Evaluating a Software Project Management Course Collaboration Framework at a Second Institution

Dr. Stefan Christov, Quinnipiac University

Stefan Christov holds Ph.D. and M.S. degrees in Computer Science from the University of Massachusetts Amherst and a B.S. degree in Computer Science from the State University of New York, College at Brockport. He has experience in teaching undergraduate introductory computer science and engineering courses as well as upper-level software engineering courses, including software quality assurance, software project management, and software engineering in health care. His current research interests include improving the quality of human-intensive processes (HIPs), such as medical processes, with a focus on detecting human errors before harm is done and preventing such errors. He has used software engineering techniques to formally represent and analyze models of complex HIPs and industrial engineering techniques to elicit and validate models of such processes. He is also interested in educational approaches for pedagogical collaboration between different courses in the curriculum. His work has resulted in publications in international journals and conferences.

Dr. James Walker, Michigan Technological University
Dr. Mark Hoffman, Quinnipiac University

Mark Hoffman is a professor of computer science at Quinnipiac University. He joined the University in 2001 following a career in industry and has taught a wide variety of courses including data structures, computer architecture and organization, software development, and the senior capstone project. His research interests include communication and critical thinking skills in computer science education, and the impact of technology on work/home boundary management. He received his Ph.D. from Polytechnic University in Brooklyn, NY.
Evaluating a Software Project Management Course Collaboration Framework at a Second Institution

Abstract

Previous work has described and evaluated a collaboration between a software project management (SPM) course and an introductory software development course. That collaboration has successfully supported the experiential learning of SPM skills as well as skills to effectively interact with a project manager. This paper further evaluates that collaboration framework by reporting its application at a second institution for two semesters. Overall, the learning outcomes and student experiences achieved at the second institution were similarly positive to those achieved at the first institution. The major components of the framework were fully implemented at the second institution with specific aspects customized to meet the institution’s educational outcomes and setting. In that customization, three factors were identified for consideration when adapting the collaboration framework: the existence of a course structure to support such collaboration, the ratio between the students in the two courses, and the ability to schedule regular interactions between these students.

1 Introduction

Software project management (SPM) is a key knowledge area in the Software Engineering Body of Knowledge [1] and for project managers, in general, in the Software Extension to the Project Management Body of Knowledge [2]. However, teaching key SPM skills such as project initiation, scope definition, planning, estimation, measurement, and control is challenging because practicing them requires management of non-trivial, relatively long-term software projects developed by a team. Without such a “hands-on” environment, SPM skills are typically practiced using a hypothetical project where students produce project management documents, such as project plans. While this approach does allow for practice with the principles of some skills, it does not afford the opportunity to apply those skills in environments where human factors, such as the motivation, the ability, and the background of individual team members, vary. To provide Software Engineering (SE) majors at Quinnipiac University (QU) an opportunity to apply and develop various SPM skills, in previous work we developed a course collaboration framework where students in an SPM course serve as managers of student teams maintaining a software system in an introductory software development course [3–5]. Results reported after the third year of the collaboration [5] show that students are engaged and report learning and applying key skills.
The main contribution of the current paper is the further evaluation of the course collaboration framework developed at QU by applying it for two years at a different institution—Michigan Technological University (MTU). The collaboration at QU was entering its fifth year when the number and significance of improvements had diminished leaving a well-defined framework for adoption at MTU. The transfer of that framework to MTU raised the following research questions:

R1: To what degree are similar student learning experiences achieved at a second institution (MTU)?
R2: To what degree can the collaboration at QU be replicated at another institution (MTU)?
R3: What factors at the target institution (MTU) need to be considered in the adoption process?

To answer these questions, we followed the evaluation methodology used to evaluate the collaboration at QU with minor modifications to accommodate differences at MTU. That methodology consisted of surveying students in both courses of the collaboration and analyzing instructor observations made throughout the semester. We have found that the collaboration developed at QU was successfully adopted at MTU and realized similar results during the second year—students achieved similar learning experiences at MTU as achieved at QU. We also identified key factors that needed to be addressed at MTU to successfully adapt the collaboration. We report observations on the resiliency of the course collaboration in a remote learning setting necessitated by the COVID-19 pandemic.

The next section of the paper discusses related work. Section 2 compares the institutional contexts at QU and MTU and factors that needed to be considered to adapt the collaboration at MTU. The evaluation methodology is described in Section 3. Section 4 summarizes the results which are then analyzed in Section 5. Section 7 concludes the paper and discusses future work.

2 Related Work

The benefits and importance of experiential learning in the technology disciplines in general (e.g., [6]) and in computer science (CS) and software engineering (SE) in particular (e.g., [7, 8]) are extensively described. A survey of the SE education literature in the period 1982-2013 indicates that the most common approach to providing practical experiences in SE education is “learn by doing” [9]. Collaborative learning, which emphasizes the importance of students to learn together and from each other, has also been reported to enhance student learning (e.g., [10]).

To support “learn by doing” in a collaborative setting in the context of SPM, course collaborations have been previously used, although surprisingly few are reported in the literature. Bavota et al. [11] as well as Maqsood and Javed [12] report collaborations between graduate SPM courses and other undergraduate courses where graduate students manage undergraduate students. The effectiveness of these collaborations is indirectly evaluated by inferring learning success in terms of size and quality of the software systems produced [11] and by comparing the structure of the graduate course to a capability maturity model [12, 13]. In [11], 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 [14–16] 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 by using course collaborations, educators have supported practical SPM experience within a single course. For example, Bollin et al. use a simulation tool to teach SPM [17]. 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 [19]. For instance, Bollin et al. report that while the simulation tool approach supported SPM experience with waterfall-like processes, the support for agile processes, which are widely used nowadays, was limited. Also, using computer simulation often makes it difficult to exercise certain SPM skills, particularly soft skills and human aspects of managing a team. The approaches reported in [18] and [19] focus on practicing SPM skills but outside of a long-term software development project ([18] focuses on producing a paper and a presentation; [19] focuses on a game of sorting candy).

Other course collaborations not necessarily related to SPM have been reported. These include studies of the feasibility of distributing large SE projects across the academic curriculum [20]; 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 [22]; collaboration between a graduate Education course and an undergraduate SE course [23]; a collaboration between two graduate Distributed Software Development courses at two different universities [24]; and a collaboration between three undergraduate courses in the areas of marketing, computer science, and product design [25].

Unlike [11] and [12], the work described in the current paper evaluates the collaboration between two undergraduate courses. Compared to [11], [12], and [14–16], 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 [17–19] 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 [20–24] 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. Unlike most of the above work, our work evaluates a proposed pedagogical approach in more than one institutional context.

