum, in particular, in software engineering educational contexts is not new. Previous work in educational settings that studied 49 capstone projects revealed that Agile approaches were more appealing to student teams and resulted in greater project success in terms of customer expectations being met by the final software product (Umphress, Hendrix, and Cross, 2002). Mahnic (2012) reported overwhelmingly positive student perceptions about an undergraduate capstone course in software engineering where Agile software development using Scrum was introduced. Kamthan (2016) advocated for the use of Scrum in software engineering courses to improve collaboration within teams and to equip students with the practical experience of Agile methodologies. For instance, Master's degree programs have adapted Agile methods into their capstone courses based on observations that the use of Agile methods along with customer collaboration and programming ability had resulted in better productivity and website quality (Rico and Sayani, 2009). Shukla and Williams (2002) recommended that different Agile methodologies must be assessed by educators and integrated into courses. Their discussion was based on the introduction of extreme programming in a senior level, software engineering course at North Carolina State University. Coupal and Boechler's (2005) experience with independent external projects undertaken by final year, computer systems technology students pointed to the Agile approach supporting learning while providing practical experience to students within an academic environment, with usable software being the product.

3. PEDAGOGICAL FRAMEWORK

Cooperative learning refers to a pedagogical approach that promotes small-group learning and gives recognition based on group performance (Slavin, 1980). Cooperative learning has been identified as an effective pedagogical approach where "positive group-to-individual transfer of learning is a common result of collaborative interaction" (Sears and Pai, 2012, p. 2). A meta-analysis investigating instructional innovation in undergraduate science, technology, engineering, and mathematics (STEM) revealed that "various forms of small-group learning are effective in promoting greater academic achievement, more favorable attitudes toward learning, and increased persistence through STEM courses and programs" (Springer, Stanne, and Donovan, 1999, p. 21). In addition, associations between students' use of small-group learning strategies and students' self-efficacy for learning the course material, as well as course grade, have been found (Stump et al., 2011). Similarly, when integrating small-group learning approaches, positive effects have been regularly identified not only on student achievement outcomes (Slavin, 1980), but also on other outcomes such as "self-esteem, intergroup relations, acceptance of academically handicapped students, attitudes toward school, and ability to work cooperatively" (Slavin, 1991, p. 71).

Cooperative learning was used as the pedagogical framework that guided the integration of teamwork with Agile teaching and learning methods. Cooperative learning guided the design of this course by using Johnson, Johnson, and Smith's (1998a) five characteristics of cooperative learning as elements of the CNIT 280 course. Table 1 depicts an overview of the course design that aligns the five principles of cooperative learning that were implemented in the course. Additional details of the course implementation are presented in the next sections.

4. COURSE OVERVIEW AND CONTEXT

The Purdue Polytechnic Institute Transformation Implementation plan 2014-2017 (2015) calls for the need to produce more graduates who can meet the evolving needs of industries and communities. Specifically, it requests the integration of authentic learning experiences that are student-centered, delivered via active learning pedagogies, and situated in meaningful contexts providing students with opportunities to acquire knowledge and practices in the modes of a discipline. Cooperative learning has been identified as an approach that can effectively implement active learning, student-centered learning, problem-based learning, and project-based learning (Gol and Nafalski, 2007).

Principle	Definition	Course Implementation
Positive interdependence	The group has a clear task or goal.	The project was divided in clear milestones and deliverables (see Appendices A and B). The deliverables for the project are established from the very beginning of the semester.
Individual and group accountability	The group is accountable for achieving its goals. Each member must be accountable for contributing a fair share of the work.	Students are expected to work together as a team throughout all milestones and deliverables of the project. However, as part of the project, students are also expected to contribute individually. Two-thirds of the project are graded as a team but about a third of it is graded individually. In addition, students perform a self and peer evaluation at the end of the semester.
Interpersonal and small group skills	Basic teamwork skills: as a group, provide effective leadership, make decisions, build trust, communicate, and manage conflict.	Students are expected to work in-class and out-of-class throughout the entire semester. However, during class time, the instructor and TAs monitor group performance and facilitate conflict resolution if needed.
Face-to-face promotive interaction	A group member teaches classmates about a topic.	As part of the individual portion of the project students become specialized in one aspect of the system. However, in order for them to complete the prototype, everyone must understand the system functionality as a whole. In addition, students utilize in-class time to work on the project and help each other as a team.
Group processing	As a group, make decisions about which behaviors to continue and which behaviors to change.	Team retrospectives are used as a mechanism for group processing. For every milestone students are asked to reflect on what went well and what challenges they encountered. They commit to improve at least one team behavior from milestone to milestone.

Table 1. Alignment between Principles of Cooperative Learning and Elements of the Course

The design of the CNIT 280 systems analysis and design course embodies some of the elements of the transformation implemented via cooperative learning. It also includes approaches used by today's information system developers to discover and model the requirements and then construct an acceptable design to implement a successful system solution and a functional prototype. Course emphasis focuses on techniques that a programmer or analyst uses to develop information systems, such as object-oriented tools and the Unified Modeling Language (UML). In addition, this course surveys other important skills for a systems analyst, such as fact-finding, communications, project management, and cost-benefit analysis. The elements of the Purdue Polytechnic transformation integrated into CNIT 280 are the following:

4.1 Theory-Based Applied Learning

The course is designed as an active learning rather than a passive learning experience. The students are responsible for exploring and gathering relevant information and then constructing meaningful personal experiences that add to their own individual knowledge. The instructor's role is to establish parameters and facilitate the learning process. The instructor, therefore, is not the primary source of information, but one of many potential sources available to students. As such, this class relies heavily on student participation in the form of classroom collaborative activities, discussions, and projects. This is similar to an inverted classroom approach (Gannod, Burge, and Helmick, 2008), also called the flipped classroom approach, in the sense that class time is devoted to projects and other active learning approaches. However, it is also different because students did not have to access video lectures online. Lectures were delivered during class time and were, in addition, combined with team exercises and practice.

4.2 Learning in Context

As part of the course, students are engaged in a semester-long design experience where they apply tools and techniques practiced during class-time via mini cases, to then transfer their knowledge and skills to a more complex design problem. Students continuously work on their design problem inside and outside of the classroom, which culminates with a design specification documentation, a functional prototype, a usability evaluation of their prototype, and a sales presentation along with a prototype walkthrough.

4.3 Modernized Teaching Methods

A typical class is composed of 15-20 minutes of lecture, 10-15 minutes of practice and feedback, and 30 minutes of term-project work. At the beginning of the class, the instructor presents an agenda of the day followed by an introduction of the topic. Then, the instructor presents mini-cases for students to solve in teams. Mini-cases describe certain functionality of a system that usually has the scope of a complete use case narrative. Students work in teams on the mini-cases using whiteboards, while the instructor and graders walk around the classroom providing feedback on their work. Then, the instructor solves/models the mini-case for students using a document camera.

