Involving IT professionals in Scrum student teams: An empirical study on the impact of students' learning

Panagiotis K. Linos, Ph.D.
Computer Science & Software Engineering
Butler University, Indianapolis, USA
linos@butler.edu

Ryan Rybarczyk, Ph.D.
Computer Science & Software Engineering
Butler University, Indianapolis, USA
rrybarcz @butler.edu

Nathan Partenheimer
Information Technology
Butler University, Indianapolis, USA
npartenh @butler.edu

Abstract— **This is a full paper under the Innovative Practice category. It presents an empirical study that describes our efforts, reflections and lessons learned from including Information Technology (IT) professionals in an undergraduate Computer Science and Software Engineering (CSSE) course, which introduces Scrum as an agile software development methodology. We discuss how students, who took such a course, perceive their interaction with IT professionals as an overall positive learning experience and an opportunity to interject real world lessons and scenarios into course content. Moreover, we report on some useful observations and the feedback we received from the professionals, who volunteered and committed to engage regularly with our student Scrum teams throughout this course. We discuss how we leveraged the feedback we received, from both students and IT professionals, to incrementally fine-tune our course while teaching it. Based on the empirical data gathered during our study, we explain how this win-win collaboration has been beneficial to everyone involved. More specifically, we have found that this approach helps students better understand and apply important principles, methods and tools for software development, become familiar with team dynamics, communicate effectively with an IT professional and gain a better appreciation of the inherent challenges involved in crafting larger and realistic software applications. In addition, we observed that such an engagement provides not only some gratification to IT professionals, who have the opportunity to mentor our students both inside and outside of the classroom, but also an opportunity for them to refresh their knowledge on certain methodologies, technologies and tools used during our course. We conclude by discussing some of the instructor's challenges while teaching this course as well as our plans to continue improving it by strengthening the involvement of IT professionals in student teams.**

Keywords— Scrum, industry-academia collaboration, IT professional stakeholders, student teams, software engineering undergraduate education.

I. INTRODUCTION

Integrating current Software Engineering (SE) trends and best practices into the classroom has long been a desire of educators. One such framework that has been successfully taught in various academic institutions over the last decade is *Scrum*, which entails a lightweight agile software development process. A key observation of successful integration of Scrum in an academic setting is the commitment and reliance on software professionals.

As discussed in our next section, several efforts have focused on the implementation of Scrum in the classroom but not so much on a consistent inclusion of IT professionals and their impact on students' learning. Other similar efforts, which are also mentioned in the following section, have been made to better understand the importance of student collaborations in agile course projects. In addition, educators have studied the positive impact of student-led projects which are provided by industrial partners. However, and to the best of our knowledge, very little has been reported specifically on a reliable, consistent and on-going involvement of IT professionals, as mentors, within student team projects using Scrum.

In this paper, we highlight our research efforts aiming at filling that gap by providing a much more comprehensive, dependable and organized inclusion of IT professionals in the classroom. Moreover, we discuss how students, who took an undergraduate SE course using Scrum, perceive their interaction with IT professionals as a positive learning experience. In addition, we report on some of the constructive feedback we received from the IT professionals, who volunteered and committed to engage regularly with our students during their Scrum project as well as some of the key challenges faced by the instructor of this course.

The rest of this paper is organized as follows: Section 2 describes related research, followed by a third section that presents some background information about the nature and logistics of our SE course. Section 4 discusses our empirical study where we describe, analyze and summarize our empirical research findings. Finally, in Section 5, we conclude with an overview of our course implementation and further plans.

II. RELATED WORK

In the literature, we find several efforts made towards balancing software engineering education with industrial needs. For instance, Ana Moreno et al, report on a related study conducted to assist academics and software professionals to build a win-win relationship [1]. They make recommendations on what skills the industry is expecting from graduating software engineering students who are out seeking employment. Another study by Chen and Chong discusses a collaborative senior project between industry and academia [2]. They report on the findings related to teamwork, industry participation and how to sustain the collaboration between students and software professionals.

In addition, H.J.C. Ellis et al discuss how industry can collaborate with academia in order to re-educate non-software engineering professionals [3]. They acknowledge and emphasize the shortage of software engineering professionals and the need to re-train existing non-software professionals to become software engineers. Moreover, we see efforts made to engage in a dialogue between industry and academia about the importance of software engineering education. Such authors report on their experiences from teaching software engineering courses at a University as well as in industry. They propose five potential roles that industry can play which will help software engineering university professors [4].

In another effort, Nancy Mead suggests ways on how to re-fresh our strategies towards an effective industry-university collaboration [5]. More specifically the same author discusses the reasons why the current collaboration strategies don't work and emphasizes the need for new ones [6]. On the other hand, Lutz et al summarize their experiences from developing and sustaining the first software engineering undergraduate program in the USA. They discuss the lessons learned during the past few decades and their efforts to incorporate the involvement of industry professionals within the software engineering curriculum by establishing well-organized internships and co-op programs [7]. A recent comprehensive literature review on best practices and related challenges from the collaboration between software engineering academic programs and industry can be found in [8].

