Experiential Learning of Software Project Management and Software Development via Course Collaboration

Stefan C. Christov
Quinnipiac University
Hamden, CT
stefan.christov@quinnipiac.edu

Mark E. Hoffman
Quinnipiac University
Hamden, CT
mark.hoffman@quinnipiac.edu

ABSTRACT
The skills to effectively manage software development teams and to productively interact with a project manager are important in the computing professions. Teaching these skills in a traditional academic setting, however, is challenging. To support the experiential learning of these skills, we established a collaboration between two computing courses, a sophomore-level one and a senior-level one, where the seniors serve as managers of teams of sophomores on a semester-long project. This paper describes the collaboration and evaluates it in terms of student learning and experience based on three iterations of that collaboration. The associated course materials and evaluation instruments are made publicly available.

CCS CONCEPTS
• Social and professional topics → Computing education;

KEYWORDS
Software project management, collaborative teaching, experiential learning, software development

1 INTRODUCTION
Software project management (SPM) is recognized as an important skill in software engineering and in project management in general. Software engineering management, including subareas such as project initiation and scope definition, project planning and estimation, and project measurement and control, is one of the knowledge areas in the Software Engineering Body of Knowledge [11]. Also, recently, the Software Extension to the Project Management Body of Knowledge [12] was approved by both the Project Management Institute [2] and the IEEE Computer Society [1].

Teaching SPM skills in a traditional academic setting, however, is challenging primarily due to the difficulty of providing students with opportunities to apply the SPM theory they learn. To truly practice SPM skills, a student needs to manage a relatively long-term, non-trivial software project executed by a team of software developers. Running such a project, finding such teams, and empowering students to be in a managerial position is logistically difficult. Thus, SPM is often taught via imaginary or non-concurrent software projects where the course instructor provides project materials, such as requirements or design documents, and asks students to apply SPM knowledge to generate another set of documents, such as effort/size estimates or a project plan. Such projects allow students to practice some aspects of SPM, but they preclude the opportunity to exercise other important aspects. These other aspects include the enactment of a project plan, and especially the typically iterative, incremental nature of planning and estimation based on what is learned during an ongoing project, as well as the human aspect of SPM, including intensive communication with developers and potentially dealing with intricate personality issues.

To address the above challenges, we established a collaboration between a sophomore-level Introduction to Software Development (ISD) course and a senior-level SPM course in which the seniors serve as managers of teams of sophomores on a semester-long project. This collaboration allows the seniors to exercise the variety of SPM skills discussed above and the sophomores to experience interacting with a manager. We also found that the collaboration supports other non-SPM related skills in the ISD course.

The main contributions of our work are: a pedagogical framework for conducting an undergraduate SPM-related course collaboration; an evaluation methodology for assessing and continuously improving such collaboration; and results on student learning and experience based on a three-year study as well as lessons learned. We also make publicly available all course materials associated with the collaboration and all evaluation instruments.

The next section of this paper discusses related work. Section 3 describes our approach to supporting experiential learning of SPM and Section 4 presents how we evaluated that approach. Evaluation results are discussed in Section 5 and Section 6 concludes the paper.

2 RELATED WORK
The importance of providing students with opportunities to apply the theoretical knowledge obtained in computer science (CS) and software engineering (SE) courses has been widely recognized (e.g., [9, 16]). A survey of the SE education literature in the period 1982-2013 indicates that the most widely used approach to providing practical experiences in SE education is “learn by doing” [20].

To support “learn by doing” in the context of SPM, course collaborations have been previously used, although surprisingly few are reported in the literature. Bavota et al. [4] as well as Maqsood and
Javed [19] report collaborations between graduate SPM courses and other undergraduate courses where graduate students manage undergraduate students. The effectiveness of these collaborations is indirectly evaluated—in [4], by inferring learning success in terms of size and quality of the software systems produced and in [19] by comparing the structure of the graduate course to a capability maturity model [7]. In [4], a survey was also administered several years after the course completion and a positive general experience is reported. Detail about specific skills or knowledge that the course collaborations promoted is not provided.