For the rest of the class time, students have time to attempt to apply the new knowledge or skills to their term-project design problem. Students work in teams on their solutions, and the instructor, teaching assistant, and graders walk around the classroom providing feedback and guidance to students. At the end of the class, students are required to model their solution using a CASE tool and submit it via Blackboard the following day for grading. Figure 1 and Figure 2 show instances of a typical day during in-class teamwork.

Figure 1. Students Brainstorming Requirements for their Design Project

Figure 2. Students Generating a Product Backlog for their Projects

4.4 Team Project-Based Learning

The course implemented a Scrum approach (Rising and Janoff, 2000) to software development for the enactment of the teams' prototypes. Scrum is an iterative and incremental approach to product development where teams work as a unit to reach a common goal. Students worked in teams of five members where they analyzed individual requirements, documented them, and implemented them into a functional prototype. Following this approach, students created their prototypes throughout the entire semester where they delivered their project in increments at the end of every one or two weeks. Students started their analysis by identifying a set of requirements that were organized by priority in the product backlog. Each item in the product backlog was then organized into user stories. Students selected a set of user stories to be implemented and delivered as a project increment. Each project increment is called a sprint. In addition, within each major delivery, students also reflected on the process as a team by performing a team retrospective. The team reflected on the previous deliverable and identified and agreed on continuous process improvement actions. Once the sprint was delivered, the sprint review was performed by the teaching assistant.

Scrum practices promoted team collaboration and reflection by implementing specific roles (Kamthan, 2016). The three most important roles in a Scrum approach are:

- Development team, performed by teams as a unit, where students did the work of creating a product.
- Product owner, rotated by all team members throughout each sprint. A student was responsible for keeping the Gantt chart and product backlog for that sprint.
- Scrum master, also rotated by each team member, was the person responsible for supporting the development team, enabled communication during meetings, and facilitated conflict resolution.

In addition to their prototypes and evaluation of their prototypes, students generated a comprehensive design document. The document specified the systems request, systems requirements, and systems specification using the Unified Modeling Language (Rumbaugh, Jacobson, and Booch, 2004) to visualize the functional, structural, and behavioral views of the system. The design document was delivered in the form of four major milestones and a final project. The solution of the design problem concluded with self and peer evaluations. Appendices A and B describe the elements students had to complete as part of each milestone.

4.5 Formative Feedback and Course Assessment

Throughout the entire semester, students for both approaches had the opportunity to receive formative feedback on their sprints and milestones. That is, in each of the sprints, students could incorporate elements of the feedback received in the previous sprints. During class time, the teaching assistant met with each group to discuss the feedback and address clarifications. To expose students to the iterative nature of systems analysis and design, students had an opportunity to revise and resubmit their milestones within one week after receiving detailed feedback. Each sprint was assessed based on students' ability to clearly align the user stories with the functionalities implemented through their prototypes. Students also generated a navigation map which also had to align with the navigation of the prototype. The design document was assessed based on students' ability to 1) accurately and thoroughly apply project management techniques, 2) accurately identify and analyze system requirements and present those as user stories, 3) accurately construct UML models to represent the functional, structural, and behavioral views of the system, and 4) professionally prepare and present an organized final report. As explained in Table 1, components of the final report were evaluated individually and also as a team. Appendices A and B describe the components evaluated for each of the milestones. At the end of the semester, students submitted a final design document compiling all previous milestones along with an executive summary.

The course also implemented traditional, individual assessment methods including weekly quizzes delivered online as well as three exams where students demonstrated their acquisition of conceptual knowledge as well as modeling skills. The course final grade was calculated as follows: eight online quizzes (10%), class participation in the form of attendance (15%), three conceptual and modeling exams (20%), functional prototype and interface usability evaluation (20%), final project group grade including all milestones (20%), final project individual grade (10%), and self and peer evaluations (5%).

5. METHODS

This design-based research presents two iterations of an implementation of Agile teaching methods to promote elements of cooperative learning.

5.1 Participants

Participants of this study included two cohorts of a systems analysis and design methods course offered in the Fall of 2016 and Spring of 2017. Each cohort had 100 students. As mentioned earlier, students from this course were undergraduate learners with most of them majoring in computer and information technology. Table 2 presents detailed information of students' majors for both implementations of the course. In the Fall of 2016, 1 student was a freshman, 52 were sophomores, 32 were juniors, and 15 were seniors. In the Spring of 2017, 47 students were sophomores, 41 were juniors, and 12 were seniors. The Fall of 2016 class consisted of 81 male students and 19 female students. The Spring of 2017 class consisted of 84 male students and 16 female students.

Major	Fall of 2016	Spring of 2017
Computer & Information Technology	83	81
Computer Engineering	1	0
Computer Graphics Technology	1	1
Computer Science	0	1
Electrical Engineering Technology	2	2
Explorers	0	1
Network Engineering Technology	8	9
Pre Management/Management	2	0
Systems Analysis & Design	3	4
Undesignated	0	1

Table 2. Number of Students per Major

5.2 Procedures

The integration of Scrum with cooperative learning was implemented slightly differently in the Fall of 2016 than in the Spring of 2017. The differences in the implementation process were based on student feedback as well as observations and feedback gathered from the instructional team. In both semesters, the instructional team consisted of one instructor, one graduate teaching assistant with industry experience, and two undergraduate graders who had previously taken the course and who excelled at demonstrating modeling skills while taking the course. The two approaches for implementing Scrum with cooperative learning are herein called the *overlapped approach* in the Fall of 2016 and the *delayed approach* in the Spring of 2017.

The *overlapped approach* included all the design elements described in Section 4 Course Overview and Context. However, sprints for the implementation and delivery of the prototype and milestones for the design document documenting the functional, structural, and behavioral views of the system overlapped throughout the entire semester (see Figure 3). In addition, each sprint was incremental and delivered every two weeks. Specifically, in each sprint, students were asked to specify detailed functionality of two user stories each time and generate the corresponding functionalities in the prototype.

Milestones were also delivered incrementally, meaning that only user stories identified in each sprint were considered as part of the documentation required for each milestone. For example, as indicated in Appendix A, Milestone 2 required students to detail the user stories through use-case narratives, corresponding activity diagrams, and corresponding class diagrams up to sprint 3. Since students were requested to work on only two user stories in each sprint, they needed to only deliver those elements for the first six user stories. However, students had to keep adding two detailed user stories for each remaining milestone (i.e., Milestone 3 and Milestone 4).