At the same time, the agile methodology has become a popular approach in solving the software development challenge throughout industry [9]. Change is inevitable, and this mantra fits well within the agile methodology and subsequently a software development process known as *Scrum* [10]. The philosophy of Scrum is such that customers tend to change their mind frequently throughout the design and development process and as part of the agile methodology, it promotes a lightweight iterative process. As this becomes a popular process and consistent trend in professional SE industry, we have chosen to focus on the role that agile software development and Scrum have played in existing courses as described through a literature review.

In [11], the author provides a systematic overview of the implementation of agile methods into existing courses. Key observations pertain to how to properly address real-world challenges and integrate them into an educational setting. The same author discusses in [12] the students' perceptions and some useful teacher's observations from teaching a SE course with an emphasis on a Scrum project. Another survey, found in [13], provides an overview of existing implementation attempts of Scrum into a classroom. We have used this survey as a starting point for our specific implementation to leverage its strengths and weaknesses. Previous work reported in [14] and [15], describes some specific Scrum implementations without however the inclusion of IT professionals and their impact on students' learning.

Baham discusses a recent effort to teach Scrum wholesale and makes specific recommendations including the requirement for students to deliver actual software, to acquire a Product Owner, to provide Scrum training, and to carefully adapt the Scrum method and use documentation to monitor it [16]. However, no discussion regarding the involvement of IT professionals playing the role of stakeholders is provided. In [17], the authors describe how they each implemented the Scrum philosophy into existing SE courses within their respective programs and outline the results of this effort. In [18] the author argues that creating a game to simulate such an exercise in order to effectively teach the benefits of Scrum can yield positive results in a classroom setting and address many of these concerns.

In [19], the authors describe the inclusion of industrial partners into a graduate-level course. Our work is in a similar vein in the proposed inclusion, where our work does differ is that our inclusion of this is in an introductory undergraduate software engineering course and thus this is the students first exposure to such topics as Scrum or agile development. Therefore, the industrial practitioners are able to take a much more hands on and active role in describing processes and really teaching the students from Day 1 the protocols and practices used and applied out in industry.

One of the common arguments that we found in our literature review, pertains to the inability to adequately implement Scrum into a classroom setting due to the nature of the various actors involved, and necessary, within the Scrum philosophy. Evidently, providing a realistic environment in which to implement this practice is difficult due to the nature of academia and the limited availability and commitment of IT professionals. In addition to this fact, the various actors must have a wide skill set and often their experience is vital.

In this paper, we attempt to leverage the experiences that others have found when attempting to implement any agile philosophy into an existing undergraduate course [20]. We are proposing to consistently include IT practitioners as "stakeholders" with faculty serving the role of the "product owner" and build teams in which each participant plays a key role in the overall development of a semester-long realistic team project. We have found that such inclusion, and the depth of the inclusion, is unique to the discipline and attempts to address many of the cited challenges when integrating an agile software development process, specifically Scrum, into an existing SE undergraduate course.

Finally, we have decided to prepare and launch an empirical study based on qualitative methods introduced by other software engineering researchers [21-23]. Such methods have proven to be effective when studying non-technical areas such as teamwork and team dynamics [24-25]. Moreover, the authors in [26] claim that the use of qualitative methods has been useful in studying computer science and software engineering education.

III. COURSE INFORMATION

Our Computer Science and Software Engineering (CSSE) department has approximately 150 majors and minors, five full-time and five part-time faculty members offering two separate four-year undergraduate degrees, one in CS and another in SE. We offer a required third-year introductory class on SE. Students enroll in this course after having taken four required courses: Discrete Math, Object-Oriented Programming, Data Structures, and Database Management Systems. This prerequisite structure is designed to ensure that the students not only have the necessary skills but also adequate academic maturity to enter such a SE course.

During this course, students are expected to learn and apply object-oriented design concepts, tools, and technologies by completing a realistic team project using Scrum. During the first day of class, all students are grouped into Scrum teams of about 5-7 students each. Every team is then matched with an IT professional, who is acting as a "stakeholder" throughout this project interacting and mentoring the team inside and outside the classroom regularly. The instructor, and other invited faculty members, play the role of "product owners." Student teams and their stakeholders are then expected to choose a realistic project to design, implement and test throughout the semester. The instructor provides a sample list of project options to the students to guide their selection. The product owner is the only individual who has the authority to change any of the initial requirements of the selected project. Some of the recommended projects include an Insurance Claims Management System, an IT Help Desk Ticket Management System, and a Workflow Management System. The students are provided with a complete and detailed set of initial requirements of their selected application. More details about these suggested projects can be found in [27].

Finally, we include in Table I a summary of all the learning objectives, technologies and resources listed in the syllabus of our course.

A. Scrum Ceremonies

During the first two weeks of class, Scrum teams are introduced to all Scrum ceremonies and related activities such as stand-up, planning, review and retrospective meetings. In addition, the instructor provides a detailed 7-Sprint schedule to the teams, where each "Sprint" is a 2-week time-boxed period for a total of 14 weeks in a typical academic semester. In the first week, teams and their stakeholders start with a planning meeting in order to elect a Scrum master (e.g., team leader), then select a software application to develop from the project choices provided by the instructor, decide and document the goals for their first Sprint, and finally assign tasks to each team member for the next two weeks.

During the planning meetings, the stakeholders help the teams identify and set appropriate Sprint goals. They also stress the importance of some key Scrum concepts such as what it means to be "done" with a task or a goal. Consequently, all teams must demonstrate a tangible, integrated, functioning and tested piece of software during each Sprint review and demo session. By the end of the semester, they are expected to complete and translate all the initial requirements to a fully functioning software application.