3 Institutional Contexts and Adaptation Process

3.1 Courses and collaboration at QU

Computer Science (CS) BA and BS majors as well as Software Engineering (SE) majors are required to take a course on introduction to team-based software development. In the rest of the
paper, we refer to this course as Team Software Project (TSP). CS and SE majors typically take the course in fall of their second year after completing three programming courses during their first year: introductory programming (CS1), data structures (CS2), and object-oriented programming (OO). In the beginning of the semester, students are randomly assigned to teams of 4 or 5 students to work on a semester-long project to maintain and enhance a legacy software system. The first five weeks are spent on exploring the legacy system, identifying areas for improvement in consultation with a sample of potential users, learning about the Scrum development methodology and about version control. The next eight weeks are dedicated to four two-week Scrum sprints in which the teams fix bugs and make enhancements to the legacy software system. The last week is dedicated to packaging the resulting software system and preparing a report and a final presentation. (TSP course details are in the middle column of Table 1. A full description of the course may be found in [4].)

Software Project Management (SPM) is a required course for SE majors. It is typically taken during the first semester of senior year. The course introduces students to a variety of typical SPM skills including project planning, project estimation, risk management, as well as human aspects such as teamwork and performance evaluations. Students obtain a “hands-on” experience with these and other SPM skills through a collaboration with the TSP course. (SPM course details are in the middle column of Table 2.)

The collaboration began in 2015 with the first offering of the SPM course as part of a new SE program [3]. SPM students are assigned to manage TSP student teams. In addition to the SPM course meetings, managers attend one TSP course meeting per week to meet with their teams (scheduled as a lab meeting time for SPM students.) Managers perform tasks consistent with topics in the SPM course, such as project planning, estimation, risk management, developing teamwork skills, leading a meeting, and individual and team performance management. At the beginning of each Scrum sprint, managers create a work plan with work units assigned to each team member. During each sprint, managers monitor individual and team performance. At the end of each sprint, managers attend their team’s product demonstration and facilitate their team’s retrospective. In addition to structured tasks, managers provide leadership and guidance for their teams. Since managers have taken the TSP course, they frequently share their experience and strategies for team success. (Course collaboration details are in the middle column of Table 3. A full description of the collaboration may be found in [5].)

3.2 Corresponding courses at MTU

The Computer Science and Software Engineering programs at MTU have equivalent courses to QU’s TSP and SPM. TSP is required for CS and SE majors, and SPM is required for SE majors. A significant number of non-majors take both courses as well. TSP is a junior-level course which majors take after having completed introductory programming courses and data structures, while SPM is a senior-level course usually taken by undergraduate majors in their last year, and by graduate students.

In TSP, during the first five weeks students learn fundamentals of software development in a team, including UML documentation, revision control, and automated testing. Students are taught the basics of the Scrum process and instructed to use this process when developing their projects.
### Weeks 1-5

**Program Review (Week 1):** Install legacy software system and evaluate its operation

**Customer Requirements (Week 2):** Gather from potential users

**Prioritized Bug and Enhancement List (Week 3):** Develop a vision statement for the software system and prioritize bugs and enhancements from Program Review and Customer Requirements

**Scrum Sprint 0 (Weeks 4 & 5):** Create git repository, create user stories for items on the Prioritized Bug and Enhancement List, and create test cases for user stories

### Weeks 6-13

**Scrum Sprints 1-4 (4 two-week Scrum Sprints):**

- Create work plan with work units assigned to team members
- Perform work units
- Submit status reports
- Submit final work plan and demo project
- Conduct and report a team postmortem

### Weeks 14-15

**Final Report, Presentation/Demo, and Project Packaging:**

- Package project code and documentation for next team
- Submit a final report and make a final presentation/demo to a general audience

### Team Software Project (MTU)

- Software Processes – An overview of software processes, the software development life cycle, and agile.
- Revision Control – General overview and how to use Git and GitHub
- Testing – Automated testing, unit testing, regression testing.
- Scrum – Details of the Scrum process.
- Documentation – Types of documentation with a focus on UML and design documentation.
- Development Environments – Building large software projects and handling dependencies.