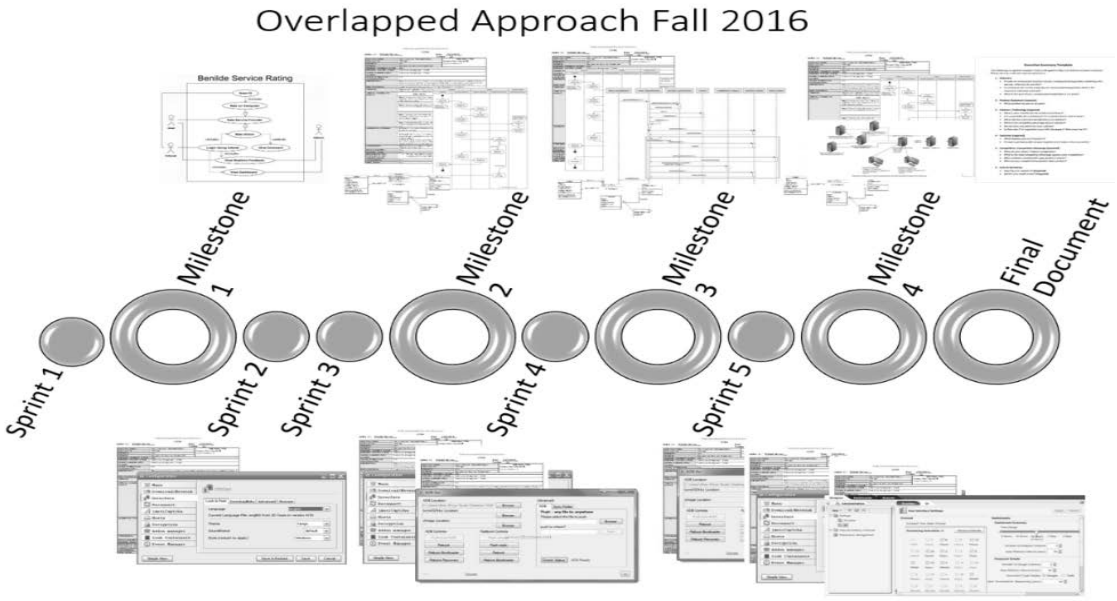

Figure 3. Alignment between Project Documentation and Prototype Development (Fall 2016)

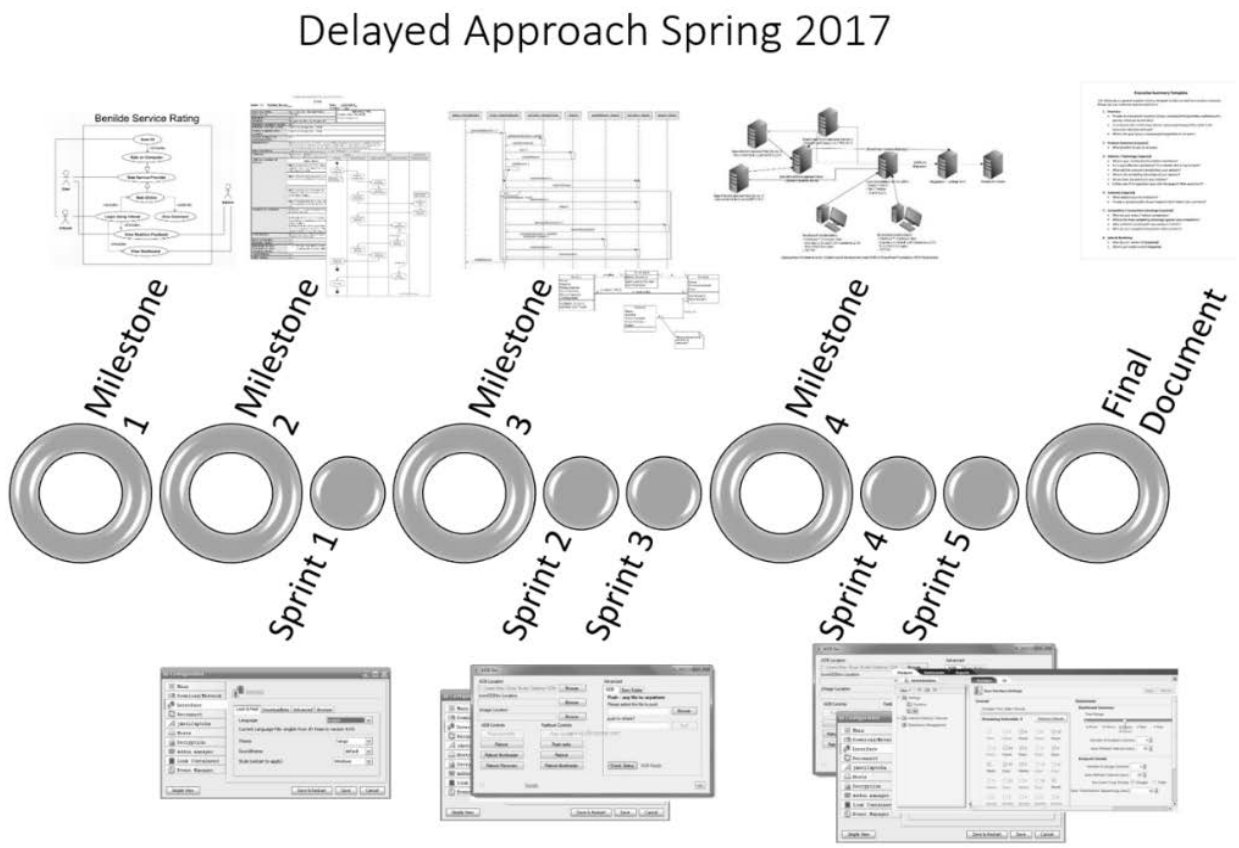

Figure 4. Alignment between Project Documentation and Prototype Development (Spring 2017)

The *delayed approach* also included all the design elements described in Section 4. However, in this approach, sprints were delayed until after students had detailed all their user stories into detailed use case narratives (see Figure 4). That is, students were not able to start on their prototypes until all user stories were documented and approved by the instructional team. In this implementation, the sprints were delivered every single week. In each sprint, students were asked to implement the corresponding functionality detailed in the use case narrative in their prototypes, and in addition, keep track of their projects using Kanban. Kanban is a Japanese method of a flow control used in manufacturing as a scheduling system (Ahmad, Markkula, and Oivo, 2013). This approach has been adopted in software development due to its benefits in improving "lead time to deliver software, improved quality of software, improved communication and coordination, increased consistency of delivery, and decreased customer reported defects" (Ahmad, Markkula, and Oivo, 2013, p. 10). The description of the deliverables required for each milestone for the Spring of 2017 implementation is detailed in Appendix B.

The main distinction between these two approaches was that in the *overlapped* approach students were uncovering user stories two by two at a time and implementing them incrementally every two weeks from the beginning of the semester. On the other hand, the *delayed* approach guided students to first identify all requirements described as user stories in their product backlog before actually starting to construct their prototypes toward the middle of the semester.

5.3 Data Collection and Data Analysis Methods

This study hypothesized that combining both Scrum Agile approaches with cooperative learning guidelines will promote student systems analysis and design knowledge as well as develop student teamwork skills. We, therefore, focus on three main constructs as evidence of the effectiveness of this approach: student performance in the course, student reflections on their team performance, and student overall perceptions of the teaching approach. Table 3 presents an alignment between the research questions, data collection method, and data analysis methods for each of the three constructs.

