IACOB, C. and FAILY, S. 2020. The impact of undergraduate mentorship on student satisfaction and engagement, teamwork performance, and team dysfunction in a software engineering group project. In *Proceedings of the 51st ACM technical symposium on computer science education (SIGCSE 2020), 11-14 March 2020, Portland, USA*. New York: ACM [online], pages 128-134. Available from: https://doi.org/10.1145/3328778.3366835

The impact of undergraduate mentorship on student satisfaction and engagement, teamwork performance, and team dysfunction in a software engineering group project.

IACOB, C. and FAILY, S.

2020

OpenAIR @RGU

This document was downloaded from https://openair.rgu.ac.uk

CC BY NC

The Impact of Undergraduate Mentorship on Student Satisfaction and Engagement, Teamwork Performance, and Team Dysfunction in a Software Engineering Group Project

Claudia Iacob
University of Portsmouth
Portsmouth, United Kingdom
iacob@port.ac.uk

Shamal Faily
Bournemouth University
Bournemouth, United Kingdom
sfaily@bournemouth.ac.uk

ABSTRACT

Mentorship schemes in software engineering education usually involve professional software engineers guiding and advising teams of undergraduate students working collaboratively to develop a software system. With or without mentorship, teams run the risk of experiencing team dysfunction: a situation where lack of engagement, internal conflicts, and/or poor team management lead to different assessment outcomes for individual team members and overall frustration and dissatisfaction within the team. The paper describes a mentorship scheme devised as part of a 33 week software engineering group project course, where the mentors were undergraduate students who had recently completed the course successfully and possessed at least a year's experience as professional software engineers. We measure and discuss the impact the scheme had on: (1) student satisfaction and engagement, (2) team performance, and (3) team dysfunction.

CCS CONCEPTS

• **Applied computing → Interactive learning environments**;

KEYWORDS

Software engineering, teamwork, mentorship scheme

ACM Reference Format:
Claudia Iacob and Shamal Faily. . The Impact of Undergraduate Mentorship on Student Satisfaction and Engagement, Teamwork Performance, and Team Dysfunction in a Software Engineering Group Project. In *Proceedings of ACM Conference (Conference 2020).* ACM, New York, NY, USA, 7 pages.

1 INTRODUCTION

To successfully deliver an undergraduate software engineering group project course, two competing tensions need to be managed: team performance and team engagement. It is often difficult for instructors to fairly quantify team performance. The challenge comes from the difficulty of defining a metric for quantifying the quality of a team's submission that fairly reflects individual team members' contributions. While team engagement is less challenging to measure as a reflection of individual team members' engagement, it does significantly impact team performance. Therefore, managing team engagement in group project courses is paramount. Team engagement is often contingent on extraordinary situations that put significant stress on its members; these include lack of involvement of members of the team, personal conflicts, poor time and team management. Such situation may have ripple effects medium and long term causing team dysfunction and impacting team engagement, and as a result, team performance.

For managing team engagement in large software engineering group project courses mentors are often included in the design of such courses [2], [3], [9]. Usually, mentors are experienced professional software engineers who act as advisors [8]. While their role is clearly defined, their precise responsibilities are often flexible, and based on the requests of the teams and their overall performance. This flexibility is less of a liability when mentors are professional software engineers, but it can cause inconsistency and confusion if the role of the mentors is taken by undergraduate students themselves. On the other hand, professional software engineers are less available to act as mentors especially when the number of teams involved is high.

We designed a mentorship scheme involving undergraduate students as part of a 33 week software engineering group project course during the 2018/2019 academic year. The design of the scheme focused on precisely defined responsibilities for the mentors, mentorship sessions scheduled for all teams, and consistency in the advice and feedback given to all teams by all mentors. Our aim was to better understand the impact such a scheme has on the competing tensions of team performance and team engagement in software engineering group project courses. To achieve this, we consider three research questions: (1) How do students perceive and engage with a mentorship scheme in a teamwork exercise when mentors are themselves undergraduate students?, (2) Does an undergraduate mentorship scheme impact teamwork performance?, and (3) Does an undergraduate mentorship scheme impact team dysfunction?