### Table 1: Team Software Project Course Comparison during Fall 2020.

<table>
<thead>
<tr>
<th>Weeks</th>
<th>Team Software Project (QU)</th>
<th>Team Software Project (MTU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-5</td>
<td>Program Review (Week 1): Install legacy software system and evaluate its operation</td>
<td>Software Processes – An overview of software processes, the software development life cycle, and agile.</td>
</tr>
<tr>
<td></td>
<td>Customer Requirements (Week 2): Gather from potential users</td>
<td>Revision Control – General overview and how to use Git and GitHub</td>
</tr>
<tr>
<td></td>
<td>Scrum Sprint 0 (Weeks 4 &amp; 5): Create git repository, create user stories for items on the Prioritized Bug and Enhancement List, and create test cases for user stories</td>
<td>Testing – Automated testing, unit testing, regression testing.</td>
</tr>
<tr>
<td>6-13</td>
<td>Scrum Sprints 1-4 (4 two-week Scrum Sprints)</td>
<td>Scrum Sprints 1-4 (4 two-week Scrum Sprints)</td>
</tr>
<tr>
<td></td>
<td>Each Scrum Sprint includes the following tasks:</td>
<td>Each Scrum Sprint includes the following tasks:</td>
</tr>
<tr>
<td></td>
<td>• Create work plan with work units assigned to team members</td>
<td>• Assign work units (user stories) to team members</td>
</tr>
<tr>
<td></td>
<td>• Perform work units</td>
<td>• Perform work units</td>
</tr>
<tr>
<td></td>
<td>• Submit status reports</td>
<td>• Conduct status reports and peer reviews</td>
</tr>
<tr>
<td></td>
<td>• Submit final work plan and demo project</td>
<td>• Demo project</td>
</tr>
<tr>
<td></td>
<td>• Conduct and report a team postmortem</td>
<td>• Conduct a team postmortem</td>
</tr>
<tr>
<td></td>
<td>• Package project code and documentation for next team</td>
<td>• Give a comprehensive presentation describing project development over the semester and demo the project in its final state.</td>
</tr>
<tr>
<td></td>
<td>• Submit a final report and make a final presentation/demo to a general audience</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Team Software Project Course Comparison during Fall 2020.
<table>
<thead>
<tr>
<th>Weeks</th>
<th>Software Project Management (QU)</th>
<th>Software Project Management (MTU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-5</td>
<td>Intro to SPM, Human Aspects of SPM I (Week 1)</td>
<td>Software Processes Review, Project Planning I (Week 1)</td>
</tr>
<tr>
<td></td>
<td>Human Aspects of SPM II, Software Project Foundations (Week 2)</td>
<td>Project Planning II (Week 2)</td>
</tr>
<tr>
<td></td>
<td>Software Project Estimation (Week 3)</td>
<td>Out-of-Class meeting minutes (Week 2)</td>
</tr>
<tr>
<td></td>
<td>Project Plans, Individual Team Member Evaluation (Week 4)</td>
<td>Scheduling Techniques (Week 3)</td>
</tr>
<tr>
<td></td>
<td>Project Planning Techniques (Week 5)</td>
<td>Software Project Estimation (Week 4)</td>
</tr>
<tr>
<td></td>
<td>In-Class team meeting minutes (each week)</td>
<td>Risk Management, Review of Scrum Process (Week 5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Final Project Plan and Scrum Sprint 1 Work Plan (Week 5)</td>
</tr>
<tr>
<td>First</td>
<td>Version Control Tools (Week 6)</td>
<td>The Development Pipeline (Week 6)</td>
</tr>
<tr>
<td>Week of</td>
<td>Measuring and Controlling Work Products (Week 8)</td>
<td>Midterm Exam Review (Week 8)</td>
</tr>
<tr>
<td>Scrum Sprints</td>
<td>Performance Evaluation, Managing Risk I (Week 10)</td>
<td>Measuring Progress (Week 10)</td>
</tr>
<tr>
<td>(6, 8, 10, 12)</td>
<td>Managing Risk II (Week 12)</td>
<td>Survey of Agile Software Processes (Week 12)</td>
</tr>
<tr>
<td></td>
<td>In-Class team meeting minutes, Scrum sprint work plan (each week)</td>
<td>Mid-Sprint report, including meeting minutes (each week)</td>
</tr>
<tr>
<td>Second</td>
<td>Software Project Estimation II (Week 7)</td>
<td>Secure Software Development (Week 7)</td>
</tr>
<tr>
<td>Week of</td>
<td>Midterm Exam (Week 9)</td>
<td>Requirements Elicitation Techniques (Week 9)</td>
</tr>
<tr>
<td>Scrum Sprints</td>
<td>Managing and Controlling Work Processes (Week 11)</td>
<td>Human Aspects of Management (Week 11)</td>
</tr>
<tr>
<td>(7, 9, 11, 13)</td>
<td>Ethics in SPM (Week 13)</td>
<td>Service-Oriented Software (Week 13)</td>
</tr>
<tr>
<td></td>
<td>In-Class team meeting minutes, individual team member evaluation rubric (each week)</td>
<td>End-Sprint report, including meeting minutes (each week)</td>
</tr>
<tr>
<td>14</td>
<td>Organizational Issues in SPM</td>
<td>Final Exam Review</td>
</tr>
<tr>
<td></td>
<td>In-Class team meeting minutes, performance evaluations of team members, course reflection paper</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Software Project Management Course Comparison during Fall 2020
Students are then randomly assigned to teams of 4-5 members and propose software development projects to work on, subject to the instructor’s approval. The next eight weeks are spent working on these projects, divided into four two-week sprints, giving demonstrations and submitting reports at the end of each sprint. Projects are required to use revision control and include a robust testing plan. During the final week, each team gives a 30-minute demonstration and presentation of their project and the course of its development for the preceding eight weeks. (TSP course details are in the rightmost column of Table 1.)

SPM teaches students about management of software processes. It introduces students to other processes besides Scrum, such as Extreme Programming and Lean, and discusses issues such as scheduling, time estimation techniques, and risk management. Prior to adopting QU’s course collaboration framework at MTU, SPM students did not apply these principles to actual projects. (SPM course details are in rightmost column of Table 2)

3.3 Adaptation of collaboration at MTU

Due to the high similarity between QU and MTU’s TSP and SPM courses, adapting QU’s course collaboration at MTU seemed like a natural fit. Fundamentally, SPM students needed to be made managers for TSP teams, adding this collaboration to the curriculum of both courses. (See Table 3.) There were several program differences that needed to be addressed, however.

First, SPM was only offered in the fall at MTU, so the collaboration could not take place during the spring. No action was taken to address this difference, so the collaboration was applied during the fall 2019 and fall 2020 semesters, but not the spring 2020 and spring 2021 semesters.

Second, both courses at MTU were taught by the same instructor. No action needed to be taken to address this difference.

Third, TSP and SPM had no overlapping instruction time. The instructor for these courses proposed a program change to add an overlapping lab hour, but the proposal did not carry through. To address this, the instructor had both TSP and SPM students take a survey listing which hours they were free every week and used this information to assign teams where every team member had at least one overlapping hour every week so they could meet.

Fourth, the ratio of SPM to TSP students was different at MTU. Each semester, TSP is divided into two sections, with each section having about 35 students for a total of about 70, while each fall, SPM has about 45 students. The TSP students are divided into 15-16 teams. With this ratio, it was impossible to assign just one manager to each team. To address this, the instructor created two different managerial roles, a technical manager and a personnel manager, and moved the Scrum Master role from the TSP students to the SPM students. Having two managers plus a Scrum Master for each team created enough positions for every SPM student to have a role.

The technical manager’s responsibilities included enforcing revision control, testing, and build restrictions for the teams and assisting them in setting these up as necessary, as well as helping to pull tasks from the product backlog to create the backlog for each sprint. The personnel manager’s responsibilities included facilitating team communication and collaboration, reviewing each team’s peer reviews, resolving conflicts with the option of asking for instructor intervention if needed, and assigning user stories (tasks) to TSP developers. The Scrum Master’s
<table>
<thead>
<tr>
<th>Weeks</th>
<th>Interaction between TSP and SPM Courses (QU)</th>
<th>Interaction between TSP and SPM Courses (MTU)</th>
</tr>
</thead>
</table>
| 1     | In-class weekly meeting  
• Introductions and ice breaking activity  
Installation and evaluation of software system | None |
| 2     | In-class weekly meeting:  
• Team strategies for teamwork success  
• Requirements elicitation and specification  
Out-of-class:  
• Review requirements report | Out-of-class weekly meeting:  
• Introductions  
• Develop Project Ideas |
| 3     | In-class weekly meeting:  
• Vision statement and bug/enhancement prioritization  
Out-of-class:  
• Review vision document and backlog | Out-of-class weekly meeting:  
• Project planning: Scope, deliverables, develop an Architecture Decomposition View and Work Breakdown Structure |
| 4     | In-class weekly meeting:  
• Create work plan  
• Create project repository using git  
Out-of-class:  
• Monitor team progress | Out-of-class weekly meeting:  
• Project planning: Detailed work schedule, configuration management, testing plan |
| 5     | In-class weekly meeting:  
• User story and test plan development  
• Individual team member evaluation rubric development  
Out-of-class:  
• Team post-mortem meeting | Out-of-class weekly meeting:  
• Project planning: Risk management plan |
| First Week of Scrum Sprints (6, 8, 10, 12) | In-class weekly meeting:  
• Select backlog items and create a work plan  
Out-of-class:  
• Monitor team progress  
• Performance evaluations of individual team members (Week 10) | Out-of-class weekly meeting:  
• Select backlog items and create a work plan  
• Update time estimates for work units |
| Second Week of Scrum Sprints (7, 9, 11, 13) | In-class weekly meeting:  
• Monitor team progress  
• Suggest strategies for issues identified  
In-class last day of Scrum Sprint  
• Team demo/presentation  
Out-of-class:  
• Team post-mortem meeting | Out-of-class weekly meeting:  
• Monitor team progress  
• Sprint retrospective/postmortem  
• Team identifies areas for improvement and implements action plans to address those areas. |
| 14    | In-class weekly meeting:  
• Reflection on last Scrum Sprint and semester-long collaboration  
Out-of-class:  
• Team final report and presentation | None |