B. Instruction and Mentoring

All stakeholders and product owners are expected to participate in every planned Scrum meeting. For instance, during a typical class period that starts with a 10-minute stand-up meeting, the stakeholder and product owner join the Scrum master asking each team member questions like: *What are you currently working on? Are there any obstacles that prevent you from making progress? And what are you planning to do next?* After each stand-up meeting, the instructor will typically use the rest of the class time to cover related course material and deliver in-class tutorials on object-oriented design. However, if there is another scheduled Scrum meeting that day, it follows immediately after the stand-up session. Such meetings include a planning session (e.g., deciding on the next Sprint's goals and assigning effort estimates on them), code review (e.g., walking through code and explaining what it does and why, to peers, stakeholders and product owner), Sprint demo (e.g., formal presentation and demo of all completed Sprint goals), or a retrospective meeting at the end of each Sprint, where students discuss and document what has worked for their team, what has not worked.

TABLE I. SE Course Learning Objectives, Technologies and Resources

Learning Objectives	**Technologies and Tools**	**Resources and Readings**
Apply the Scrum agile methodology to develop realistic software applications	• MS Visual Studio IDE • C#.NET programming language	• Scrum textbook [28] • C#.NET textbook [29]
Derive user stories and related tasks from a given set of requirements	Best practices with user story templates taken from [28]: *As a <type of user>, I want <some goal> so that <some reason>.*	• Scrum textbook [28] • Functional requirements from [27]
Craft a layered (n-tier) architecture to create and organize object-oriented designs efficiently	• Visual Studio as the software development environment • C#.NET as the implementation programming language	• OO Design book [27] • In-class tutorials from textbook [29] • Instructor's lecture notes
Use UML to document, communicate and modify object-oriented designs effectively	• MS VISIO for drawing UML diagrams • Visual Studio's built-in UML drawing tool	• UML textbook [30] • Instructor's UML tutorials & guides
Experience team dynamics and learn effective team management	• GitHub for source code management • Slack for team communication [31]	• Trello or Kanban boards • MS Planner for project management
Understand software engineering professional standards	The Software Engineering Code of Ethics and Professional Standards Guide [33]	Clean Coder textbook by James Martin [32]
Develop effective oral communication and technical presentation skills	• By-weekly Power Point Sprint review presentations • Video recordings of Sprint reviews	University Speakers Lab tutorials and required lab visits

Early in the course, students are shown, by the instructor and with the help of stakeholders, how to use all the necessary tools including Visual Studio and C#.NET for software development, GitHub for version control, Visio for drawing UML diagrams, Trello for project management and Slack for team communication.

C. Performance Assessment

Peer assessment has been shown to be an effective mechanism to evaluate student progress during capstone projects [35]. Based on that, we decided to incorporate peer assessment during our course in order to evaluate the progress of our Scrum teams. More specifically, at the end of each Sprint review demo, everyone in the audience (except the members of the presenting team) including stakeholders, product owners, members of other teams and occasionally other guests are asked to complete an online form asking questions about the presenting team's performance. We used a score of 1-10 (1 very poor -- 10 excellent) to rate the presenting team by everyone who attended their review demos and provide a justification for such score. This anonymous scoring and related comments are then shared with the presenting team, who is expected to discuss and address them during their next Sprint. At the end of the semester all teams give a formal presentation (e.g., final Sprint review presentation). Everyone attending provides a score and constructive feedback at the end.

Overall, all students appeared to be comfortable with peer assessment and they welcomed the feedback received from the audience. It is fair to mention that at the beginning the teams asked for further clarification about their ratings from their peers. This was resolved by adjusting our feedback forms to include a section were a detailed justification was provided regarding their ratings. All teams were expected to address such constructive input and explain how it helped them improve their work during the following Sprint presentations.

All students earn a final grade in the course, which is the result of several deliverables including the completion of their Scrum team project (50%), assignments (10%), exams (15%), online discussions related to the readings from our textbooks (5%) and a final exam (20%).

IV. EMPIRICAL STUDY

The main purpose of this study is to solicit, collect, and analyze feedback from students and IT professionals, who collaborate in order to complete a semester-long Scrum team project. Another objective of this study is for the instructor of this class to create and maintain a journal of experiences and related challenges faced while teaching such a course. We report and discuss such challenges below as part of this study.

We start by focusing on students' perception of how well they learn Scrum, key object-oriented design concepts, and related software technologies. Next, we observe the impact on their understanding of the scale and complexity of real software systems and their realization of the fact that change is inevitable. Moreover, we look at the effect on their ability to learn on-the-job and communicate with a real professional. Finally, we study students' appreciation of team dynamics and their perception of professionalism.

To this end, we collected data by means of anonymous surveys from three course offerings during the spring 2018 and 2019 semesters. All the surveys were voluntary in nature and were not required as part of the course. From a total of 32 students, who enrolled in these sections and formed six Scrum teams, we received 17 comprehensive responses. In addition, we collected feedback from six IT professionals who engaged with the Scrum teams consistently throughout the course offerings.