In Section 2 we look at studies of peer-assisted learning in undergraduate courses and its impact on software engineering group work courses. We introduce the course and the mentorship scheme we put in place in Section 3 and Section 4, respectively. We present the design of the study run in Section 5. We illustrate the answers to the three research questions asked by the paper in Section 6, and we further discuss the implications of these results in Section 7 before drawing conclusions in Section 8.

2 RELATED WORK

2.1 Mentoring in Undergraduate Courses

Mentoring is a form of knowledge exchange, where someone with more experience provides guidance to someone less experienced. Mentoring relationships are common in professional practice, and are increasingly being introduced into the computing curriculum to expose students to mentoring relationships, both as mentees and mentors. Mentoring models in undergraduate computer science education and experience reports with these models are reported in [11] and [7]. Holmes et al. [5] describe how, for a capstone course where students collaborate with open-source projects, industrial mentors are assigned to students. Mentors played varied roles, ranging from helping students learn new development skills and helping them comprehend programming tasks and code structures. They were, however, particularly useful for emphasising code quality, and reasoning about non-functional aspects of code. The different roles of mentors are elaborated in by Hatfield et al. [4], who identified several roles that mentors fulfilled as part of a HCI course where mentors were used. These range from facilitator, design and software developer expert, and reflective observer. However, mentors were encouraged to complete their role in such a way that their own experience would add the most value in helping students learn. Davis & Rebelsky [2] describe the use of alumni mentors for a revised software design course. Although their department has a track record of peer mentoring, the decision was made to use alumni mentors given their knowledge of the technology ecosystem and their potential to act as role models. Collectively, the mentors also acted as an advisory panel for all project teams. This is particularly useful as mentors can provide institutional memory when instructors move to new roles.

2.2 Peer-assisted Learning

Peer assisted learning is the the practice of "more able students helping less able students learn in co-operative working pairs or small groups carefully organised by a professional teacher" [13]. Topping [13] has identified the most dominant forms of peer assisted learning as (i) cross-year small-group tutoring, (ii) personalised system of instruction at a student's own pace, and (iii) supplemental instruction that target high risk courses. Peer-assisted learning can assist students in improving their reasoning and study skills while, simultaneously, increasing engagement and student retention rates. Not all students feel comfortable seeking the support of the course instructors when they fall behind, so such instruction can promote student interaction and mutual support [1]. It can also provide useful feedback to instructors on the problems that students encounter. Tennyson et al. [12] describe how peer-assisted learning techniques were incorporated into an introductory programming course. Peer interactions allowed students to help each other learning different topics. While students learned more from the traditional teaching methods, they enjoyed the peer-assisted sessions more. Wills & Finkel [14] describe their experiences using peer learning to instil in students the need to take responsibility for their learning and the learning of their colleagues. For an introductory computer science course, student peer learning assistants (PLAs) were used to support instructors in labs and facilitate group learning outside of timetabled teaching sessions. They found that, as interaction with PLAs increased, students accountability for their own learning increased. While the intervention led to better retention rates, dealing with less motivated students remained a challenge, given their failure to meet deadlines affects their group as well as themselves. Simpson & Storer [10] report their experiences using final year undergraduates as mentors for software engineering group projects. Mentors were responsible for monitoring teams during labs and provided formative feedback on their work. They noted that students considered mentors role models, and that their formative assessment support was helpful in developing team-mentor relations.

3 ABOUT THE COURSE

We run a 33 week software engineering group project course where students work in teams on 6-7 members and develop a medium size software system of their choice. We define medium complexity as a system including an element of storage, one or more software components encapsulating some application logic, and a graphical user interface component. Teams are free to choose the application domain for the systems they develop. Examples of such systems include educational quizzes, product reviewing systems, and finance management systems. Methods, tools, and techniques for tackling common software development problems are introduced during the lectures. However, students are free to select which ones to apply for their project. Teams are given two constraints. First, the problem chosen needs to be of medium complexity and approved by the course coordinator. Second, each team is expected to submit the following five deliverables throughout the year (Figure 1):

1) Project proposal and plan (PPP) worth 10% of the final mark - dedicated to presenting the project idea and a plan for achieving it.