Table 3: Course Collaboration Comparison during Fall 2020
responsibilities included all the normal duties for that position, as well as scheduling meetings and recording meeting minutes. All three SPM students assigned to each team participated in writing reports, with different students being responsible for different sections of the reports. After analyzing the results of the fall 2019 semester, the instructor noted several areas for potential improvement, particularly focusing on:

- Using SPM students as Scrum Masters, even with that role’s responsibilities expanded, did not provide the same valuable managerial experience as the Personnel Manager and Technical Manager roles.
- Even with a clear division of responsibilities, three direct managers per TSP team could feel suffocating.
- Many students noted that the involvement and usefulness of the managers dropped sharply after initial project planning.

To address these issues, the instructor made several changes and applied them during the fall 2020 semester. First, the SPM Scrum Master role was dropped (the Technical and Personnel Manager roles were retained) and three supporting roles were created, with each supporting role being responsible for 3-4 TSP teams. These new roles were:

- Metrics Analyst: Responsible for assisting with scheduling, assigning tasks to developers, and tracking velocity.
- Testing Coordinator: Responsible for assisting in the design of tests, setting up testing environments, performing manual tests, and evaluating teams’ test plans.
- Middle Manager: Responsible for coordinating the other managerial roles for each team under their purview, facilitating communication between all managers involved with the team and the course instructor, evaluating the state of each project, and making suggestions for improvement.

In addition, in an effort to increase manager involvement throughout the lifetime of the projects compared to the previous semester, the technical and personnel managers were charged with planning each sprint by choosing tasks from the product backlog and assigning them to developers in coordination with the metrics analyst, and the technical manager was tasked with performing weekly code reviews to provide more regular technical assistance.

For the fall 2020 semester, due to institutional COVID-19 policy, the TSP and SPM courses were conducted remotely.

### 3.4 Comparing QU and MTU course collaborations

Table 1 compares the TSP courses at QU and MTU. Table 2 compares the SPM courses at QU and MTU. Table 3 compares the collaborations as implemented in fall 2020 at QU and MTU. QU and MTU have comparable schedules in their TSP courses where the first five weeks are devoted to providing appropriate background to prepare student teams to engage in agile development of a team-based project. Teams at QU are assigned a legacy project whereas teams at MTU select their own project subject to instructor approval. In both courses, teams complete four two-week development-oriented Scrum sprints (Teams at QU also complete an additional “Scrum Sprint 0” during weeks 3 and 4 that focuses on non-development tasks and allows students to experience...
Scrum before starting development). During the last week at QU, teams package their project to be handed off to another team during the next iteration of the course and prepare a demo and presentation for the exam week while at MTU teams prepare a comprehensive presentation of project development and a demo during the last week of classes.

The course structure of the SPM courses at QU and MTU is similar—students learn different SPM topics each week, apply their knowledge to the course collaboration, and take a midterm and a final exam. There is a difference in the topic presentation order and some small differences in the topic selection. While most of the topics covered in the SPM course at QU and MTU overlap, only QU covers ethics in SPM, organizational issues in SPM, and individual performance evaluation, whereas only MTU covers the development pipeline, a review of software processes, and service-oriented software. Most of these topics covered in the SPM course in one of the institutions, but not the other are covered in other courses in that institution’s curriculum.

The course collaboration begins the first week at QU since SPM students attend a lab session that overlaps with one of the weekly TSP class meeting. Each week, managers meet with their teams on SPM assignments that are coordinated with software development tasks in the TSP course. During the four Scrum sprints, these meetings focus on project management activities such as creating a work plan, monitoring team member progress toward completing assigned tasks, and reflecting on team performance to identify areas for improvement. Since there is not commonly-scheduled class time for TSP and SPM students at MTU, these activities are performed during separately scheduled meetings between managers and their teams.

4 Evaluation Methodology

To evaluate the course collaboration at MTU, we applied the methodology used to evaluate the course collaboration at QU [5]. In particular, we analyzed end-of-semester surveys completed by the students in each course and the instructor observations logged during the semester. The study has been approved by the Institutional Review Board (IRB) at MTU. The study at QU, which is referenced in this paper for comparison purposes, has also been approved by QU’s IRB.

4.1 Participants

Information about the participants in this study is included in the first four columns of Table 4. For comparison, the remaining columns include analogous information for the corresponding courses at QU.

4.2 Data Collection Instruments

End-of-semester surveys. At the end of the semester, the students in each course at MTU completed a survey designed to gauge their overall learning and experience as well as the effect of having or being a manager. Students report how the collaboration contributed to their achievement of the course learning outcomes (CLOs) on a 5-point Likert scale (Strongly Disagree to Strongly Agree). The CLOs for both courses are listed in Figures 1 and 2. The surveys also include questions pertaining to the students’ experiences with being or having managers, such as their thoughts on the amount of manager authority and involvement, the workload, and areas for
### Table 4: TSP and SPM Participant Demographics.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TSP Course</td>
<td>Enrollment</td>
<td>66/66</td>
<td>56/58</td>
<td>32/34</td>
<td>29/29</td>
<td>38/38</td>
<td>34/37</td>
<td>33/35</td>
</tr>
<tr>
<td></td>
<td>Number of Teams</td>
<td>16/16</td>
<td>16/16</td>
<td>8/7</td>
<td>8/7</td>
<td>8/8</td>
<td>8/8</td>
<td>10/10</td>
</tr>
<tr>
<td></td>
<td>Number of Managed Teams</td>
<td>16/16</td>
<td>16/16</td>
<td>4/7</td>
<td>7/7</td>
<td>5/8</td>
<td>8/10</td>
<td></td>
</tr>
<tr>
<td>SPM Course</td>
<td>Enrollment</td>
<td>46/46</td>
<td>45/45</td>
<td>2/2</td>
<td>9/9</td>
<td>7/7</td>
<td>5/5</td>
<td>12/13</td>
</tr>
<tr>
<td></td>
<td>Number of Managers</td>
<td>46/46</td>
<td>45/45</td>
<td>2/2</td>
<td>9/9</td>
<td>7/7</td>
<td>5/5</td>
<td>12/13</td>
</tr>
</tbody>
</table>