All surveys were conducted through our online Moodle LMS (Learning Management System) portal. All student responses were saved and exported into spreadsheets containing both numeric data as well as brief textual answers. Due to space limitations in this paper, we only present selected brief responses as well as tabular summaries of all numeric data collected. A complete set of our data can be available upon request.

The student survey has five sections as shown in Appendix A. The first section sought information about the students' perception of the stakeholders as part of the project. The second section of the survey was to explore their comfort level not only working on a team but also their intrapersonal skills. Section 3 focused on the students' perception of how well they learn object-oriented design principles, as well as related software development tools and technologies. Moreover, in section 4 we asked open-ended questions about the students' perception of large, complex and realistic software applications. The final section served as a way for us to collect information regarding the students' background.

The data was collected during both the early part of the semester and the end of the semester. One of the benefits of collecting this data earlier in the semester was to make any necessary adjustments to the implementation method. This also allowed us to see the growth of our students' knowledge base.

We also included a systematic process of eliciting continuous feedback and suggestions from the stakeholders and product owners. Such recommendations reflected real world challenges and subsequent solutions that our IT professionals had faced. This list of suggestions was then distributed to the students and during weekly meetings these topics were discussed with both the stakeholder and product owner. The stakeholders' active engagement and commitment were vitally important to the overall success of this project.

A. Student Survey Data Analysis

In this section, we analyze and summarize all collected survey responses from our students. A Likert scale of 1-5 (1--Strongly Disagree, 5--Strongly Agree) was used from [34] in order to measure student responses to most of the survey questions with some requiring brief answers. In addition to the ratings, we provided open space for comments from students to solicit additional feedback. We describe all students' responses below in each survey section separately.

Survey Section 1: Experience with IT professionals

With this part of the survey we wanted to know how comfortable students felt interacting with their stakeholders, asking questions, receiving mentorship, expressing their own opinion etc. Their responses are summarized and depicted in Table II below. As we can see in the table, most students indicated that they felt comfortable engaging with their stakeholders and didn't have any problem communicating and asking them questions. Furthermore, high scores were given regarding the stakeholders' assistance learning Scrum and being effective mentors and role models.

TABLE II. Student average scores to survey section 1: Experience engaging with an IT professional

Survey Item	**Average Score (Likert scale 1-5)**
I feel comfortable engaging with my team's stakeholder (IT professional)	4.3/5.0
Our stakeholder is helping me better understand the Scrum methodology	4.5/5.0
Our stakeholder is helpful with the technologies and tools used in the project	4.4/5.0
I consider our stakeholder to be an effective mentor and role model	4.5/5.0
I can communicate effectively with our stakeholder	4.2/5.0
I can understand clearly our stakeholder	4.2/5.0
I feel comfortable asking our stakeholder questions	4.3/5.0
I am satisfied with the answers provided by our stakeholder	4.4/5.0

In addition, we asked our students to discuss their perception of their stakeholders' role and provide some suggestions for them. Some interesting related comments are included in Table III below. We shared these valuable anonymous suggestions with the stakeholders throughout the semester and made any necessary adjustments accordingly.

TABLE III. Selected student responses to survey section 1 questions

Q: Can you describe the role of your team's stakeholder?
"teammate mentor/expert"
"an authority in the subject matter who was helpful and receptive"
"He acted both as a teammate and an expert. Always available to help set up some of the more complex features of our application while at the same time giving unbiased and truthful feedback on the state of those features."
"I feel like the stakeholders were great! We got sent some cool articles to read. Our perspectives on some of the topics got shaken up. We received really good feedback!"
Q: Do you have any suggestions for your stakeholder?
"Don't be afraid to join in on the discussion or butt in where you have valuable insight--sometimes most valuable things you know are things we don't know to ask about"
"Sometimes I felt very intimidated because I felt like what he was saying was too advanced or over my head, so if the stakeholders could maybe try and bring things down to a simple level that would be helpful"

Survey Section 2: Teamwork

We observed the students' perception and experiences while working as a team. Our survey results are shown in Table IV including the average score of their responses. As we can see from Table IV below, most students indicated that they were happy overall with their participation in their team and they were able to take initiatives. Moreover, most students felt that they communicated effectively their needs to the rest of their team.

Low ratings were found regarding students' perception on their ability to identify and resolve conflict in their teams as well as being able to assist other team members. Another, lower than we hoped score, was found when students were asked if they are willing to share their own work with other teammates. This attitude of being reluctant to share their code with others became more apparent, and then discussed openly, during our code review sessions and at the end of each Sprint.

On the other hand, we received higher scores indicating that students felt that their non-technical skills (e.g., teamwork, communication, etc.) and work ethic were improved by the end of this course. Some other interesting responses shown in Table IV indicate that most students felt able to learn on-the-job while under pressure to deliver tangible results. They were also able to undertake a task and complete it on time. In addition, they better understood the fact that change is inevitable during software development and they are willing to embrace it.