Johns-Boast and Flint [13–15] describe a course collaboration at the undergraduate level where students in the senior part of a software development capstone course manage student teams in the junior part of that course. Based on feedback from different stakeholders (e.g., students and industrial partners), general benefits of this collaboration are noted. Few specifics are provided on SPM skills and knowledge obtained by students.

Besides through course collaborations, educators have supported practical SPM experience within a single course. For example, Bollin et al. use a simulation tool to teach SPM [5]. Malachowsky reports on observations related to introducing a project manager role in projects for an undergraduate Process and Project Management course [18]. Kuhrmann and Münch describe a teaching unit to support learning of group dynamics issues and strategies [17].

Other course collaborations not particularly focusing on SPM have been reported. These include studies of the feasibility of distributing large SE projects across the academic curriculum [8]; students from different courses collaborating by taking different roles in a simulated software factory [21]; teaching large scale teamwork in a small college environment by using multi-semester, multi-course projects [23]; collaboration between a graduate Education course and an undergraduate SE course [3]; and a collaboration between two graduate Distributed Software Development courses at two different universities [22].

Unlike [4] and [19], the work described in the current paper evaluates the collaboration between two undergraduate courses. Compared to [4], [13–15], and [19], we evaluate the collaboration in terms of specific skills and knowledge that the collaboration promotes for students in both courses. We also make publicly available all course material supporting the collaboration as well as the instruments used for evaluation and continuous course improvement. While all the single-course approaches described in [5, 17, 18] are useful for teaching certain SPM skills, we believe that the course collaboration approach presented in the current paper allows students to practice a richer set of SPM skills over a longer period of time. Our work is similar to the work described in [3, 8, 21–23] in terms of identifying general course collaboration issues and strategies to address them, but our work adds lessons learned specific to issues with and strategies for supporting learning of SPM.

3 APPROACH

This section describes the two courses that facilitate the reported collaboration, the structure of the most recent collaboration, and our continuous improvement process.

3.1 The ISD and SPM Courses

Introduction to Software Development (ISD) is a first semester, sophomore-level course required of CS and SE majors, and CS minors. Prior to taking the ISD course, CS and SE majors take an introductory programming course (CS1), a data structures course (CS2), and an object-oriented programming course (OOP). CS minors are not required to take the OOP course. Students work in teams of 4 or 5 on a semester-long software maintenance project that serves as a platform to exercise various software development practices.

Software Project Management (SPM) is a required course for SE majors, typically taken first semester of senior year. The course acquaints students with various aspects of SPM. Students are familiar with the ISD course having already taken it.

The topics covered and the deliverables produced each week in both courses are listed in Table 1. Detailed description of the ISD and SPM courses can be found in [10] and [6], respectively. Course materials are available online at https://tinyurl.com/y7kwdhzw.

3.2 ISD and SPM Course Collaboration

Beginning in 2015 (the first time the SPM course was offered), students in the SPM course were assigned to manage ISD student teams. The collaboration has evolved through a continuous improvement process described in Section 3.3. This section describes the most recent collaboration (2017).

Each SPM student (manager) is assigned to manage one ISD student team. Managers meet with their team weekly during one of the scheduled ISD class periods. At these meetings, managers perform a number of management tasks coordinated with the assignments in the ISD and SPM courses. The fourth column in Table 1 details the interaction between the ISD teams and their managers.

Managers work with their team to create a work plan at the beginning of each Scrum sprint, finalize all work unit assignments, and submit the work plan to the ISD instructor. Managers run the weekly team meetings by providing an agenda and submit minutes to the SPM instructor. Managers receive periodic individual team member status reports that they use to create individual and team performance evaluations. The managers’ evaluations are factored in grades of the ISD students. Managers facilitate individual and team postmortems at the end of each Scrum sprint.

Managers also provide leadership and guidance to their teams. Managers sometimes need to motivate underperforming teams, provide moral support, and facilitate intra-team conflict. Managers do not code for their team, but are allowed to suggest strategies to facilitate coding tasks. Managers are free to share their experience as ISD students to facilitate their teams’ success.