### Table 5: Response rates for end-of-semester surveys.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TSP Course</td>
<td>66/66 (100%)</td>
<td>56/58 (97%)</td>
<td>32/34 (94%)</td>
<td>29/29 (100%)</td>
<td>38/38 (100%)</td>
<td>34/37 (92%)</td>
<td>33/35 (94%)</td>
<td>39/41 (95%)</td>
</tr>
<tr>
<td>SPM Course</td>
<td>43/46 (93%)</td>
<td>43/45 (96%)</td>
<td>2/2 (100%)</td>
<td>9/9 (100%)</td>
<td>7/7 (100%)</td>
<td>5/5 (100%)</td>
<td>12/13 (92%)</td>
<td>11/12 (92%)</td>
</tr>
</tbody>
</table>

The surveys administered at MTU also included additional questions on the different managerial roles used at MTU and the skewed manager-to-managed students ratio necessitating these roles (as discussed in Section 3.3) as well as a question on comparing learning remotely in fall 2020 to on-ground learning (such a question was also asked at QU in fall 2020 but on a reflection paper, not on the surveys).

**Instructor observations.** During the semester, the TSP and SPM course instructor recorded observations from formal meetings with students, informal conversations, and other experiences related to the course collaboration. A major source of information were the instructor’s bi-weekly discussions with the SPM students where issues related to the collaboration were discussed with the managers. A pre-defined protocol was not used; at each meeting, the instructor asked students about their experience with the current components of the course and about their interaction with the managed teams specifically. Some information related to the course collaboration also emerged during the bi-weekly TSP student demos. The instructor also held regular meetings (every two or three weeks) throughout the semester with the instructors from QU to discuss current experience with the course collaboration. Some additional observations were elicited during these meetings.
5 Results

This section reports results from applying the course collaboration at MTU and for comparison includes the corresponding results from QU. The response rates for the end-of-semester surveys at each institution are shown in Table 5. The questions from the end-of-semester surveys discussed in this section are the ones that address the research questions stated in Section 1. Some of the other survey questions (described in Section 4) are pertinent to our course assessment and continuous improvement processes and are thus not discussed here. As previously stated, the course collaboration at QU started in 2015. Some of the questions on the end-of-semester surveys, however, were added later than 2015 as the collaboration was evolving and new aspects of it were identified for evaluation. In such situations, the figures in this section include the data from QU starting with the earliest year they were collected.

Figures 1 and 2 show the student responses to the end-of-semester survey questions related to the contribution of the course collaboration to the students’ achievement of the course learning outcomes (CLOs). Students answered these questions on a 5-point Likert scale, where 1 is “Strongly Disagree” and 5 is “Strongly Agree.” The CLOs for the SPM courses are the same at both institutions whereas the CLOs for the TSP courses are institution-specific.

Complementing these quantitative questions was the open-ended question, “Are there other skills or knowledge that you learned [by having managers/managing a TSP team]?” In 2019, 23 TSP students answered that question—13 answered ‘no’, 5 mentioned that hearing about managers’ previous experiences with their own TSP projects was valuable. In 2020, 11 students answered that question—4 answered ‘no’, 3 mentioned managers were helpful with facilitation/providing structure/improving teamwork skills, 3 mentioned the technical managers were most useful, and 1 mentioned that having too many managers was counter productive. In 2019 and 2020, 22 and 23 SPM students answered that question, respectively. There was a great diversity of answers including variations of developing interpersonal skills, learning how to resolve conflicts among team members, assisting with and teaching technical skills, and gaining experience with being involved in team software development from a managerial perspective—e.g., organizing, planning, scheduling—rather than a developer perspective. Two SPM students in 2020 also commented that the interaction with the TSP teams resembled their experience from internships/discussions with professionals.

Figures 3 and 4 show the students’ perception of the amount of authority the managers had and Figures 5 and 6 show the students’ perception of the amount of manager involvement with the managed teams. The perceived workload in the SPM courses is shown in Figure 7. A similar question was not included on the end-of-semester surveys for the TSP courses as the course collaboration did not substantially change the TSP students’ workload.

TSP students provided many common responses to the question “What about having managers for your team worked best?” (62 and 51 students answered that question in 2019 and 2020, respectively):

- The managers helped them stay focused, organized, and on-track. (24/62 in 2019; 31/51 in 2020)
- The managers were helpful for early project planning to get their projects pointed in the right direction. (11/62 in 2019; 4/51 in 2020)
A: Ability to determine the scope of a software project by taking into account various constraints.
B: Ability to develop a software project plan.
C: Ability to enact a software project plan.
D: Ability to estimate various software project parameters.
E: Ability to measure and control software products and processes.
F: Ability to manage software project risk.
G: Ability to lead a diverse team of software developers.

Figure 1: Self-assessed contribution of course collaboration to CLO achievement in SPM course.

A: An ability to work on a software product at the scale of a multi-month team project.
B: An ability to work on software development documents including a project plan, backlog, and postmortem.
C: An ability to use tools to manage the software development process.
D: An ability to develop and apply tests to demonstrate product quality.
E: An ability to work effectively on a team.
F: An ability to apply written communication in a team software development context.
G: An ability to apply oral communication in a team software development context.

Figure 2: Self-assessed contribution of course collaboration to CLO achievement in TSP course.
Figure 3: Amount of manager authority as perceived by students in SPM course.

Figure 4: Amount of manager authority as perceived by students in TSP course.

Figure 5: Amount of manager involvement as perceived by students in SPM course.
Figure 6: Amount of manager involvement as perceived by students in TSP course.

Figure 7: Amount of overall course workload as perceived by students in SPM course.
• Having managers with prior experience working on TSP projects was helpful. (10/62 in 2019; 8/51 in 2020)
• Technical managers were helpful for setting up development environments and resolving problems with source control. (7/62 in 2019; 5/51 in 2020)

As discussed in Section 3.3 to address the issues with the managerial roles identified at MTU in 2019, the managerial roles were modified in 2020 (the Technical and Personnel managers were retained, but the Scrum Master role was replaced by the Metrics Analyst, Testing Coordinator, and Middle Manager roles). Out of 39 responses to the question “Which managerial roles seemed more useful/less useful?” (not all survey participants addressed this optional open-ended question) 17 students (44%) expressed that the Middle Manager role seemed redundant and unnecessary. In general, students felt that the direct managerial roles (Personnel and Technical Managers) were more helpful than the other roles. The instructor also noted increased difficulty of collaboration and communication between students with so many roles, as well as increased complexity in regular reports.