TABLE IV. Student average scores to survey section 2: Teamwork

Survey Item	**Average Score (Likert scale 1-5)**
I am happy with my participation in our team's discussions	4.0/5.0
I can take initiatives during this project	4.1/5.0
I can identify and resolve conflict during this experience	3.8/5.0
I feel comfortable asking for help from my team	4.3/5.0
I can communicate my own needs to the rest of the team	4.3/5.0
I can assist other team members	3.6/5.0
I am a good listener during our meetings and discussions	3.8/5.0
I am willing to share my own work with others	3.6/5.0
I can learn on-the-job when under pressure to deliver tangible results	4.1/5.0
I am happy with my overall work ethic	4.4/5.0
I am effective with undertaking a task and completing it on time	4.0/5.0
My non-technical skills (e.g. teamwork, listening, communication, etc.) have improved due to my participation and experience from this project	4.2/5.0
I understand that change is inevitable while developing software and I am willing to embrace it	4.0/5.0
I have a good understanding of working with a real client	4.4/5.0
I have a good understanding of what it means to be a professional software engineer	4.0/5.0
Overall, I feel that I am a better team player from this experience	4.3/5.0

Survey Section 3: Learning OO design principles, related software development technologies and tools

The results of this section are included in Table V below. Most students indicated that they felt comfortable learning and using Scrum as well as the C#.NET programming language. Moreover, they were able to incrementally construct a layered (3-tiered) software architecture by the end of the semester for their final project.

We received some lower scores regarding their willingness to create UML class diagrams as shown in Table V. It is fair to mention here that students resisted the drawing of UML diagrams due to their complexity and because they became larger and larger while progressing on their project. Based on that, we introduced them to some other automatic drawing tools available, including one that comes packaged with Visual Studio.

Early in the semester, we noticed that some teams were challenged using GitHub for managing their code base especially with their application's database backend. However, after a steep learning curve, and the assistance of their stakeholders, all teams felt comfortable with it by the midpoint of the semester.

TABLE V. Student average scores to survey section 3: Learning OO design principles, related software development technologies and tools

Survey Item	Average Score (Likert scale 1-5)
I feel confident learning and using the Scrum software development methodology	4.2/5.0
I can create UML diagrams that depict easy-to-maintain OO designs	2.6/5.0
I am comfortable using a version control tool (e.g., GitHub)	2.8/5.0
Using a project management tool (e.g., Trello) is useful	4.3/5.0
I can use the C#.NET programming language to write realistic software	3.9/5.0
I understand how to create a layered (e.g. 3-tier) architecture to organize any OO designs	3.9/5.0

Survey Section 4: Perception of large, complex software applications

The rationale of this category was to observe whether the students' initial perception of a large and more complex software application had changed after completing this course. Therefore, we asked their opinion at the beginning of the semester and then we asked the same question at the end of this course. Some of their answers are shown in Table VI below, which provide some promising evidence that students gained a much better understanding, and appreciation, of what it means to work with a larger, more complex realistic software application.

TABLE VI. Selected student responses to survey section 4 questions

Question asked at the BEGINNING of the semester: What is your perception of large, complex software applications?
"It is very hard to do"
"Beyond my skill set"
Same question asked at the END of the semester:
"After this class I think that large and complex software applications are not as intimidating as they were when I first started, and I can better break them down into their requirements"
"There are many, many, many things that go into developing software that many people do not realize are involved in developing software. However, once these large pieces are broken down into smaller pieces and worked on by a team, large and complex software suddenly doesn't seem so large and complex."

B. Feedback from IT Professionals

We included six stakeholders who committed to engage with our Scrum teams and be present to all Scrum meetings. All of them are experienced software professionals. Four of them are currently employed by our University's IT center with the remaining two having full-time software engineering jobs in our city. They all have extensive experience using Scrum.

We conducted separate and regular meetings with the stakeholders and asked them questions pertaining to what challenges they were facing while interacting with our student teams. We sought their input to better guide the students through the process, bringing their wealth of experience into the course. Some specific examples of suggestions from our stakeholders and our response to them are described next.

a) Establish an early formal communication mechanism among teams. We addressed this concern immediately by asking all teams to adopt and use an effective communication tool such as Slack. They also included their stakeholders, in their Slack groups, who helped them establish and agree upon some important norms for effective communication.

b) Create a separate initial Sprint 0. This was a good idea since the students were learning a lot of new material during the first two weeks of the course. It was announced to the class that Sprint 0 wouldn't be graded and teams were expected to select goals mostly related to setting up all their development tools and platforms.

c) Improve student engagement with stakeholders. We observed that initially while students were trying to figure out their own role and that of their stakeholders, they were somewhat hesitant to actively engage with them. We therefore clarified more effectively their stakeholders' role and provided an actual card with the writing "What would and IT professional do?" We watched students using that card in multiple occasions where they were facing a challenge (technical or not). This proved to be an effective way to encourage them to engage more with their stakeholders.

d) Conduct regular code review sessions. Before each Sprint demo, all teams were asked to conduct a code review session. During these sessions, code authors would explain their part in detail to everyone in the team including their stakeholders and product owners. We observed that this provided a very useful opportunity for the teams to receive constructive feedback from the professionals, instructor and teammates. It is worth mentioning here that some students' survey responses indicated that this was one of the best learning experiences they had during this course.

e) Give a professional final presentation. All teams gave their final project presentation at a selected professional setting suggested by their stakeholders. The session was celebratory and included some light food and refreshments provided by the stakeholders' institution. In addition to the regular stakeholders, product owners and other students involved, more IT professionals were invited from the hosting company to watch the final team presentations. Everyone attending provided a score and constructive feedback at the end, which was forwarded to the teams anonymously. We received very positive feedback, especially from students, after this experience and observed a sense of student pride while presenting their accomplishments in front of such a professional audience.