3.3 Continuous Improvement Process

Over the three years of this collaboration, we have used a continuous improvement process to identify issues and propose course changes to address them. During the fall, while the courses are in session, we gather data using the methods described in Section 4.2. After the courses end, during the winter and spring, we analyze the data to determine to what degree the course collaboration supports the courses’ learning outcomes. Based on the analysis results, we propose changes for the collaboration and modify course materials based on these changes during the summer, prior to the next offering of the courses. The continues improvement process repeats, starting with the next offering of the courses.
4 EVALUATION METHODOLOGY

4.1 Participants

Participants for this study consisted of students enrolled in a sophomore-level ISD course and students enrolled in a senior-level SPM course (Table 2). In addition to CS and SE, several students pursued double majors while others minored in CS. The ISD course had two sections each year and students were randomly assigned to teams of 4 or 5 students within the same section. SPM students were randomly assigned to manage teams. The first year of the study, 2 SPM students managed 2 teams each; four teams did not have a manager. In 2016, 9 SPM students managed 7 teams; two teams had 2 managers. In 2017, 7 SPM students managed 7 of the 8 teams; one team did not have a manager.

4.2 Data Collection

Data for this study was collected using methods developed for the 2015 collaboration [6]. The methods include end-of-semester surveys, course reflections, and instructor observations throughout each semester. Based on the continuous improvement process described in Section 3.3, methods were revised in 2016 and 2017. Each method is described in detail below and the supporting data collection instruments are available online at https://tinyurl.com/y7kwzdzw. To reduce bias during data analysis, collected data were independently analyzed by two reviewers and discrepancies were resolved by reaching consensus.
5 RESULTS AND DISCUSSION

5.1 Educational benefits of the collaboration

Figure 1 shows the self-assessed achievement of the CLOs of the ISD and SPM courses reported in the end-of-semester surveys. Based on these results, the course collaboration is contributing to students’ achievement of most of the CLOs. For the SPM course, the achievement of most CLOs was rated around 4.5 on average, which means that students agreed or strongly agreed that the collaboration contributed to their achievement of these CLOs. For the ISD course, most of the CLOs were rated around 4, the “Agree” level. It is not surprising that the collaboration has a slightly stronger contribution to the achievement of CLOs in the SPM course than in the ISD course because serving as a manager of a real team exercises a large set of project management skills. Working under a manager allows ISD students to exercise important software development skills, such as working on a team and evaluating and maintaining a software product, but other skills in the ISD course, such as explaining social and ethical implications of software development were addressed by other components of the ISD course.

The self-assessed level of achievement of CLOs increased from the first course collaboration in 2015 to the third course collaboration in 2017. We believe this is a result of the continuous improvement process described in Section 3.3. Note that some of the CLOs received a lower score in 2016 compared to 2015 (all CLOs received higher score in 2017 compared to 2015 and 2016). The dip in 2016 is most likely due to making significant modifications to the course collaboration in 2016 based on limited experience (the collaboration in 2015 that involved only two managers). There were also numerous scheduling conflicts in 2016, making the interaction between managers and their teams difficult. The issues encountered in 2016 were carefully analyzed and addressed in the 2017 collaboration (see discussion in Section 5.2).

The end-of-semester surveys and reflections from the SPM course indicated that the course collaboration contributed to the SPM students’ learning and practicing of soft skills in SPM, such as leading a meeting, handling difficult team members, and conducting performance evaluations. This is a strong advantage of an SPM course that allows students to manage a software development team compared to traditional SPM courses that lack such a component.

The ISD students reported multiple benefits of having a manager. For example, the managers provided useful advice during the course project, coordinated and organized their teams, set expectations and reviewed project artifacts. Some managers also provided moral support and even taught their teams some software development technologies, such as Git. It is interesting to note that the number of unsolicited comments about the benefit of having a manager in the ISD end-of-semester reflections substantially increased in 2017. The manager-related comments in 2017 were also more specific compared to similar comments from previous years. We believe this is an indication of the ISD students’ increased involvement with their manager and better understanding of the manager’s role.