6. RESULTS

Results from this study are presented by research question identifying similarities and differences between the overlapped approach (Fall of 2016) and the delayed approach (Spring of 2017) in terms of student performance in the course, student reflections on their team performance, and student overall perceptions of the teaching approach.

Research Question	Data Collection Method	Data Analysis Method
Construct: Student academic performance		
What are students' level of achievement in a systems analysis and design course that integrates learning and Agile methods through a semester-long project?	Student performance on elements of the course: • Overall final grade as evidence of overall student performance throughout the semester. • Final project as evidence of student ability to document the analysis and design of a system. • Total average of the five sprints as evidence of students' ability to translate a system design into a working prototype.	Data was analyzed using descriptive statistics including measures of central tendency and variability (mean and standard deviation). Inferential statistics (Wilcoxon-Mann-Whitney U-Test) was used to identify possible performance differences between Fall of 2016 and Spring of 2017.
Construct: Student reflections on team performance		
What are students' team reflections on their learning and performance as a team working on a semester-long project facilitated with Agile methods?	Team reflection in the form of Scrum team retrospective: Evaluation of the team about their performance on the milestone just finished: • What went well in this milestone? • What went wrong in this milestone? • What are possible concerns? • What should we keep doing? Plan of action: • How to improve people? • How to improve processes? • How to improve tools? • You must commit to have something to improve every milestone.	Qualitative analysis was performed on team reflections. Open coding was used to identify categories. Axial coding was then used to identify themes. Patterns were then then quantified by identifying frequencies or counts of themes.
Construct: Student perceptions of the course		
What are students' perceptions of a systems analysis and design course that integrates cooperative learning and Agile methods through a semester-long project?	Final teaching evaluation multiple choice survey (sample questions): • The instructor employs effective teaching methods. • The instructor gives valuable feedback on each student's performance. Final teaching evaluations open-response questions: • What is something/are some things that the instructor does well, e.g., something you hope that the instructor will continue to do in the class in the future? • Make a suggestion(s) for improving the course (a criticism alone is not helpful; tell your instructor how you would fix any problem).	Student survey data analyzed using descriptive statistics including measures of central tendency and variability (i.e., mean and standard deviation). Inferential statistics (two sample-t test) used to identify possible differences between groups. Qualitative analysis performed on student open-response questions. Open coding used to identify categories. Axial coding then used to identify themes.

Table 3. Alignment between Research Questions and Data Collection and Data Analysis Methods

6.1 Student Academic Performance

This section responds to the first research question: What are students' levels of achievement in a systems analysis and design course that integrates learning and Agile methods through a semester-long project? The Mann-Whitney U-Test was used to check for significant differences in academic performance between students in the Fall of 2016 and Spring of 2017 semesters. As shown in Table 4, academic performance was measured in terms of final project scores, prototype sprint scores, and final course grade. Effect size as given by Cohen's d was computed using the point-biserial correlation estimate (Ivarsson et al., 2013).

The Mann-Whitney test showed that the final project score in the Spring of 2017 semester (Median = 96) was significantly greater than the final project score in the Fall of 2016 semester (Median = 83), $U = 3092.5$, $p < 0.0001$. The effect size given by Cohen's $d = 1.06$ indicated a large practical significance. The test also revealed that the total sprints score in the Spring of 2017 semester (Median = 100) was significantly greater than the total sprints score in the Fall of 2016 semester (Median = 94), $U = 1695$, $p < 0.0001$. The effect size given by Cohen's $d = 3.01$ indicated a large practical significance. The Mann-Whitney test also indicated that the final grade in the Spring of 2017 semester (Median = 87.55) was significantly greater than the final grade in the Fall of 2016 semester (Median = 84.68), $U= 3799.5$, $p < 0.005$. The effect size given by Cohen's $d = 0.59$ indicated a moderate or medium practical significance.

The overall academic performance was proficient in both semesters. However, performance in the Spring of 2017 was significantly higher than in the Fall of 2016 in the three overall measures, with moderate to extremely large practical significance.

	Semester								
	Fall of 2016			Spring of 2017					
	Median	Range	N	Median	Range	N	U	P	d
Final Project	83.00	40.00	100	96.00	20.00	99	3092.5	< 0.01	1.06
Total Sprints	94.00	30.00	100	100.00	40.00	99	1695.0	< 0.01	3.01
Final Grades	84.68	47.01	100	87.55	32.40	99	3799.5	< 0.01	0.59

Table 4. Academic Performance Comparing the Overlapped Approach (Fall 2016) and the Delayed Approach (Spring 2017)

6.2 Student Reflections on Team Performance

This section responds to the second research question: What are students' team reflections on their learning and performance as a team working on a semester-long project facilitated with Agile methods? A qualitative analysis was performed on students' team reflections for each of their four milestones. Since each team was composed of 5 students and each course offering had 100 students, a total of 20 teams and their reflections were analyzed. Because the team reflections were qualitatively analyzed, not all teams discussed the same themes. Based on this analysis, five main themes were identified: time management, teamwork, communication among team members, quality of work, and progress toward project completion. Table 5 below describes each of these themes, their definitions, and sample quotes. In addition, Table 6 depicts the counts of each of the identified themes per team and for each of the milestones.

Themes	Definition	Sample Quote
Time Management	Episodes where students reflected on scheduling and on-time task completion. They talked about how they scheduled group meetings as well as how they went about making sure they adhered to the project deadlines.	Some possible concerns is team scheduling. Since we have five members who are all involved in various activities and classes besides this one, it can be difficult to find a time suitable for all of us which may lead to some team members doing more work than others. This milestone, that hurt us in particular. We also had a tough time completing this on time because it was due the night Thanksgiving Break started.
Teamwork	Episodes where students reflected on how well they worked together as a team. Things that students touched on included fair share of work, contribution of different skills, as well as efficiency of work.	During this milestone as a group we worked very well in small groups. We split off into smaller groups to work on the different parts of the project which worked a lot better than all of us working on the same part of the project at the same time. After we finished whatever part we have chosen to work on, we were able to pass it over to another person within the group to be able to check over the work so that everyone is in agreement with the work.
Communication	Episodes where students reflected on the efficiency and effectiveness of communication within and outside the team. This included communication between team members and communication between the team and the instructional team. They talked about medium of communication, conflict resolution, clarity of exchanged messages, as well as frequency of communication.	We had no trouble dividing the work almost ourselves, and communication was constant and conductive. Our communication, while already strong, could be used better to make sure everyone is completing the work in a timely manner.
Quality of Work	Episodes where students reflected on the quality of their work. They discussed about the accuracy of their work as well as the consistency of their work across milestones.	We are concerned that our estimates in the Cash Flow diagram might not be as accurate as we would like. Another concern is that the information included in our Gantt chart is not specific enough.
Project Progress	Episodes where students reflected on their progress in the project. They mentioned things like which part of a milestone they have completed and what their next step was in terms of project completion.	Milestone 3 brought together some of the final pieces of the project. We were able to update our product backlog to represent more closely how close everything is and how long they will take to complete. The team has worked hard to finalize activity diagrams for all of the use case narratives. Along with the activity diagrams the team has completed sequence diagrams.