Survey Section 5: Background Information

Most students were CS&SE with some double majors in Math, Marketing and EE. We received additional comments including *"I like the use of SCRUM to accomplish biweekly goals. The in-class tutorials are very helpful to understand the software engineering concepts we need to learn.."*

C. Stakeholders' Benefits

We asked the IT professionals whether they felt that engaging with student teams had any positive impact on them. They indicated that some of their personal benefits from such an experience include:

a) Seeing students get "real world" perspective and advice. All IT professionals enjoyed the interaction with their teams while receiving some gratification by watching them grow professionally and personally.

b) Contribute to institutional mission. This applies to the IT professionals who are employees of our University. Since they work for an educational institution, they felt obligated to not only provide technical support to the University but to contribute to its overall mission.

c) Seeing firsthand how teaching and learning happens. It became quickly apparent to us that the professionals who volunteered to participate in this project had a passion and curiosity for teaching. Evidently, this was an enjoyable experience for them.

d) Opportunity to practice mentoring and leadership. All professionals told us that they really valued the opportunity to mentor the student teams. They all felt that this was a rewarding experience while watching students absorbing their feedback and advice.

e) Opportunity to revisit fundamental concepts. They felt that it was beneficial to them to refresh their understanding of some basic CS and SE concepts. They also indicated that it helped them build a better perspective when they had to explain such concepts to others.

f) Practice using agile techniques. Professionals found some of the newer technologies introduced in our class to be valuable to them. Some even indicated that they were able to bring some ideas back to their office to improve their own processes.

g) Seeing multiple teams concurrently. The IT professionals explained to us how they benefited from having to manage a student team in addition to their own workplace team.

h) Team dynamics. Most professionals expressed their gratification while watching their team members learning and working together. They enjoyed guiding them to select and undertake specific roles in their team. They also kept reminding them the importance of having team building experience in the real workplace. Moreover, professionals were intrigued by seeing different approaches to common problems.

D. Stakeholders' Challenges

We asked all stakeholders to share some of their challenges throughout this experience, which are briefly described next.

a) Clarifying their role early. They told us that we had to ensure from the very beginning that IT professionals know their role and understand the expectations in the class and project. This is something we didn't do successfully since we didn't know ourselves what to expect. Their roles evolved naturally during the course and we were addressing and clarifying related issues as they appeared.

b) Difference of experiences. Their familiarity with some of the software tools and technologies used in class was different. They stressed that such tools may work differently in industry (e.g., using object relation mappings instead of writing database handlers).

c) Time commitment. They indicated that their biggest challenge was time commitment. This was something we expected from the beginning. However, it turned out that their engagement was consistent and their participation almost perfect.

E. Instructor's Challenges and Observations

Software engineering education becomes meaningful when we effectively relate it to real life. In order to achieve such a goal, educators need to be willing to take risks. The instructor of this course has taken such a risk by orchestrating the involvement of IT professionals who committed to play the role of a "stakeholder" and engage with students inside and outside the classroom.

In addition, the instructor kept a personal journal during the Scrum journey with observations and challenges faced by everyone involved. The rationale behind maintaining such journal is to continue improving the delivery of the course with an emphasis on identifying and assisting students who struggle with poor attendance or performance, incomplete or lower quality work etc.

In this section, we mention some of the instructor's challenges, observations and experiences that emerged while teaching such a course. They are listed below in the form of questions and are currently being deliberated in order to improve this course.

- *How do we educate and prepare IT professional stakeholders to effectively engage and interact with undergraduate software engineering students?*
- *How do we establish and secure a consistent and regular commitment from the IT professional stakeholders?*
- *How do we effectively manage and coordinate all IT professionals involved in Scrum related team meetings and ceremonies?*
- *How can we leverage regular feedback received from the IT professional stakeholders and at the same time making real-time changes based on such feedback during the course?*

- *How do we help students understand the role of their IT professional stakeholders and encourage and facilitate a meaningful interaction with them?*
- *How do we effectively balance the delivery of course content while at the same time allowing students to learn ahead on their own and from others?*
- *How do we establish a safe learning environment and a caring community where everyone involved feels respected and welcome to learn from each other?*
- *How can we best identify and assist those students who exhibit poor performance during the Scrum journey?*
- *How can we use Scrum as a pedagogical approach to teach other SE courses such as the one described in this paper?*

F. Summary of Data Analysis

Overall, we received useful and positive feedback from both student participants and IT professionals. Such feedback helped us improve the class while teaching it. More specifically, based on the overall responses that we received from our students we are convinced that the inclusion of IT professionals has a positive impact on a) their appreciation of Scrum and other related technologies taught, b) their attitude, work ethic and willingness to actively engage with their team, c) conducting themselves professionally in front of a real "stakeholder" and "product owner" and d) their understanding of what it means to work on realistic, larger and more complex software applications.

We gathered interesting and valuable feedback from the IT professionals with respect to their own benefits, gratifications and challenges while engaging with our Scrum teams. Overall, they felt that this was a rewarding experience and expressed a willingness to do it again. Their main challenge was, as expected, time commitment. We are now working with their employers to find a way to reward them in some way and justify their time when engaging with our students. In addition, the instructor of this course kept a journal of observations and challenges while teaching this course. These challenges are now being addressed and will be used in order to improve future offerings of this course.