5.2 Structuring the collaboration

Establishing the collaboration between the ISD and SPM courses posed several challenges. One of the most important challenges was providing sufficient time for interaction between teams and
their managers. In 2015 and 2016, we asked the managers and their teams to find a time period outside of class for a weekly meeting. This proved to be difficult due to scheduling conflicts and due to some students’ time management problems. To address this issue, in 2017 we added a lab to the SPM course that consisted of the SPM students attending one class period of the ISD course per week to meet with their teams. This lab does not increase the credit hours of the SPM course as it is considered part of the out-of-class time students need to spend working on the course. In 2017, multiple students commented in the end-of-semester surveys and reflections that these meetings significantly facilitated the course collaboration. Students also indicated that using a group chat service to communicate outside of in-person meetings was useful.

Determining the right amount of authority for the managers is an important challenge. The managers’ authority during the first course collaboration in 2015 primarily consisted of requiring the ISD teams to consult their managers regarding various project issues at least once a week and to submit some project artifacts to the manager prior to submitting them to the course instructor. There was no mechanism for a manager to affect the grade of the managed team, as we were reluctant to have students influence the grade of other students, especially with no prior experience on this collaboration. The amount of manager authority in 2015, however, was determined to be insufficient by both ISD and SPM students as the SPM students acted more as consultants than as managers. As a result, we gradually increased the manager’s authority in 2016 and 2017 to the level described in Section 3.2.

This gradual increase of manager authority seems to have had a positive effect. In 2017, the majority of the ISD and SPM students considered the amount of manager authority “just about right” (Figure 2). The course instructors also observed that teams tended to follow their manager’s guidance and complete tasks assigned by the managers by the deadlines set by the managers. Figure 2 suggests that with the gradual increase of manager authority from 2016 to 2017, a larger percentage of the ISD students (about 64%) considered the amount of manager authority appropriate rather than low in 2017 compared to 2016 (about 48%), with still about a third of the ISD students in 2017 desiring more manager authority. This result is consistent with the instructors’ observation that ISD students tend to appreciate higher manager authority which seems to correlate with more support from the manager in various aspects of the software development project, including resolving team conflicts and dealing with less engaged/underperforming team members.

On the other hand, Figure 2 suggests that the gradual increase of manager authority from 2016 to 2017 was associated with smaller, but still substantial percentage of SPM students considering the amount of manager authority appropriate in 2017 (about 57%) than in 2016 (about 89%). Interestingly, however, the SPM students who did not consider the amount of manager authority appropriate in 2017, fall on both sides of “Just about right”—namely “High” and “Low”. This might be due to varying personal managing styles (e.g., hands-off vs. hands-on) and/or not clearly communicating the same expectations to ISD and SPM students.

Besides providing managers with appropriate amount of authority, setting up the course collaboration such that the manager is appropriately involved with his/her team is challenging. Manager involvement was incrementally increased over the three course offerings. The activities not directly related to manager authority, but promoting manager involvement are the weekly meetings with the managed team, attending the team presentation at the end of each Scrum sprint, and participating in team postmortems. An appropriate amount of manager involvement seems to have been achieved in Fall 2017 based on the survey results—79% of the ISD students and 100% of the SPM students rated the amount of manager involvement as “Just about right.”

Achieving appropriate amount of manager authority and involvement might have come at the expense of reaching or slightly exceeding an appropriate workload limit in the SPM course. Whereas...
in 2016, 100% of the respondents of the SPM end-of-semester survey indicated that the workload was "just about right", only about "57%" gave this answer in 2017 and 43% considered the workload high or too high. Based on our analysis of the 2017 data, we believe we could reduce the workload of SPM students without hurting the collaboration. One possibility is to reduce the amount of paperwork, such as status reports ISD students submit to SPM students to review. Student feedback suggests that these reports duplicate the weekly meetings between managers and teams.

Overall, the structure of the course collaboration has stabilized. Our analysis of the 2017 data suggests fewer potential changes compared to previous course offerings. In the end-of-semester reflections, the focus of student writing has shifted from complaints and suggestions for course changes to reporting of learning diverse skills and discussing the benefits of the course collaboration.