2) System requirements specification document (SRS) worth 25% of the final mark - including the requirements specification and the description of the elicitation process.

3) Design documentation (Design) worth 25% of the final mark - including elements of software design such as architectural model, data model, mock-ups.

4) Video demo of their prototype and the source code and test cases accompanying it (Proto) worth 30% of the final mark

5) Retrospective of the project (Retro) worth 10% of the final mark - detailing the development process put in place, lessons learned, and the main challenges overcome throughout the process.

All deliverables are group submissions, and each deliverable is assigned one mark, markA. Should the team declare even contributions from all its members, markA is awarded to everyone. If uneven contributions are declared, each team member specifies his/her individual contribution to the submission. The individual marks are then further decided based on the percentage of each individual contribution. The course is structured around one hour weekly lecture sessions and one hour fortnightly practical sessions. The lecture sessions cover theoretical aspects of software engineering related to project management and planning, requirements engineering, software design, implementation, and testing. The practical sessions are dedicated to peer reviewing deliverables post submission, exam preparation, talk shops, and software engineering task simulation sessions around key tasks in Software Engineering (eg. requirements engineering, architectural design) [6].

4 MENTORSHIP SCHEME DESIGN

During the 2018/2019 academic year, we implemented a mentorship scheme as part of the course described in Section 3. We involved six mentors. Five of the mentors were recruited via an open call sent to selective undergraduate final year students. The criteria for selection was successful completion of the software engineering group project course, with a mark in excess of 70%. Additionally, candidates were required to have at least a year experience of professional software engineering experience. The five undergraduate mentors selected completed a one year internship in software development companies, holding positions such as testers, developers, and requirement analysts. Additionally, all achieved marks in excess of 70% in the software engineering group project course during the 2016/2017 academic year. The sixth mentor was the course coordinator, who had spent several years as a professional software developer prior to her career in academia. Each of the five undergraduate mentors mentored 5 teams, and the course coordinator mentored 7. All mentors were randomly assigned to teams.

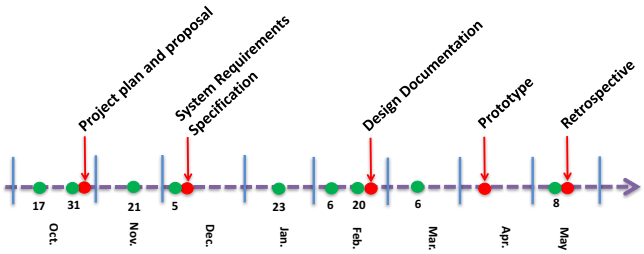

Figure 1: The timeline of the mentorship sessions (represented as green dots) and the course deadlines (represented as red dots)

Mentor responsibilities included providing feedback on draft deliverables, mediating team conflicts, discussing the requirements of each deliverable, and advising teams on course related issues. These issues included both technical aspects of the project, such as tools to use, most suitable methods to put in place for specific tasks, but also team management aspects of the project, such as strategies for splitting work or remote communication. These responsibilities were communicated to all mentors prior to the academic year, and written documents detailing these responsibilities were shared with all the mentors at the start of the academic year.

The scheme was designed around 9 mentorship one-hour sessions scheduled throughout the year as shown in Figure 1. These session were scheduled in addition to the lectures and practical sessions designed for the course. At least one session was scheduled before each deliverable deadline. The aim of these sessions was for teams to get feedback on their deliverable drafts (Oct 31, Dec 5, Feb 20, Mar 6, May 8). The aim of all the other sessions was to discuss the requirements of each deliverable, and address any course related issues (Oct 17, Nov 21, Jan 23, Feb 6). No session had a precise agenda, but an overview of all the sessions and their respective aims was provided to all mentors before the mentorship programme began. Reminders for each session and its aim were emailed to all mentors and all teams a couple of days before the session was scheduled. All sessions were optional for all teams. Four group meetings were organised with all mentors throughout the year in September, November, January, and March. The aim of these meetings was to both escalate any significant issues noticed for any of the teams and to ensure that all the information mentors have about the deliverables and the course is consistent. All undergraduate mentors were remunerated for their work.