We are aware of some threats to the validity of our study and its limitations. First, due to the size of our department, we have access to a small population of students. This was our primary reason for deciding to conduct an empirical study and collect qualitative data (rather than quantitative). We also tried to gather as much data as possible from three sections of the course taught over two consecutive years (i.e. 2018 and 2019). Second, the voluntary response rate to the study was low (i.e. less than 50%) despite our frequent reminders.

Finally, it is always difficult to find professionals who are able and willing to commit their time on an on-going basis, especially during week-day mornings for a long period of time (i.e. a whole semester) without compensation. We feel fortunate to have access to many of our alumni, who are IT professionals that are willing to engage with our students and reside near our campus. In addition, some of our University's IT personnel has expressed an interest in engaging regularly with our students both in and out of the classroom.

V. CONCLUSION AND FUTURE STEPS

Our overall experiences from engaging IT professionals in Scrum student teams are promising and beneficial to everyone involved. Students have indicated that working with such professionals in a classroom setting has helped them learn important software engineering principles, technologies and tools as well as become more mature professionally. Our role as educators is to prepare our students for their future careers, this approach has allowed us to do this to a satisfactory extent.

At the same time, IT professionals appear to enjoy such engagement and receive professional gratification from mentoring, coaching and guiding our students. It also allows for personal growth and reflection on their own skills. Therefore, we plan to continue fostering such an engagement in our software engineering courses while at the same time improving the course based on the feedback, we receive from everyone involved.

In addition, we are currently considering previous research and recommendations on how to conduct training workshops and seminars for IT professionals so that they can be better prepared to engage with our students and help them succeed. Finally, we are planning to leverage some of the instructor's observations regarding how to improve our overall course delivery with an emphasis on ensuring that all involved benefit and succeed.

APPENDIX A: Student Survey

Please rate the following using a scale from 1-5 (1 means strongly disagree and 5 strongly agree). There are some questions requiring brief answers.

Survey Section 1--Experience engaging with an IT professional

1. I feel comfortable engaging with my team's IT stakeholder
2. Our stakeholder is helping me better understand Scrum
3. Our stakeholder is helpful with the technologies used in the project
4. I consider our stakeholder to be an effective mentor and role model
5. I can communicate effectively with our stakeholder
6. I can understand clearly our stakeholder
7. I feel comfortable asking our stakeholder questions
8. I am satisfied with the answers provided by our stakeholder
9. Please describe your perception of the role of the stakeholder in your team (e.g. colleague, authority, teammate, etc.)
10. What suggestions do you have for the stakeholder to improve?

Survey Section 2--Teamwork

1. I am happy with my participation in our team's discussions
2. I can take initiative during this project
3. I can identify and resolve conflict during this experience
4. I feel comfortable asking for help from my team
5. I can communicate my own needs to the rest of the team
6. I can assist other team members
7. I am a good listener during our meetings and discussions
8. I am willing to share my own work with others
9. I can learn on-the-job when under pressure to deliver results
10. I am happy with my overall work ethic
11. I am effective with undertaking a task and completing it
12. My non-technical skills (e.g. teamwork, communication, etc.) have improved due to my participation and experience from this project
13. I understand that change is inevitable while developing software and I am willing to embrace it
14. I have a good understanding of working with a real client
15. I understand what it means to be a professional software engineer
16. Overall, I feel that I am a better team player from this experience

Survey Section 3-- Learning OO design principles, related software development technologies and tools

1. I feel confident learning and using Scrum
2. I can create UML diagrams that depict easy-to-maintain OO designs
3. I am comfortable using a version control tool (e.g., GitHub)
4. Using a project management tool (e.g., Trello) is useful
5. I can use the C#.NET language to write realistic software
6. I understand how to create a layered software architecture

Survey Section 4: Perception of large, complex software applications

1. At the beginning of the course, what is your perception of a large and complex software application?
2. At the end of the course, what is your perception of a large and complex software application?

Survey Section 5—Background Information

1. What is your year of study?
2. What is your declared program or major?
3. How was this class taught overall?

REFERENCES

[1] A. Moreno, et al, "Balancing software engineering education and industrial needs" in Journal of Systems and Software, vol. 85, issue 7, July 2012, pp. 1067-1620.

[2] C. Chen, P.Chong, "Software engineering education: A study on conducting collaborative senior project development" in Journal of Systems and Software, vol. 84, issue 3, March 2011, pp. 479-491.

[3] H.J.C. Ellis, N.R. Mead, A.M. Moreno and S.B. Seidman, "Industry/University software engineering collaborations for the successful reeducation of non-software professionals" in Proceedings 16th Conference on Software Engineering Education and Training, 2003.

[4] L. Jaccheri, S. Morasca, "On the importance of dialogue with industry about software engineering education" in SSEE '06: Proceedings of the 2006 international workshop on Summit on software engineering education.

[5] N. Mead, "Industry/University Collaboration in Software Engineering Education: Refreshing and Retuning Our Strategies" in 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering.

[6] N. Mead, et al, "Industry/university collaborations: different perspectives heighten mutual opportunities" in Journal of Systems and Software Volume 49, Issues 2–3, 30 December 1999, Pages 155-162.