Although in the previous semester students had specifically requested that the Technical Manager perform code reviews, once this responsibility was added, the TSP students did not appear to perceive the code reviews as helpful. The instructor notes that the Technical Managers never had substantive suggestions for improvement as a result of their code reviews, with the overwhelming majority of these reviews being simply, “The code is fine.” Some of the technical managers themselves made note of this in communications with the instructor.

6 Discussion
6.1 Addressing the Research Questions.

R1: To what degree are similar student learning experiences achieved at a second institution (MTU)? Overall, the student learning experiences at MTU seemed to be similarly positive to those at QU. As the left side of Figure 1 indicates, SPM students at MTU on average agreed (4 being “Agree” and 5 being “Strongly Agree”) that the course collaboration contributed to their achievement of CLOs. One CLO, “ability to manage software project risk” received a score lower than 4 in 2019. This was recognized as a potential issue at the end of the 2019 offering and the SPM course at MTU was modified by introducing risk management earlier in the semester and adding risk analysis as a required part of the project planning. This course modification seems to have worked as the corresponding CLO received a rating higher than 4 in 2020 and this was also the largest change in a CLO rating from 2019 to 2020.

The CLO ratings for the SPM course at MTU are similar to those for the last four years (2017-2020) at QU, which is when the course collaboration at QU stabilized as reported in prior work [5]. The first two years (2015 and 2016) at QU were largely exploratory and major changes to the course collaboration were made to address what was being learned. The fact that results from the first two years at MTU are comparable to those from the last four years at QU might be an indication that the course collaboration has reached a level of maturity that makes it fairly adaptable to other institutions, subject to the constraints described in the discussion of research question 3 below.
Besides the high ratings for the course collaboration’s contribution to the achievement of CLOs in the SPM course, SPM students at MTU reported that they learned a variety of other skills, such as organizational skills and conflict resolution as described in Section 5. These skills are consistent with those reported by the SPM students at QU [5]. This is a further indication of the educational benefits of the course collaboration.

In addition to the self-reported results from the end-of-semester surveys, the observations of the SPM instructor at MTU suggest that the course collaboration improved the educational experience of the SPM students. In particular, prior to the introduction of the course collaboration, the SPM students had little to no opportunity to apply “soft” project management skills, such as facilitating team communication and collaboration, conflict resolution, and reviews of team member performance. The introduction of the course collaboration also allowed the SPM students to exercise some of the more quantitative project management skills such as planning, estimation, measurement, and control in the context of a team of real people working on a real project. Real projects entail unpredictable challenges that are nearly impossible to be artificially replicated (such as team members failing to perform, communication breakdowns, large-scale technical failures), and the human factor is such a vital component of project management, that students gain a much greater understanding by applying management skills as opposed to merely being tested on their knowledge of them. Thus, our experience suggests that the course collaboration enables students to achieve educational objectives at the most advanced levels of the cognitive domain of Bloom’s taxonomy [26, 27].

The ratings for the course collaboration’s contribution to the achievement of CLOs in the TSP course at MTU (shown on the left side of Figure 2) are overall positive—they are on average around 3.5 with 3 being “Neutral” and 4 being “Agree.” When comparing these ratings to those at QU, it needs to be noted that the CLOs for the TSP course at the two institutions are very similar but not identical. The 7 CLOs at MTU overlap with the first 6 CLOs at QU, but the last two CLOs at QU do not have an analog among MTU’s CLOs. The CLO ratings at MTU are higher than the ratings from the first two years of the corresponding course at QU, which might be an indication of the collaboration’s level of maturity as previously discussed. At the same time, the ratings for the CLOs in the TSP course at MTU are lower than those for the last 4 years in the corresponding course at QU. We believe there is room for improvement in the course collaboration at MTU to increase its impact on the learning experience of the TSP students. The skewed manager-to-developer ratio (addressed in detail in the discussion of research question 3 below) might also be negatively affecting the experience of the TSP students.

Another factor that might be affecting the impact of the collaboration on the TSP students’ learning experience is institution size. QU is a smaller institution with smaller class sizes and a tightly knit student community of computing majors. The course instructors at QU have observed that many SPM students have gone beyond course expectations by meeting with their teams in addition to the mandatory weekly meetings as well as responding to team members’ questions via group chatting services. At larger institutions, such as MTU, additional scaffolding within the course collaboration might be needed to increase its impact on the learning experience of TSP students.

It is worth noting that the ratings for the course collaboration’s contribution to CLO achievement are higher for the SPM courses than for the TSP courses at both institutions. The difference at QU
is not as pronounced as at MTU and we believe that as the course collaboration at MTU evolves and incorporates some of the modifications discussed in the previous paragraph, that difference will become smaller. It is likely, however, that this difference will not disappear as the collaboration plays a central role in the learning experience of the SPM students, whereas it plays a secondary role in the TSP course. Many of the CLOs in the TSP course are supported by other course components. Regardless, a substantial number of students in the TSP course at both institutions have reported various benefits from the interaction with the managers, as discussed at the end of Section 5 for the TSP course at MTU and in previous work for the TSP course at QU [5].

From the observations of the instructor of the TSP course at MTU and their discussion with TSP students, it appears that many TSP students found it helpful to be able to draw on the experience of SPM students who have already taken the course. Many TSP students also found the managers helpful for keeping them on track and accountable. The instructor noted, however, that this last benefit seems to be manager-dependent as some managers are a lot more hands-off and TSP students might not be seeing much value in such managers. Additional ways to hold SPM students accountable to be more active with their teams will need to be explored. We might be able to draw from the experience at QU where managers are required to submit additional artifacts that demonstrate their involvement with their teams, such as work plans, team member evaluation rubrics, and reports on in-person performance evaluations.

Besides the educational benefits related to the achievement of CLOs, the students at MTU seem to have had positive experience with other aspects of the course collaboration. One of the most challenging aspects based on the experience at QU has been determining the right amount of manager authority. Too little authority has led to the SPM students acting as consultants instead of as managers whereas too much authority can be overbearing for the TSP students and also raises ethical issues about the amount of control one group of students has over the educational experience of another group of students. As reported in previous work [5] and as suggested by the right side of Figures [3] and [4], it has taken 2-3 years at QU for a large portion of the students in both the SPM and TSP courses to perceive the amount of manager authority as “just about right” (note that a question on authority was not included in the 2015 end-of-semester surveys as the authority issue was identified after the 2015 course offerings and a corresponding question was added starting with the 2016 surveys). In contrast, the results from the SPM and TSP courses at MTU shown in Figures [3] and [4] indicate that the appropriate amount of manager authority seems to have been achieved at MTU as soon as the course collaboration was introduced. We believe this suggests that the lessons on manager authority learned at QU seem to have successfully transferred to MTU. In particular, managers at MTU were given the authority to lead project planning and assign tasks to developers, whereas managers at QU were primarily advising their teams on these aspects during the first couple of iterations of the course collaboration. Also, like at QU, managers at MTU served as a “first port of call” for issues of human conflict and only escalated problems to the course instructor if they could not handle the issues themselves.