5 STUDY DESIGN

This paper looks into three elements of the course and the impact the mentorship scheme has had on them: student satisfaction and engagement, teamwork performance, and team dysfunction.

5.1 Research Questions and Participants

We aim to answer the following research questions:

RQ1: How do students perceive and engage with a mentorship scheme in a teamwork exercise?

RQ2: Does an undergraduate mentorship scheme impact teamwork performance?

RQ3: Does an undergraduate mentorship scheme impact team dysfunction?

Thirty-five teams were enrolled on the course during the 2017/2018 academic year, and thirty-two teams were enrolled on the course during the 2018/2019 academic year. All students were allowed to form the teams themselves during the first practical session designed for the course during both years. The course coordinator did not interfere in the team formation process.

5.2 Metrics

As part of the study we define the following metrics:

a) Student satisfaction: measures the level of satisfaction students have with the course both half way through its delivery and at the end of the academic year.

b) Team attendance record: measures the number of mentorship sessions where the team was represented by at least one of its members.

c) Team performance: is defined as the final mark a team is awarded for the overall project. This mark is calculated based on the individual weights assigned to each deliverable and discussed in Section 4. As contributions are not necessarily even, it is common for different members of the same team to be awarded different marks. We associated each team with the highest mark awarded to any of its members, i.e. the mark assigned to the submission.

d) Team dysfunction coefficient: is measured as the ratio between the number of different final marks assigned across the team for the project and the number of team members. A low ratio indicates that all the members of a team were awarded the same marks across all the deliverables. This would be the case for teams where all individual contributions were even for each deliverable. A dysfunction coefficient of 1 would indicate that every team member on the team got a different overall final mark than all the other members. This would be the case with teams where individual contributions varied significantly across the year for all deliverables.

5.3 Data Collection and Analysis

We used a mixed method approach and collected data using multiple venues.

Mid-year feedback collection. As part of the mid-year (mid December) feedback collection process, we asked all students enrolled on the 2018/2019 course to anonymously comment on the mentorship scheme. We asked the open question "What are your thoughts on the mentorship scheme?". We collected all the answers and we associated each with one or more themes which summarised the answer. Such themes included suggestions for improving the scheme or reported uses for the mentorship sessions. For each theme we calculated the percentage of answers we associated with it.

End of year feedback collection. As part of the 2018/2019 end of year (beginning of May) feedback collection process, students were asked to anonymously comment on additional things they would implement as part of the course (Q1: What should we start doing, and why?), things they would not want to see on the course in the future (Q2: What should we stop doing, and why?), and things they enjoyed as part of the course and they think should be continued (Q3: What should we continue doing and why?). We collected all answers, and filtered those that mentioned the mentorship scheme. We calculated the percentage of answers referring to the mentorship scheme for each of the questions in the end of year student feedback survey.

Attendance. For each mentorship session, we tracked the number of teams that were represented by at least one member. Given the optional character of the sessions, there was no requirement for the entire team to attend. For this reason, we did not track the number of students who attended each session, but the number of represented teams.

Performance. We calculated the teamwork performance for all the teams enrolled on the course in 2017/2018 and 2018/2019. We grouped the team performance values in five intervals as depicted in Figure 3. We then calculated the percentage of teams awarded values in each interval for the academic year 2018/2019 when the mentorship scheme was put in place and 2017/2018 when the scheme was not put in place.

Dysfunction. We calculated the team dysfunction coefficient for all the teams enrolled on the course in 2017/2018 and 2018/2019. We considered three levels of team dysfunction as depicted in Figure 5. A low level of dysfunction is assigned to any dysfunction coefficient lower or equal to 0.3. A medium level of dysfunction is assigned to any dysfunction coefficient higher than 0.3 and lower or equal to 0.6. Any dysfunction coefficient higher than 0.6 is assigned with a high level of dysfunction. We calculated the percentage of teams associated with dysfunction coefficients at each level for the academic year 2018/2019 when the mentorship scheme was put in place and 2017/2018 when the scheme was not put in place.