Table 5. Themes Emerged from Team Performance from Both Approaches

Theme	Semester									
	Fall of 2016					Spring of 2017				
	Total	Total (M1)	Total (M2)	Total (M3)	Total (M4)	Total	Total (M1)	Total (M2)	Total (M3)	Total (M4)
Time Mgmt.	35	9	6	9	11	45	8	12	14	11
Teamwork	37	12	7	7	11	52	11	14	13	14
Communication	27	8	5	6	8	33	6	9	9	9
Quality of Work	26	5	7	5	9	11	0	2	5	4
Project Progress	27	3	5	6	13	21	3	7	3	8

Table 6. Overall Count of Themes from Student Reflections Comparing the Overlapped Approach (Fall of 2016) and the Delayed Approach (Spring of 2017)

Results from Table 6 suggest that students in the Spring of 2017 semester reflected more on their time management and teamwork skills. On the other hand, students from the Fall of 2016 semester reflected more on the quality of their work. Students from both semesters reflected about the same number of times on their communication strategies and project progress.

6.3 Student Perceptions of the Course

This section responds to the third research question: What are students' perceptions of a systems analysis and design course that integrates cooperative learning and Agile methods through a semester-long project? Students from both semesters responded to a final teaching evaluation where they rated statements about the instructor and the course on a Likert scale with the options Strongly Disagree, Disagree, Undecided, Agree, and Strongly Agree. These were assigned scores from one to five with the extremes of the scale (strongly disagree and strongly agree) assigned scores of one and five, respectively.

The students from the Spring of 2017 semester reported more positive perceptions of the course, the instructor, and the teaching methods employed as compared to the students from Fall of 2016 semester. This is shown in Table 7 with a much larger percentage of responses recorded by the students of Spring of 2017 for the agree and strongly agree options as compared to those of Fall of 2016. As recommended by McCrum-Gardner (2008), Mann-Whitney U-test was used for analyzing ordinal data recorded using Likert scales. Resulting p-values were less than 0.05, thereby indicating a significant difference in perceptions between the Spring of 2017 and Fall of 2016 semesters.

As part of the final teaching evaluations, students were also given the opportunity to provide feedback to the instructor. Students' comments were categorized in positive comments, constructive comments, and complaints. Although we do not provide in-depth detail over such comments, we offer here representative comments from each semester. Overall positive comments from both semesters are consistent. Students appreciated in-class exercises, detailed feedback, and teamwork. For instance, samples of quotes include:

> I really enjoyed being able to resubmit milestones if our group received a poor grade the first time we submitted it. This helped our grade a lot, and I think it also helped us learn the material better by learning from our mistakes and correcting them. I also very much appreciated having class time to work on the project, since it was difficult to find times outside of class when everyone could meet. I found our TA to be very patient and helpful with our sprints and milestones. (Student, Fall of 2016)

> She was prepared for every class. There was a great amount of time to work on the projects, which was great (depending on teammates showing up). I was able to use the diagrams in other classes. I know how to use those in real life applications and on development teams I won't feel lost trying to understand the documentation, which is great. Also, she really made the class engaging and made sure everyone was on the same page before totally moving on which really helped. (Student, Spring of 2017)

From our own reflections of the course offered in Fall of 2016, and from students' comments such as the one below, it was clear that students felt the first implementation of the course following an overlapped approach was confusing:

> Some of the problem in the class were the material was very confusing and it was not clear what we were learning in the class. The classroom itself did not fit the needs of the students it was very tough to understand what the instructor was doing because of the unusual setup of the classroom. The amount of work in the class, the class needs a lab in it. (Student, Fall of 2016)

7. DISCUSSION AND IMPLICATIONS FOR TEACHING AND LEARNING

This study implemented and compared two approaches, an *overlapped* approach and a *delayed* approach, for the integration of Scrum practices to promote cooperative learning in a systems analysis and design course. The implementation of the two approaches were compared in terms of academic achievement, teams' perceptions of their performance, and students' perceptions of the course. In terms of these three constructs, findings from this study suggest that in terms of student achievement, the *delayed* approach was more supportive of student learning. In terms of teams' perceptions of their performance, students from both approaches reflected equally about their communication strategies and project progress. However, students in the *delayed* approach reflected more on their time management and teamwork skills, while students in the *overlapped* approach reflected more on the quality of their work.

	Semester									
	Fall of 2016					Spring of 2017				
	SA	A	U	D	SD	SA	A	U	D	SD
The instructor's course materials are helpful	12.5	39.6	12.5	16.7	19.0	36.2	37.7	11.6	13.0	1.4
The instructor employs effective teaching methods and techniques	10.4	37.5	16.7	18.8	17.0	33.3	30.4	20.3	13.0	2.9
The instructor teaching is creative and innovative	12.5	38.3	21.3	10.6	17.0	31.9	30.4	17.4	17.0	2.9
The instructor demonstrates how to apply concepts and methodologies	16.7	45.8	12.5	10.4	15.0	33.8	42.6	14.7	7.4	1.5
The instructor presents sufficient and relevant examples	16.7	41.7	14.6	16.7	10.0	35.8	40.3	11.9	10.0	1.5
The instructor relates course material to industry	12.5	47.9	16.7	6.2	17.0	33.8	41.2	11.8	10.0	2.9
The instructor the instructor's tests or assignments are relevant to the subject	12.5	48.9	6.4	14.9	17.0	33.3	52.2	8.7	2.9	2.9
The instructor gives valuable feedback on each student's performance	12.5	33.3	14.6	27.1	13.0	36.2	33.3	15.9	10.0	4.3
The instructor provides help and suggests ways for students to improve	20.8	43.8	16.7	10.4	8.3	31.2	50.6	13.0	3.9	1.3
The instructor explains difficult material clearly	8.3	30.4	15.2	23.9	22.0	27.5	36.2	15.9	17.0	2.9
The instructor is effective in instruction	10.4	39.6	14.6	20.8	15.0	26.5	42.6	14.7	13.0	2.9

Note. SA – Strongly Agree, A – Agree, U – Undecided, D – Disagree, SD – Strongly Disagree

Table 7. Percentage (%) Distribution of Student Responses Comparing the Overlapped Approach (Fall of 2016) and the Delayed Approach (Spring of 2017)

Finally, considering students' perceptions of the course, students from the *delayed* approach on average reported positive perceptions of the course, the instructor, and the teaching methods employed. In contrast, students from the *overlapped* approach were undecided on their perceptions of the usefulness of the course materials and the overall effectiveness of the instructors' teaching methods including examples, tests, explanations, and feedback.