Based on the left side of Figures [3] and [4], the majority of the students at MTU perceived the amount of manager involvement as “just about right”, which is generally consistent with the perception of manager involvement at QU. Furthermore, a larger portion of the students at MTU reported the amount of manager involvement as “just about right” in 2020 than in 2019, which
might be an indication that the changes made to the course collaboration in 2020 described in Section 3.3 have positively affected manager involvement. It is interesting to note that almost all the students at MTU who did not rate the amount of manager involvement as “just about right” rated it “low” or “too low.” This seems to indicate that there is an opportunity for increasing manager involvement at MTU. Indeed, based on student feedback, in both semesters a substantial minority felt that the managers were uninvolved or otherwise not useful. This indicates that there is a minority of underperforming managers who do not provide useful contributions for their team. The combination of regular reports that managers must submit to the instructor, together with developer reviews of their managers, are intended to curtail the effect of managers shirking their duties, but the results indicate that the current measures in place might be insufficient. Code reviews were also found to not be a helpful contribution by the Technical Managers, so this requirement will be dropped for future semesters.

Some approaches to incentivize manager involvement were presented in the discussion of CLOs above. Another approach would be to make the quality of the projects a factor in the managers’ grades as well as the developers’. This practice is already used at QU, which anecdotally reports that student managers have specifically inquired about the project quality portion of their grade, suggesting that this is serving as an effective incentive to keep them involved. This approach does raise issues of fairness, however, since it is possible that managers could give a genuine best effort and the resulting project could nonetheless be poor due to unresolvable negligence or lack of ability among the developers. At QU this issue is addressed by the SPM instructor combining evaluations of the managers by their teams with personal observations based on the weekly discussions with the managers and the quality of collaboration-related submissions by the managers. If the SPM instructor deems a manager’s involvement sufficient based on the above observations/evaluations, then the portion of that SPM student’s grade related to the quality of the managed project is not lowered, even if that project’s quality is poor.

An important consideration when implementing the course collaboration has been its effect on the managers’ workload as the managers are the ones who carry most of the collaboration’s weight. In particular, at QU it has been observed that there is a trade-off between the amount of manager authority and involvement on one hand and the managers’ workload on the other. The right side of Figure 7, for example, shows an increase of the amount of perceived workload in the period 2016-2019 which coincides with efforts at QU to increase manager authority and involvement. In 2020, certain aspects of the course collaboration at QU were streamlined (e.g., the amount of collaboration-related documentation the managers had to submit and the way they submitted it), which might explain the significant shift toward “just about right.” At MTU, the vast majority of the students perceived the amount of workload as “just about right” in both 2019 and 2020 as indicated by Figure 7. This is certainly a positive result and it might also afford an opportunity to increase manager involvement at the expense of slightly increasing workload.

R2: To what degree can the collaboration at QU be replicated at another institution (MTU)?

The major components of the collaboration framework developed at QU were fully implemented at MTU: students from the SPM course served as managers of teams of students from the TSP course and applied skills learned in the SPM course to a real project and team via mandatory regular meetings, offline communication with their team, and written assignments related to
working with their team. The instructors at both institutions have reported that implementing the course collaboration framework does not result in substantial increase of workload compared to teaching the SPM and TSP courses in isolation and the educational benefits of the collaboration greatly outweigh the associated workload.

As discussed in Section 3.3, the overlapping lab hour at QU was not adopted at MTU due to administrative issues involving numbers of course credits and course scheduling. For the same reasons, SPM is only offered one semester, which simultaneously creates two limitations: This contributes to the skewed ratio of managers to developers, and means that about half of TSP students will not experience the managerial collaboration. The degree to which the collaboration can be applied depends significantly on such scheduling concerns.

Another difference between the collaboration at MTU and QU is that QU has students work on ongoing projects that carry over between semesters, while at MTU students work on self-chosen, self-contained projects each semester. This part of the collaboration was not adopted because no suitable code bases exist at MTU. Without a preexisting project for students to work on, this aspect of the collaboration cannot be used unmodified.

R3: What factors at the target institution (MTU) need to be considered in the adoption process? Three main factors were identified. The first was the existence of a course structure to support the collaboration. In particular, MTU already offers a course that focuses on SPM and another that focuses on team software development. As discussed in section 3.3, this structure was a natural fit for the collaboration framework. A lower-level course about team software development and a higher-level course about managing software projects are critical factors to be able to adapt the collaboration. Alternatively, either half of the collaboration could be supported in isolation. For a team software course, this would require a source of managers who could be recruited to manage the student projects. For a software management course, this would require a source of software projects for which the students could act as managers.

The second factor to consider was the ratio between students in the SPM and TSP courses. At QU, that ratio is usually in the range of around 1:3—1:7, whereas at MTU that ratio was around 1:1.5. Thus, at QU each team has had a manager or at most two co-managers, whereas at MTU the managerial position had to be split into several managerial roles (as discussed in Sections 3.2 and 3.3) to provide a meaningful educational experience. One can extrapolate that if the ratio were unbalanced in the other direction, individual SPM students may need to manage multiple teams. Creativity is required if the ratio of students enrolled in each course is unbalanced.

The third factor to consider was scheduling the weekly meetings between the managers and their teams. At QU, the SPM and TSP courses shared a common class period whereas at MTU that was not approved by the institution. Alternate scheduling arrangements were made by the instructor, but that caused inconvenience for both the instructor and the students. The lack of a common class period is a hindrance to the collaboration, but not a fatal one – the collaboration can still be applied successfully in the absence of a common period, but it will create scheduling and communication difficulties.
The structure of the TSP course at both MTU and QU is project-based and team-based. Prior to the pandemic, in the TSP course at QU, teams met in class three times per week for one hour where a portion of the time was available for team-based activities. On Mondays, managers from the SPM course met with their teams during the TSP class period. Outside of class time, teams could meet with their manager in person, but often met virtually using tools such as Discord. With the move to remote-only instruction caused by the pandemic, the opportunities to meet in person no longer existed. Instead, classes met synchronously via Zoom at the scheduled class time. Team meetings were conducted using Zoom breakout rooms. Managers continued to attend one class a week and met with their team in the team’s breakout room. Their management tasks remained largely the same and the managers with their teams achieved results comparable to meeting in person. When the TSP instructor randomly dropped in on a team meeting with their manager, the team was actively engaged in project-based activities with very few exceptions. The main disadvantage for the instructor was the inability to observe all teams with their managers at the same time, which was possible when all students were physically in the classroom. Outside of class, managers met with their teams exclusively using tools like Discord. Managers found it easier to attend their team’s presentations remotely, and, if a conflict occurred, the presentations were recorded via Zoom and available on the course website.