6 RESULTS

6.1 Student Satisfaction and Engagement

We collected 52 (28%) individual answers from students during the mid-year feedback collection process. After coding these answers, we identified four themes students reported: criticism of the mentorship scheme (12% of the answers), suggestions for changes to be applied to the mentorship scheme (21% of the answers), reported uses for the mentorship sessions (50% of the answers), and positive feedback on the sessions ((85% of the answers). The positive feedback reported labelled the sessions as being 'helpful', 'constructive', and 'productive' with respect to the project. In terms of reported uses, students used the sessions to ask for advice or tips from their mentors, check and receive feedback on drafts of their deliverables, and ask questions about the course. Students found it particularly useful to get the opinion of someone who had taken the course before. Some of the suggestions students provided include organising more sessions throughout the year, involving more mentors for a lower ratio team/mentor, organising sessions longer than an hour, and allowing the rotation of mentors so that each team receives advice from all mentors throughout the year. The main criticism from students was that advice would sometimes be given that contradicted the requirements of the project.

A total of 12 (17%) responses provided by students during the end of year feedback collection process related to the mentorship scheme. Most of these answers (27%) were in response to the question 'What should we continue doing'. Answers included *"Having mentors helping with the project who have previously completed it themselves"*, and *"Mentors make project assessments easier to understand and improve"*. The scheme related answers to the question 'What should we start doing' (16%), reiterated the need for more consistency across the feedback and advice provided by the mentors. They also suggested organising more sessions in the future. A small number of answers (9%) to the question 'What should we stop doing' referred to the mentorship scheme, and these answers highlighted a need for mentors to be provided with more information about each project deliverables.

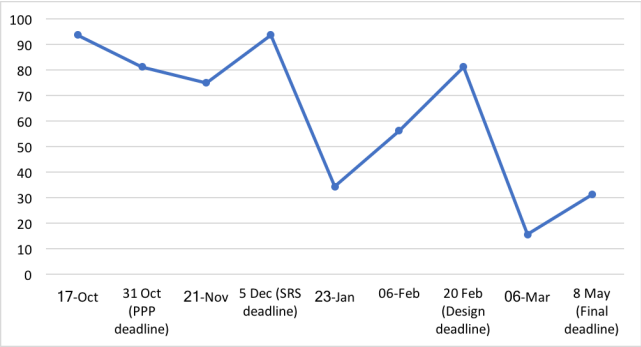

Figure 2: Percentage of teams represented in each mentorship session across 2018/2019

While attendance gradually declined throughout the year, significant spikes were noted during the sessions scheduled directly before deliverable deadlines (Figure 2). The only exception for this was the session scheduled before the final deadline where only 31.2% of the teams were represented. This final deliverable was designed as a 3 page retrospective of the project, which provided a reflective account of the team's experience with the project. The highest attendance rate was recorded for the first session and the

session scheduled prior to the System Requirements Specification deliverable deadline; 93.7% of the teams were represented in both these sessions. The lowest attendance rate was recorded during the session scheduled after the Design Documentation deliverable deadline, with representation from 15.6% teams.

6.2 Teamwork Performance

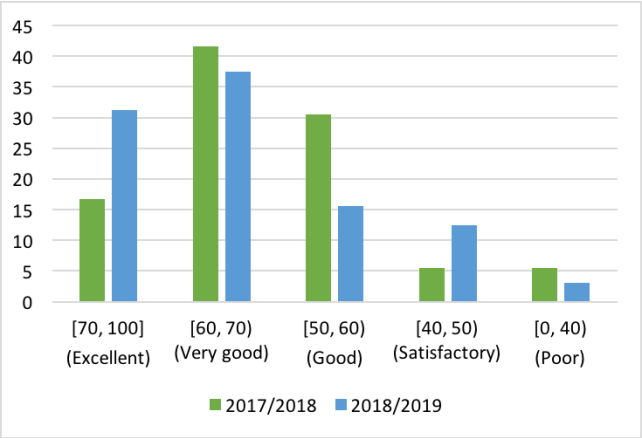

Figure 3: Teamwork performance across 2018/2019 (mentorship) and 2017/2018 (control) academic years