We hypothesize that differences in students' performance, team perceptions of their performance, and students' perceptions of the course, in general, can be attributed to the students' level of uncertainty experienced throughout the semester in terms of the organization of the course, as well as their performance in their work regarding the solution of the case study. The Control-Value Theory of Achievement Emotions (Pekrun, 2006) posits that students' appraisals of their control are central to the activation of achievement emotions affecting their levels of engagement and consequently their achievement. Specifically, students in the *overlapped* approach might have felt confused or might have experienced uncertainty pertaining to the course structure, as well as their limited understanding of the case study. For instance, as shown in the last quote from the previous section, some students felt unclear about the unusual setup of the classroom. This discomfort was also evidenced in the final teaching evaluations where students found the course materials and the instructor's explanations difficult to understand (see Table 7). Also, students in the *overlapped* approach actually reflected more about the quality of their work (as opposed to team management and planning), possibly suggesting that they were uncertain or non-confident of their performance, and therefore felt the need to reflect on it. On the other hand, students in the *delayed* approach might have had a clearer understanding of the case study before moving onto the prototype. Similarly, each milestone in the *delayed* approach was self-contained in terms of the scope, and not incremental as in the *overlapped* approach, giving students a better sense of control.

The implications for teaching and learning relate to the need of timing formative feedback and guidance to support students' design processes (Shute, 2008). In both implementations of the

course, students were provided with verification and elaboration feedback. Verification feedback was provided using a rubric. Students were informed about their correct or incorrect approaches to systems analysis and design. Elaboration was provided via detailed corrections in every milestone specifically telling the students what needed to be addressed (Black and Wiliam, 1998). Although students submitted their design documents through the course management system, each team was required to print their documents and deliver them in the next class for detailed correction of the diagrams and overall documentation. As mentioned earlier, teams were allowed to revise and resubmit their documentation within the next week after receiving the feedback. What did change from both approaches (i.e., overlapped and delayed) was the timing of the feedback relative to the time students started with their prototypes. While students in the *delayed* approach started their prototype once they received feedback on all identified user stories and requirements of the system, the students in the *overlapped* approach started their sprints before receiving feedback on their requirements and the feedback they received was more incremental. The timing of the feedback received might have had an impact on students' appraisals of their control and therefore their perceptions of and performance in the course.

8. CONCLUSION, LIMITATIONS AND FUTURE WORK

Findings from our study suggest that cooperative learning combined with Scrum can effectively guide students in analyzing and designing software solutions. Other studies that implemented collaborative group projects with Scrum have also identified that this combination allows students to effectively frame, plan, and manage group projects (Pope-Ruark, 2012). Our study suggests that in addition, the implementation of team retrospectives allowed students to reflect not only on their learning process, but also on aspects of team performance such as time management, communication, quality of work, and progress toward completion.

Our study also presented two approaches for integrating cooperative learning with Scrum: an *overlapped* approach and a *delayed* approach. Although the *overlapped* approach may be closest to the way Scrum is applied in industry settings, results from our study suggest that for learning purposes, a *delayed* approach for Scrum implementation may support better student learning allowing for timely verification and elaboration feedback. As a result, students may have a better sense of control in their learning and consequently feel better prepared to activate their levels of engagement. That is, better guidance can be provided to students when they have an opportunity to receive feedback in their preliminary analysis before moving on into aspects of design and implementation. Evidence of that includes better performance on the final project, the implementation of their functional prototypes, and their overall course performance. In addition, students reported overall positive perceptions of the course because they thought it was better organized and that more guidance, feedback, and opportunities to revise their work were provided to them.

Limitations of our study relate to the potential impact of our variations of Scrum for learning purposes. However, we are confident that students benefited from their learning and are now better prepared for internships and future careers. For instance, in the Summer of 2017, the course instructor received the following email from a student with the subject "Thank you":

> Dr. Magana,
> My name is [student's name] and I just took your class (CNIT 280) this past spring and I just wanted to take a second to thank you for being a great professor. Honestly, when I first started your class I did not know how useful the information that I would learn in your class would be but, I was wrong. This summer I am interning at Intel Corp and the first day of my internship I was introduced to our Scrum master and was told that my team followed an Agile Framework method. I was shocked because being a cybersecurity major I did not think I would be using user stories and everything else during my internship. Thanks to your class I was able to skip an entire day of training regarding how to write a user story and how the entire process works and I was also able to help my team be more efficient in the way we track our stories.
> Thank you once again and I hope you are having a wonderful summer.

We will continue making improvements in the course based on feedback from students, the curriculum committee, and our advisory board to continuously adapt the course to industry needs. Specifically, in the near future, we will continue our work toward a more rigorous application of the Scrum methodology integrating other practices such as stand-up meetings, burn charts to track the velocity of the team, and so forth.

9. ACKNOWLEDGEMENTS

We thank Jeffrey Brewer, associate professor of CIT; Kevin Dittman, associate professor of CIT; and Jeffrey Whitten, professor of CIT, for their continuous feedback in improving the course. We would also like to show our gratitude to the Purdue Polytechnic Institute for their support with transformation funds for curricular innovations.

10. REFERENCES

ABET. (2016). Criteria for Accrediting Engineering Programs Effective for Reviews During the 2017-2018 Accreditation Cycle. Retrieved from http://www.abet.org/wp-content/uploads/2016/12/E001-17-18-EAC-Criteria-10-29-16-1.pdf.

Aasheim, C. L., Li, L., & Williams, S. (2009). Knowledge and Skill Requirements for Entry-Level Information Technology Workers: A Comparison of Industry and Academia. *Journal of Information Systems Education*, 20(3), 349-356.

Ahmad, M. O., Markkula, J., & Oivo, M. (2013). Kanban in Software Development: A Systematic Literature Review. *Proceedings from the Software Engineering and Advanced Applications (SEAA), 2013 39th EUROMICRO Conference*, 9 - 16.

Bailey, J. L. & Stefaniak, G. (1999). Preparing the Information Technology Workforce for the New Millennium. *ACM SIGCPR Computer Personnel*, 20(4), 4-15.

Black, P. & Wiliam, D. (1998). Assessment and Classroom Learning. *Assessment in Education: Principles, Policy & Practice*, 5(1), 7-74.

Coupal, C. & Boechler, K. (2005). Introducing Agile into a Software Development Capstone Project. *Proceedings from the Agile Conference, 2005*, Denver, CO.

Gannod, G. C., Burge, J. E., & Helmick, M. T. (2008). Using the Inverted Classroom to Teach Software Engineering. *Proceedings of the 30th International Conference on Software Engineering*.

Gol, O. & Nafalski, A. (2007). Collaborative Learning in Engineering Education. *Global Journal of Engineering Education*, 11(2), 173-180.

Ivarsson, A., Andersen, M. B., Johnson, U., & Lindwall, M. (2013). To Adjust or not Ddjust: Nonparametric Effect Sizes, Confidence Intervals, and Real-World Meaning. *Psychology of Sport and Exercise*, 14(1), 97-102.

Johnson, D. W., Johnson, R. T., & Smith, K. A. (1998a). *Active Learning: Cooperation in the College Classroom*. ERIC.