At MTU, upon switching to remote instruction, lectures were delivered as prerecorded videos. Since there was no overlapping lab hour, there was no difference in how teams scheduled their meetings except that far more students (but not all) chose to meet remotely instead of in person. Project demos were delivered as prerecorded videos instead of live presentations. Managers’ tasks remained largely the same.

Individual managers and team members reported a range of responses to the remote-only mode. Some preferred making presentation using Zoom because they felt less intimidated. Some reported that they would rather have met in person; however, others found meeting remotely was preferable. A few managers, team members, and instructors experienced technical difficulties at times that interrupted the course and teams’ ability to meet.

The instructors at both QU and MTU observed that the quality of work produced by the teams and their managers was comparable to work produced by teams and managers from previous years when the courses were offered on ground. However, the quality of presentations and demos was noticeably improved. Beyond being less intimidated, teams generally seemed to focus on slide quality since screen sharing was the main mode of delivery and students were able to prepare scripts that could be rehearsed and read without obvious notice as would be the case in person.

Two opportunities emerged from the remote mode of the courses in 2020. First, the collaboration had worked well for five years at QU and one year at MTU where we were confident that managers and team members were gaining valuable experience. Moving to the remote mode provided an opportunity to “stress test” the collaboration. This is beyond the scope of the research questions posed; however, it did help us assess the soundness of the collaboration structure. We believe the fact that the work produced in 2020 was comparable to that from previous years, that presentations and demos noticeably improved in 2020, and that the student responses on the
end-of-semester surveys were largely positive and comparable to those from previous years speaks to the resilience of the collaboration even in the face of major alterations and challenges. Second, in conversations with graduates who have had to adapt to working remotely, we consider that the move to a remote mode is similar for our students. While some workers will return to a physical workplace, the fact that there are decided advantages to working remotely means it will be a component of our team members’ and managers’ workplace experience. Having that experience, even though unanticipated, prepares them for the workplace they will be entering.

6.3 Other Observations

While the main goal of this work has been to apply and evaluate at another institution the course collaboration originally developed at QU, experiences and insights from MTU have led to changes to the collaboration at QU. For example, to stimulate TSP teams at QU to make substantial contributions to their software systems from one Scrum sprint to another, the concept of notable system improvement was introduced in Fall 2020. As teams chose items from their backlog in the beginning of each Scrum sprint, they were required to articulate in writing why the chosen items constituted a notable system improvement and had to have that improvement approved by both their manager and the TSP course instructor. The notable system improvement exercise reduces the probability that teams get away with making only minor/unsubstantial contributions to their software systems and was inspired by practices at MTU.

The peer evaluation forms used at QU were also modified based on those used at MTU. Originally, team members at QU rated their peers by distributing fictional dollar bonuses among the team members. After discussion with the MTU instructor and reviewing how peer evaluations were done at MTU, the peer evaluations at QU were changed to ask for ratings of explicitly defined categories on explicitly defined scale, which provided instructors and the team members with more detailed, actionable feedback.

The adaptation of the course collaboration at MTU has also led to additional ideas for future changes in the course collaboration at QU discussed in more detail in Section 7.

6.4 Limitations

While the collected data generally supports the analysis of the research questions above, we need to consider limitations. That we have replicated the collaboration at only one other institution limits our ability to generalize. Other institutions pose different challenges beyond those present at MTU. In particular, the similarity of courses and course structure at MTU facilitated the adaptation. Other institutions may lack this advantage, increasing factors that need to be considered beyond those reported here.

We are also limited by the size of the data set at both institutions. Although the number of students involved at MTU is approximately double that at QU, the data set is still relatively small. This limits the ability to do statistical analysis and make reliable statements about the significance of differences between the student responses over time or results between the two institutions.
The data collection instruments described in Section 4.2 were developed at QU by the TSP and SPM instructors to assess the course collaboration and gather data for continuous improvement. These instruments have not been validated or created from other validated instruments. While inferences drawn from the data collected via these instruments seem to have led to improvements at subsequent course offerings at QU and at MTU, interpretation is limited and not generalizable.

Much of the data used to analyze the similarity of student learning experience (research question 1) is self-reported, except for the instructor observations. While our primary goal was to evaluate learning experience, self-reported data of student learning does not necessarily represent actual learning and achievement of course learning outcomes (CLOs). Another approach to assess achievement of CLOs would be to identify and directly analyze student work related to the course collaboration, for example by using standardized rubrics. We believe, however, that the self-reported data coupled with the instructor observations discussed in Section 6 are an indication of the course collaboration’s positive contribution to student leaning.

As previously discussed, at both institutions, all courses involved in the collaboration were delivered remotely in fall 2020 due to health safety consideration of the pandemic. The change in instruction mode did not appear to significantly affect the results; however, it may have done so in ways not considered in our analysis.

7 Conclusion and Future Work

The course collaboration framework originally developed and evaluated at QU was successfully implemented at MTU. Overall, the learning outcomes and student experiences achieved at MTU were similarly positive to those achieved at QU. The major components of the framework were fully implemented at MTU with specific aspects customized to meet the educational outcomes and setting at MTU. In that customization, three factors were identified for consideration when adapting this framework: the existence of a course structure to support such collaboration, the ratio between the students in the two courses, and the ability to schedule regular interactions between these students.

In the future, we would like to further evaluate the course collaboration framework by applying it for more iterations at MTU and at additional institutions. We are also interested in approaches to quantitatively evaluate the impact of that framework on student learning to complement the evaluation based on self-reported impact by students and instructor observations.

Based on the experience of addressing the challenges of a high managers-to-developers ratio for two semesters, the MTU instructor plans to retain the personnel and technical manager roles, but replace the Scrum master (from 2019)/multiple roles (from 2020) with a “staff assistant” role that amalgamates the multiple roles from 2020, providing assistance with task scheduling and testing plans. This should provide the same benefits as the Scrum master role (providing the third SPM student with a non-authoritative role so as not to overburden TSP teams with a surfeit of managers) while hopefully providing the staff assistant with more useful administrative project experience than what is offered by a Scrum master’s duties. The instructor is also considering ways to further incentivize managers to be highly involved with the projects beyond the planning stage, such as making the TSP projects a factor in the managers’ grades as discussed earlier.
Given that introducing risk management early in the semester at MTU was found to be helpful, the QU instructors plan to explore possibilities for exposing students to this topic earlier in the semester as well to give them more time to experience the benefits of risk management throughout their projects.

References