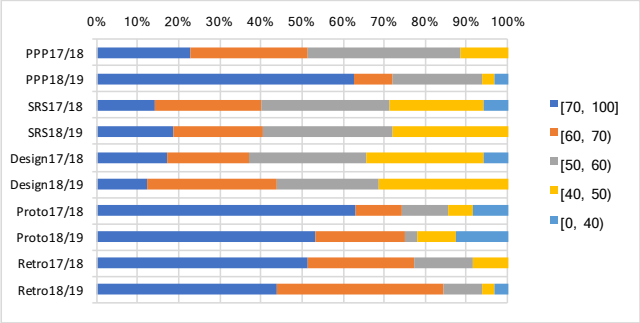

Figure 4: Teamwork performance per deliverable across 2018/2019 (mentorship) and 2017/2018 (control) academic years

Unsurprisingly, team performance during the 2018/2019 academic year correlated significantly with the team attendance record in the mentorship sessions ($r=0.52$, $p<.01$). We noted no significant correlation between team performance and team size. The percentage of teams that obtained an overall project mark greater than or equal to 70% doubled for the year we ran the mentorship scheme compared to the previous year, while the percentage of teams that obtained an overall mark lower than 40% was halved (Figure 3). The average of the overall marks for the year we ran the scheme was 62.21 (st.dev=11.22, median=62.5), while the average of the overall marks for the previous year was 59.61 (st.dev.=10.89, median=61). At deliverable level (Figure 4), the changes were less dramatic with the exception of the PPP deliverable where the percentage of teams awarded a mark in excess of 70% tripled compared to the previous year. For the SRS and the Design deliverables, no teams were awarded a mark less than 40%; this was not the case for the previous year. The percentage of teams awarded marks in excess of 60% was higher during the 2018/2019 academic year for all deliverables when compared to the previous year.

6.3 Team Dysfunction

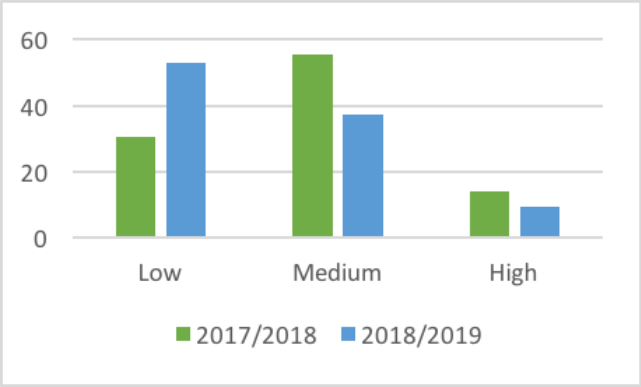

Figure 5: Team dysfunction rate across 2018/2019 (mentorship) and 2017/2018 (control) academic years

Team dysfunction for the 2018-2019 academic year was moderately negatively correlated with the team size ($r=-0.31$, $p<0.01$). Surprisingly, we noted no significant correlation between team dysfunction and team performance, or between team dysfunction and the team attendance record. The percentage of teams associated with a low dysfunction coefficient almost doubled for the year we ran the mentorship scheme (Figure 5) compared to the previous year. The percentage of teams associated with a high dysfunction coefficient was lower, with 9.3% of the teams being associated with a higher than 0.6 dysfunction rate compared to 13.8% the previous year. The average team dysfunction coefficient for the 2018-2019 academic year was 0.33 (st.dev=0.21, median=0.24), while the average team dysfunction coefficient for the previous year was 0.41 (st.dev=0.20, median=0.33).

7 DISCUSSION

How do students perceive and engage with a mentorship scheme in teamwork? Students' perception of the mentorship sessions was overall positive. At the start of the year, the sessions were significantly more popular in terms of attendance than they were at the end of the year. Spikes, however, were seen throughout the year for sessions scheduled prior to deadlines. Teams were initially curious about the efficiency of the scheme and interested in getting to know their mentor and his/her experience with the course. As the year progressed, teams became more strategic about attending these sessions, focusing precisely on what they mostly wanted to gain out of them - feedback on their work prior to deadlines.