Johnson, D. W., Johnson, R. T., & Smith, K. A. (1998b). Cooperative Learning Returns to College: What Evidence is there that it Works? *Change: The Magazine of Higher Learning*, 30(4), 26-35.

Kamthan, P. (2016). On the Nature of Collaborations in Agile Software Engineering Course Projects. *International Journal of Quality Assurance in Engineering and Technology Education (IJQAETE)*, 5(2), 42-59.

Koong, K. S., Liu, L. C., & Liu, X. (2002). A Study of the Demand for Information Technology Professionals in Selected Internet Job Portals. *Journal of Information Systems Education*, 13(1), 21.

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

McCrum-Gardner, E. (2008). Which is the Correct Statistical Test to Use? *British Journal of Oral and Maxillofacial Surgery*, 46(1), 38-41.

Pekrun, R. (2006). The Control-Value Theory of Achievement Emotions: Assumptions, Corollaries, and Implications for Educational Research and Practice. *Educational Psychology Review*, 18(4), 315-341.

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

Purdue Polytechnic Institute. (2015). *Purdue Polytechnic Institute Transformation Implementation Plan August 2014 – December 2017 (Updated August 24, 2015).* West Lafayette, IN: Purdue University.

Rico, D. F. & Sayani, H. H. (2009). Use of Agile Methods in Software Engineering Education. *Agile Conference, 2009. AGILE'09*, 174-179.

Rising, L. & Janoff, N. S. (2000). The Scrum Software Development Process for Small Teams. *IEEE Software*, 17(4), 26-32.

Rumbaugh, J., Jacobson, I., & Booch, G. (2004). *The Unified Modeling Language Reference Manual*. Pearson Higher Education.

Schwaber, K. & Beedle, M. (2002). *Agile Software Development with Scrum (Vol. 1)*. Upper Saddle River, NJ: Prentice Hall.

Sears, D. A. & Pai, H.-H. (2012). Effects of Cooperative Versus Individual Study on Learning and Motivation after Reward-Removal. *The Journal of Experimental Education*, 80(3), 246-262.

Shukla, A. & Williams, L. (2002). Adapting Extreme Programming for a Core Software Engineering Course. *Proceedings from the 15th Conference on Software Engineering Education and Training (CSEE&T 2002)*, Covington, KY.

Shute, V. J. (2008). Focus on Formative Feedback. *Review of Educational Research*, 78(1), 153.

Sivitanides, M. P., Cook, J. R., Martin, R. B., Chiodo, B. A., & Landram, F. (1995). Verbal Communication Skills Requirements for Information Systems Professionals. *Journal of Information Systems Education*, 7(1), 38-43.

Slavin, R. E. (1980). Cooperative Learning. *Review of Educational Research*, 50(2), 315-342.

Slavin, R. E. (1991). Synthesis of Research of Cooperative Learning. *Educational Leadership*, 48(5), 71-82.

Smith, K. A., Sheppard, S. D., Johnson, D. W., & Johnson, R. T. (2005). Pedagogies of Engagement: Classroom-Based Practices. *Journal of Engineering Education*, 94(1), 87-101.

Springer, L., Stanne, M. E., & Donovan, S. S. (1999). Effects of Small-Group Learning on Undergraduates in Science, Mathematics, Engineering, and Technology: A Meta-Analysis. *Review of Educational Research*, 69(1), 21-51.

Stump, G. S., Hilpert, J. C., Husman, J., Chung, W. T., & Kim, W. (2011). Collaborative Learning in Engineering Students: Gender and Achievement. *Journal of Engineering Education*, 100(3), 475-497.

Takeuchi, H. & Nonaka, I. (1986). New Product Development Game. *Harvard Business Review*.

Umphress, D. A., Hendrix, T. D., & Cross, J. H. (2002). Software Process in the Classroom: The Capstone Project Experience. *IEEE Software*, 19(5), 78-81.

VERSIONONE.COM. (2017). *11th Annual State of Agile Report*. Retrieved from http://stateofagile.versionone.com/.

Wilkins, M. L. & Noll, C. L. (2000). Critical Skills of IS Professionals: Developing a Curriculum for the Future. *Journal of Information Systems Education*, 11(3-4), 105-110.

AUTHOR BIOGRAPHIES

Alejandra J. Magana is an Associate Professor in the Department of Computer and Information Technology with a courtesy appointment at the School of Engineering Education at Purdue University. Her research program investigates how model-based cognition in Science, Technology, Engineering, and Mathematics (STEM) can be better supported by means of expert technological and computing tools such as cyberinfrastructure, cyber-physical systems, and computational modeling and simulation tools.

Ying Ying Seah is a Ph.D. student in the Department of Computer Information Technology at Purdue University. Her research interests include system analysis and design education, K – 12 STEM learning, educational CAD, student experimentation strategies, and design thinking.

Paul Thomas is a Ph.D. student in the Department of Computer Information Technology at Purdue University. His research interests include project management practices, gamification in education, and student engagement.

Appendix A: Description of Project Milestones and Deliverables Fall of 2016

Milestone	Deliverables
Milestone 1	• *Introduction:* Who are you and what is your requirement analysis strategy (BPA, BPI, BPR)? • *Project Vision Statement:* What do you want your end product to be? • *Context Diagram:* Inputs/Outputs to and from the system. • *System Request:* project sponsor, business need, business requirements, business value and special constraints. • *Product Roadmap:* (optional) a picture of your post-it notes. • *Product Backlog:* the master to-do list considering input from.. • *Use-case Diagram:* A diagram that represents the interactions between actors and use cases, including the relationship among use cases, and relationships between actors. • *Team Retrospective:* Evaluation of the milestone just finished and plan of action.
Milestone 2	• *Updated product backlog:* Created from the requirements from product roadmap. • *Documentation up to Sprint 3:* Include use-case narratives, corresponding activity diagrams, and corresponding class diagram. • *Gantt Chart:* Include the estimates for each sprint. It should be updated along the process and delivered for each milestone. • *Cash Flow:* Financial cost-benefit analysis. • *Team Retrospective:* Evaluation of the milestone just finished and plan of action.
Milestone 3	• *Updated product backlog*: Created from the requirements from product roadmap. • *Documentation up to Sprint 4:* Include updated use-case narratives, corresponding activity diagrams, corresponding class diagram, and corresponding sequence diagrams. • *Updated Gantt Chart:* Include the estimates for each sprint. It should be updated along the process and delivered for each milestone. • *Team Retrospective:* Evaluation of the milestone just finished and plan of action.
Milestone 4	• *One-Page Executive Summary:* highlights of the main points of the problem and main points of your proposed solution o Start by describing the mission of the company and briefly describe the problem they have. o Describe your solution. You may want to start by stating the project vision statement and then the system you are proposing. It would be a good idea to describe here your architecture design (web-based, cloud-based, software, hardware, etc.). o Briefly describe how features of your system address the company's problem. (You can state them as a paragraph or as bullet points). o Provide details about your estimated timeline to complete the system as well as the overall cost. o Conclude by stating your competitive advantage. • *Updated product backlog:* Created from the requirements from product roadmap. • *Documentation up to Sprint 5:* Include updated use-case narratives, corresponding activity diagrams, corresponding class diagram, and corresponding sequence diagrams. • *Packages:* Group class diagram into packages. • *Entity Relationship Diagram:* Design your data storage mechanism. • *Updated Gantt Chart:* Include the estimates for each sprint. It should be updated along the process and delivered for each milestone. • *Team Retrospective:* Evaluation of the milestone just finished and plan of action
Final design document	• *One-Page Executive Summary* (highlights of the main points of the problem and main points of your proposed solution) • *Table of Contents* • *All revised milestones* • *Updated Product Backlog* • *Deployment Diagram:* Describe the physical layer where your system will be installed and create a deployment diagram. Describe each component (e.g., servers, devices, etc.) providing the specifications of each of them and the communication protocols and type of network. • *Revised Cash Flow* • *Revised Gantt Chart* • Screen shots of the final product (working software) • *Executable file or link of the final product* (include username and password, if applies) • *Evidence of preliminary usability testing of the prototype* o Discuss strengths and weaknesses of your prototype.