[7] M. Lutz, F. Naveda, J. Vallino, "Undergraduate Software Engineering: Addressing the Needs of Professional Software Development" in acmqueue, July 21, 2014 Volume 12, issue 6, queue.acm.org

[8] V. Garousi, K. Petersen, B. Ozkan, "Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review" in Information and Software Technology Journal vol. 79, Nov. 2016, pp 106-127.

[9] K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, et al., "Manifesto for agile software development," 2001.

[10] K. Schwaber and M. Beedle, "Agile software development with Scrum", vol. 1: Prentice Hall Upper Saddle River, 2002.

[11] V. Mahnič, "Scrum in software engineering courses: an outline of the literature," Global Journal of Engineering Education, vol. 17, pp. 77-83, 2015.

[12] V. Mahnič, "Teaching Scrum through team-project work: Students' perceptions and teacher's observations," International Journal of Engineering Education, vol. 26, p. 96, 2010.

[13] A. Ciupe, R. Ionescu, S. Meza, B. Orza, "Towards Agile Integration within Higher Education: A Systematic Assessment", Broad Research in Artificial Intelligence and Neuroscience, vol. 9, pp. 69-87, 2018.

[14] P. Kamthan, "Towards an Understanding of Collaborations in Agile Course Projects" in Overcoming Challenges in Software Engineering Education, Chapter 3, edited by Yu Liguo, IGI Global.

[15] C. Sroka, "The Importance of Agile Methodologies Within Student-led Industry Projects", June 13, 2017, Center for Digital Media. Accessed Feb. 19, 2020. https://thecdm.ca/news/the-importance-agile-methodologies-within-student-led-industry-projects

[16] Baham, C., "Teaching Tip: Implementing Scrum Wholesale in the Classroom.", Journal of Information Systems Education, 2019, 30(3), 141-159.

[17] J. Campbell, S. Kurkovsky, C. W. Liew, and A. Tafliovich, "Scrum and agile methods in software engineering courses," in Proceedings of the 47th ACM Technical Symposium on Computing Science Education, 2016, pp. 319-320.

[18] A. Bassi, "Scrum Sim---A Simulation Game to Learn the Scrum Agile Framework," Ph. D. Dissertation. Harrisburg University of Science and Technology, 2016.

[19] K. Lundqvist, A. Ahmed, D. Fridman and J. Bernard, "Interdisciplinary Agile Teaching," *2019 IEEE Frontiers in Education Conference (FIE)*, Covington, KY, USA, 2019, pp. 1-8.

[20] G. S. Matharu, A. Mishra, H. Singh, and P. Upadhyay, "Empirical study of agile software development methodologies: A comparative analysis," ACM SIGSOFT Software Engineering Notes, vol. 40, pp. 1-6, 2015.

[21] C. Seaman. "Qualitative methods in empirical studies of software engineering." IEEE Transactions on Software Engineering, Vol. 25, Issue 4, Jul/Aug 1999, pp 577-572.

[22] T. Dyba, R. Priklandnicki, K. Rokkko, C. Seaman, J. Sillito, "Qualitative Research in Software Engineering", Journal of Empirical Software Engineering, Vol. 16, pp. 425-429, 2011.

[23] S. Eastrbrook, J. Singer, A. Storye, D. Damian, "Selecting Empirical Methods for Software Engineering Research", chapter in "Guide to Advanced Empirical Software Engineering", pp. 285-311, Springer.

[24] X. Huang, He Zhang, X. Zhou, M. Babar, S. Yang, "Synthesizing qualitative research in software engineering: a critical review", In Proceedings of the 40th ICSE 2018, pp. 1207-1218.

[25] F. Shull, J. Singer, and D. Sjoberg. "Guide to Advanced Empirical Software Engineering", Springer-Verlag, 2008.

[26] D. Reed, C. Miller, and G. Braught. "Empirical investigation throughout the CS curriculum." ACM SIGCSE Bulletin, March 2000.

[27] D. J. Mala, "Object Oriented Analysis and Design Using UML", McGraw-Hill Education, 2013.

[28] J. Sutherland, "SCRUM: The Art of Doing Twice the Work in Half the Time", Business and Economics Management.

[29] G. Philip, "Fundamentals of C# Programming for Information Systems", Prospect Press, Second Edition.

[30] M. Seidl, "UML @ Classroom: An Introduction to Object-Oriented Modeling", Undergraduate Topics in Computer Science, 2015th Edition.

[31] B. Lin, A. Zagalsky, M.-A. Storey, and A. Serebrenik, "Why developers are slacking off: Understanding how software teams use slack," in Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion, 2016, pp. 333-336.

[32] R. Martin, "The Clean Coder", Prentice Hall.

[33] ACM Committee on Professional Ethics, "The Software Engineering Code of Ethics and Professional Standards", Accessed February 20, 2020, https://ethics.acm.org/code-of-ethics/software-engineering-code/

[34] A. Joshi, S. Kale, S. Chandel, D. K. Pal, "Likert Scale: Explored and Explained", Current Journal of Applied Science and Technology, 7(4), pp. 396-403, January 2015.

[35] F. Fagerholm, A. Vihavainen, "Peer assessment in experiential learning: Assessing tacit and explicit skills in agile software engineering capstone projects," 2013 IEEE Frontiers in Education Conference (FIE), Oklahoma City, OK, 2013, pp. 1723-1729.