Feelings on the role of undergraduate students as mentors were mixed. On the one hand, teams were interested in knowing more

	2017/2018			2018/2019			T test	
	Teams	**Mean**	**Median**	**Teams**	**Mean**	**Median**	**p**	**t**
Performance	35	59.61	61	32	62.21	62.5	.16	0.97
Dysfunction		0.41	0.33		0.33	0.24	0.08	1.40

Table 1: 2017/2018 and 2018/2019 team performance and dysfunction comparison

about the course and the expectations for it from someone who has recently completed the course successfully. Similar to the work reported in [3], teams also felt more at ease discussing sensitive topics such as assessment and room for error with a colleague rather than a member of staff. On the other hand, some teams doubted the accuracy of some of the information provided by the mentors, and asked for confirmation from either the course coordinator or other teams. Any inconsistency spotted between the information received and the information other teams received from their respective mentors further eroded trust.

It was apparent from the mid year student feedback that more mentors and more sessions would have been helpful to students. Students felt a sense of security knowing they could turn to their mentors for advice, and wished that this advice was more accessible in more sessions. Waiting for two weeks to address an issue in the mentorship session was felt to be too long, and led to issues either being resolved in the meantime with mixed results or forgotten. While it is unclear how attendance in additional mentorship sessions would have fluctuated, teams felt that having the option to speak to a mentor more readily would have provided more reassurance.

Scaling up the mentorship scheme by involving more mentors needs to be carefully considered by the course coordinator. One of the criticism often reported on the mentorship scheme described in this paper was the inconsistency in both the information provided by mentors and the vehicle for providing this information. For a small number of teams, involving one mentor per team is acceptable. However, for a larger number of teams, a level of abstraction is required to ensure a balance between consistency and helpfulness. More work is required to better understand the right team : mentor ratio. However, based on the results obtained through the scheme described in this paper, we believe that five teams per mentor is a reasonable approximation.

Does an undergraduate mentorship scheme impact team performance? Overall, team performance improved as a result of running the mentorship scheme. Two results are significant. First, the percentage of teams receiving a mark higher than 70% doubled when the mentorship scheme was put in place. We believe two factors explain this improvement. On the one hand, teams had access to early feedback prior to submitting deliverables. This feedback focused specifically on the marking scheme, and its timing allowed teams to change the deliverables prior to submission and better answer the deliverables' requirements. On the other hand, teams perceived the mentorship sessions as an incentive to start the work on each deliverable early, thereby allowing time for asking for and incorporating feedback. Second, the percentage of teams that failed the group project (i.e. were awarded a mark less than than 40%) halved the year when the mentorship scheme was put in place. Mentors were instructed on assessing the threshold for a pass for each deliverable, and made sure that all deliverables they reviewed met this threshold. Clear and simple marking schemes and thresholds for a pass are critical for a mentorship scheme, especially when the number of mentors involved is higher.

Does an undergraduate mentorship scheme impact team dysfunction? Overall, the rate of team dysfunction as defined in Section 6.3 was lower for the year the mentorship scheme was put in place when compared to the previous year. We explain this by a number of factors. First, the mentorship sessions provided an informal environment for raising issues within the team that was open to all teams. Discussing issues that occurred within the team was, therefore, part of the course and required no formal meeting with the course coordinator. Teams felt more at ease when discuss any conflicts and disagreements, and – because other teams were present – helped them appreciate that all teams were experiencing similar issues. Second, conflicts or misunderstandings were escalated to the mentors early. This ensured they were resolved early and with less effort. Additionally, defining conflict resolution as part of the mentors' responsibilities was helpful, and ensured issues were aired early and as part of the sessions.

8 CONCLUSIONS AND LIMITATIONS

The mentorship scheme improved several key aspects of the course. First, attendance spikes were noticed for the sessions scheduled prior to a deliverable deadline. Such spikes reached a maximum of 94% representation for the teams for the System Requirements Specification document. 85% of the student evaluations of the scheme reported positive feedback, with half of the responses describing the ways teams made use of the mentorship sessions. Second, teamwork performance increased as a result of the mentorship scheme. The average of the overall marks increased, and the percentage of teams obtaining an overall project mark greater than or equal to 70% doubled compared to the previous year. Finally, we defined a team's dysfunction coefficient based on the ratio between the number of different final marks assigned across the team for the project and the number of team members. This allowed us to identify that the percentage of teams associated with a low dysfunction coefficient almost doubled when compared to the previous year.