Usability evaluation report	• Evidence of preliminary usability testing of the prototype. Suggested steps: o Select the dimensions you want to evaluate (e.g., navigation, links, layouts, etc.). My suggestion is to focus on 4 to 5 most important dimensions. o Samples of dimensions or questions can be found here: https://stayintech.com/info/UX. You can come up with your own rubric. o Select the specific items (questions) that will evaluate each dimension. My suggestion is to keep 4 to 5 questions per dimension. Make sure you attach your usability survey. o Ask each member of the team to evaluate the prototype individually. Ask at least one friend to evaluate it for you (minimum 6 evaluations, ideal 10 evaluations). o Make sure you report final number of evaluators. o Calculate average scores per dimension. o Discuss strengths and weaknesses of your prototype.
Sales presentation and final walkthrough	• Required presentation format (10 minutes per team +2 for questions) o Product Vision o Strategy proposed & Justification (BPA, BPI, BPR) o Physical design including equipment specifications o Product demonstration o Results preliminary usability testing o Estimated cost (Cash Flow) o Estimated timeline (Gantt chart) o Team's retrospective

Appendix B: Description of Project Milestones and Deliverables Spring of 2017

Milestone	Deliverables
Milestone 1	• *Introduction:* Who are you and what is your requirement analysis strategy (BPA, BPI, BPR)? • *Project Vision Statement:* What do you want your end product to be? • *Context Diagram:* Inputs/Outputs to and from the system. • *System Request:* project sponsor, business need, business requirements, business value and special constraints. • *Product Roadmap:* a picture of your post-it notes listing and prioritizing requirements • *Product backlog:* the master to-do list considering input from Product Roadmap. See guidelines for more info. • *Use-case Diagram:* A diagram that represents the interactions between actors and use cases, including the relationship among use cases, and relationships between actors. • *Team Retrospective:* Evaluation of team performance during the milestone just finished and plan of action.
Milestone 2	• *Updated product backlog:* Created from the requirements from product roadmap. o State each requirement as a user story. • *Use-Case narratives:* Describe in detail each user story including the ideal course of event and at least one alternate course of event (more than one if needed). o Each team member should build at least two use-case diagrams • *Gantt Chart:* Include the estimates for each sprint. It should be updated along the process and delivered for each milestone. • *Cash Flow*: Financial cost-benefit analysis. • *Team Retrospective*: Evaluation of the milestone just finished and plan of action.
Milestone 3	• *Updated product backlog:* Created from the requirements from product roadmap. • *Class diagram:* Identify the classes for your solution and the relationships among them. Build the class-diagram including attributes, relations and cardinality/multiplicity. • *Activity Diagrams:* Each team member should work on his or her two use case narratives. For each use case narrative, build the corresponding activity diagram. o Each team member should build at least two activity diagrams • *Sequence Diagrams:* Each team member should work on his or her two use case narratives. For each use case narrative, build a sequence diagram for at least two of the scenarios in them. o Each team member should build at least four activity diagrams • *Updated Gantt Chart:* Include the estimates for each sprint. It should be updated along the process and delivered for each milestone. • *Team Retrospective:* Evaluation of the milestone just finished and plan of action
Milestone 4	• *One-Page Executive Summary:* highlights of the main points of the problem and main points of your proposed solution o Start by describing the mission of the company and briefly describe the problem they have. o Describe your solution. You may want to start by stating the project vision statement and then the system you are proposing. It would be a good idea to describe here your architecture design (web-based, cloud-based, software, hardware, etc.). o Briefly describe how features of your system address the company's problem. (You can state them as a paragraph or as bullet points). o Provide details about your estimated timeline to complete the system as well as the overall cost. o Conclude by stating your competitive advantage. • *Updated product backlog:* Created from the requirements from product roadmap. • *Packages:* Group class diagram into packages. • *Entity Relationship Diagram:* Design your data storage mechanism. • *Updated Gantt Chart:* Include the estimates for each sprint. It should be updated along the process and delivered for each milestone. • *Team Retrospective:* Evaluation of the milestone just finished and plan of action
Final design document	• *One-Page Executive Summary* (highlights of the main points of the problem and main points of your proposed solution) • *Table of Contents* • *All revised milestones* • *Updated Product Backlog* • *Deployment Diagram:* Describe the physical layer where your system will be installed and create a deployment diagram. Describe each component (e.g., servers, devices, etc.) providing the specifications of each of them and the communication protocols and type of network.

	• *Revised Cash Flow* • *Revised Gantt Chart* • Screen shots of the final product (working software) • *Executable file or link of the final product* (include username and password, if applies) • *Evidence of preliminary usability testing of the prototype* o Discuss strengths and weaknesses of your prototype.
Usability evaluation report	• Evidence of preliminary usability testing of the prototype. Suggested steps: o Select the dimensions you want to evaluate (e.g., navigation, links, layouts, etc.). My suggestion is to focus on 4 to 5 most important dimensions. o Samples of dimensions or questions can be found here: https://stayintech.com/info/UX. You can come up with your own rubric. o Select the specific items (questions) that will evaluate each dimension. My suggestion is to keep 4 to 5 questions per dimension. Make sure you attach your usability survey. o Ask each member of the team to evaluate the prototype individually. Ask at least one friend to evaluate it for you (minimum 6 evaluations, ideal 10 evaluations). o Make sure you report final number of evaluators. o Calculate average scores per dimension. o Discuss strengths and weaknesses of your prototype.
Sales presentation and final walkthrough	• Required presentation format (10 minutes per team +2 for questions) o Product Vision o Strategy proposed & Justification (BPA, BPI, BPR) o Physical design including equipment specifications o Product demonstration o Results preliminary usability testing o Estimated cost (Cash Flow) o Estimated timeline (Gantt chart) o Team's retrospective

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