An open issue that remains to be addressed is dealing with “underperforming managers.” As in most courses, the SPM course can have a range of students in terms of academic strength and commitment. Thus, there are sometimes students who end up being underperforming managers. This could be particularly problematic in the context of the course collaboration as not only the grade of that underperforming SPM student is affected, but also the learning experience and potentially the grade of the ISD students in this manager’s team. Some of the strategies we are considering to deal with this issue include compiling a collection of supporting material for underperforming managers (such as tips and best practices based on our experience so far) and soliciting manager evaluations from the managed teams to identify early signs of underperformance that would allow instructors to intervene early in the semester. We are also careful in designing the grading scheme in the ISD course in a way that the ISD students’ grade is minimally affected by underperforming managers.

5.3 Workload of course instructors

The course collaboration did not substantially increase the workload of the instructors of the ISD and SPM courses during the semester and no teaching assistants (TAs) were used. In addition to the regular duties associated with the ISD course, the ISD instructor attended the SPM class for 30 minutes each week to discuss with the managers the weekly progress of the teams and any issues that might have arisen. The total time commitment was about 7 hours. The SPM instructor attended the first and the last class of each of the two sections of the ISD course to introduce the course collaboration and to provide feedback on final presentations, respectively. The total time commitment was about 3.5 hours. In addition, the ISD and SPM instructors met with each other for about 45 minutes two times during the semester to coordinate the course collaboration.

Most of the instructor time on the course collaboration was spent between semesters (in the summer) to analyze data from the previous course offering and continuously improve the collaboration as discussed in Section 3.3. The instructors met in person 3-4 times on average for approximately 1.5 hours each summer. In addition, each instructor individually analyzed data and made course modifications for about 10-15 hours total.

We believe that the educational benefits associated with the course collaboration described in Section 5.1 substantially outweigh the instructor workload associated with supporting this collaboration. The additional workload associated with supporting this collaboration. The additional workload during the semester is minimal and amortized over 16 weeks; the workload for course improvements between semesters is both essential for careful evidence-based course development and has been decreasing as the course collaboration has stabilized.

5.4 Limitations

The course collaboration was implemented and evaluated in a teaching-focused institution with small class sizes (at most 24 students) and no TAs. Scaling such a collaboration to a different setting, e.g., an institution with large class sizes and heavy reliance on TAs, will most likely be associated with different obstacles, such as coordinating a large number of teams and TAs, and might lead to different results. The detailed coordination structure outlined in Table 1, the complete set of supporting course material we provide, and the continuous improvement methodology should significantly facilitate the replication of our experience.

We have a curriculum structure (the ISD course, the SPM course, and the weekly lab in the SPM course) that affords the opportunity to implement this type of collaboration. Other institutions might not have such a structure. Most computing programs, however, offer a course on software engineering/development and such a course could be combined with a concurrent SPM-related elective or a graduate course to implement a similar collaboration. In addition to establishing a curriculum structure, we are working on additional mechanisms (e.g., further synchronizing assignments and course materials) to institutionalize the course collaboration, so that it can continue when other instructors teach these courses.

We evaluated the course collaboration on a relatively small number of course offerings, three, but the course structure and the educational observations seem to be stabilizing, which supports the validity of the presented results. Much of the data we analyzed is self-reported by students. This data is consistent, however, with the observations instructors made over the three course offerings. We are also considering approaches for performing further quantitative evaluation, such as using the grades of course assignments and mapping these assignments to learning outcomes.

6 CONCLUSION AND FUTURE WORK

The skills to effectively manage software development teams and to productively interact with a project manager are important in the computing professions, but providing opportunities to exercise these skills in an academic setting is challenging. To support the experiential learning of these skills, we established a collaboration between two computing courses. Our results indicate that this collaboration has been successful in terms of supporting course learning outcomes and practicing soft-skill aspects of SPM, which are difficult to practice in SPM courses that do not involve managing a team. The course collaboration is also feasible in terms of course instructors’ workload. Important issues for establishing such collaboration and strategies to address them have been identified. To further facilitate the implementation of such collaboration by others, all course materials and evaluation instruments to support continuous improvement are made publicly available. In the future, we plan to explore approaches for handling underperforming managers and quantitatively evaluating the collaboration.
REFERENCES