As discussed in Section 4, one of the mentors involved was the course coordinator. As a result of this it is possible that the teams mentored by the course coordinator may have been less open to discussing certain queries with their mentor. However, none of the answers collected via the mid-year feedback and end of year feedback collection process, raised this as an issue. All deliverables in both 2017/2018 and 2018/2019 academic years were marked by the course coordinator.

REFERENCES

[1] Robert A. Blanc, Larry E. DeBuhr, and Deanna C. Martin. 1983. Breaking the Attrition Cycle: The Effects of Supplemental Instruction on Undergraduate Performance and Attrition. *The Journal of Higher Education* 54, 1 (1983), 80–90.

[2] Janet Davis and Samuel A. Rebelsky. 2019. Developing Soft and Technical Skills Through Multi-Semester, Remotely Mentored, Community-Service Projects. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 29–35. https://doi.org/10.1145/3287324.3287508

[3] Meg Fryling, MaryAnne Egan, Robin Y. Flatland, Scott Vandenberg, and Sharon Small. 2018. Catch 'Em Early: Internship and Assistantship CS Mentoring Programs for Underclassmen. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. ACM, New York, NY, USA, 658–663. https://doi.org/10.1145/3159450.3159556

[4] B. Hartfield, Terry Winograd, and John Bennett. 1992. Learning HCI Design: Mentoring Project Groups in a Course on Human-computer Interaction. In *Proceedings of the Twenty-third SIGCSE Technical Symposium on Computer Science Education (SIGCSE '92)*. ACM, 246–251.

[5] Reid Holmes, Meghan Allen, and Michelle Craig. 2018. Dimensions of Experientialism for Software Engineering Education. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '18)*. ACM, New York, NY, USA, 31–39. https://doi.org/10.1145/3183377.3183380

[6] C. Iacob and S. Faily. 2018. Redesigning an undergraduate software engineering course for a large cohort. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering and Education Track (ICSE-SEET '18)*. IEEE Press.

[7] Yasmin Kafai, Jean Griffin, Quinn Burke, Michelle Slattery, Deborah Fields, Rita Powell, Michele Grab, Susan Davidson, and Joseph Sun. 2013. A Cascading Mentoring Pedagogy in a CS Service Learning Course to Broaden Participation and Perceptions. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*. ACM, New York, NY, USA, 101–106. https://doi.org/10.1145/2445196.2445228

[8] Joseph Maguire, Nathalie Sheridan, Steve Draper, and Quintin Cutts. 2019. Mentoring Mentors in Cooperative Software Engineering Education Programmes. In *Proceedings of the 2019 ACM Conference on International Computing Education Research (ICER '19)*. ACM, New York, NY, USA, 307–307. https://doi.org/10.1145/3291279.3341205

[9] Heather Pon-Barry, Audrey St. John, Becky Wai-Ling Packard, and Chris Stephenson. 2017. Addressing the CS Capacity Challenge by Improving Undergraduate Peer Mentoring. *ACM Inroads* 8, 3 (July 2017), 43–47. https://doi.org/10.1145/3123732

[10] R. Simpson and T. Storer. 2017. Experimenting with Realism in Software Engineering Team Projects: An Experience Report. In *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE T)*. 87–96.

[11] Rahman Tashakkori, James T. Wilkes, and Edward G. Pekarek. 2005. A Systemic Mentoring Model in Computer Science. In *Proceedings of the 43rd Annual Southeast Regional Conference - Volume 1 (ACM-SE 43)*. ACM, New York, NY, USA, 371–375. https://doi.org/10.1145/1167350.1167453

[12] Matthew F. Tennyson, John Casteele, and Andres Ricardo Pena Morena. 2018. A Study of Peer-assisted Learning in Introductory Programming Courses. *J. Comput. Sci. Coll.* 33, 5 (May 2018), 55–62.

[13] K. J. Topping. 1996. The effectiveness of peer tutoring in further and higher education: A typology and review of the literature. *Higher Education* 32, 3 (Oct 1996), 321–345.

[14] Craig E. Wills and David Finkel. 1994. Experience With Peer Learning in an Introductory Computer Science Course. *Computer Science Education* 5, 2 (1994), 165–187.