An Intelligent-Agent Facilitated Scaffold for Fostering Reflection in a Team-Based Project Course

Sreecharan Sankaranarayanan, Xu Wang, Cameron Dashti, Marshall An, Clarence Ngoh, Majd Sakr, and Carolyn Rosé

Carnegie Mellon University
{sreechas, xuwang, cdashti1, haokanga, pcn, msakr, cprose}@cs.cmu.edu
http://teel.cs.cmu.edu/

Abstract. This paper reports on work adapting an industry standard team practice referred to as Mob Programming into a paradigm called Online Mob Programming (OMP) for the purpose of encouraging teams to reflect on concepts and share work in the midst of their project experience. We present a study situated within a series of three course projects in a large online course on Cloud Computing. In a 3x3 Latin Square design, we compare students working alone and in two OMP configurations (with and without transactivity-maximization team formation designed to enhance reflection). The analysis reveals the extent to which grading on the produced software rewards teams where highly skilled individuals dominate the work. Further, compliance with the OMP paradigm is associated with greater evidence of group reflection on concepts and greater shared practice of programming.

Keywords: Online Mob Programming · Project-Based Learning · Computer-Supported Collaborative Learning · Conversational Agents

1 Introduction and Prior Work

Although team-project based courses are valued as opportunities to integrate and apply knowledge while refining skills learned in more basic courses, the anecdotal experience of many is that the inherent reward structure resulting from grades based on the quality of the end-project fosters a performance orientation, where the most capable students take on the lion's share of the work, doing tasks they already know how to do well, while undercutting the opportunity for other students to practice and for the group to reflect on underlying concepts. The contribution of this paper is a new intelligent-agent enabled paradigm for project based teamwork in computer science courses that is based on an adaptation of an industry standard team practice referred to as Mob Programming, with the goal of combating this tendency, and fostering more equal sharing of practice opportunities and group reflection on concepts.

Online Mob Programming is adapted from the industrial practice of Mob Programming, where a group of 3-6 participants rotate through four roles in order to afford participants the opportunity to experience the work from distinctly

different vantage points [11, 10]. Each participant will experience several rounds of all the roles throughout a single mob programming session, getting an opportunity to contribute as well as observe different perspectives and approaches to solve problems in the same session. The roles include - **Mob**: A participant or group of participants who consider and deliberate between multiple alternative implementations ultimately informing the decision of the Navigator. **Navigator**: A single participant who solicits input from the Mob, decides on the next action and communicates that to the Driver to be implemented into code. **Driver**: A single participant who converts high-level instructions from the Navigator into code. **Facilitator**: A single participant who observes and intervenes when necessary, such as to indicate when roles are to switch and to keep the activity progressing. This role is taken up by an Intelligent Conversational Agent, which monitors group processes [7, 5] and supports the uptake and rotation of roles.

Prior work on in-person Mob Programming describes benefits including a structured process for utilizing distributed knowledge, a unanimous positive perception of the process from the knowledge sharing, learning, and developer satisfaction perspectives [4], and the ability to learn from more experienced developers [3]. Remote Mob Programming [6] has also been attempted.

OMP participants collaboratively code in the online AWS Cloud9 IDE which includes an editor, terminal, text-chat and file navigation all on one screen. The intelligent conversational agent based on the open-source Bazaar framework [1] is integrated into the text-chat and uses a combination of static scripts that structure the activity, and dynamic role assignment based on the number of students in the chat room at any given time. Additionally, the agent receives data from the chat and the code edits to determine instances of compliant behavior associated with each role and highlights them as examples that participants can emulate. The introduction of the agent opens up the possibility of more dynamic context-sensitive conversational support for students and their roles in the future.

2 Method

Because of the distribution of responsibilities to roles in an inter-dependent fashion, we can hypothesize that - **Hypothesis 1** - Teams that demonstrate increased compliance to the OMP paradigm will discuss project-relevant conceptual content more substantively, contribute work towards the group solution more evenly, and produce a group product with as high of quality as individuals or teams with lower compliance. In designing this current study, we build on prior work developing a team formation strategy that provides benefits associated with idea sharing [9] and reduced problems with distribution of labor and conflict [8]. This team assignment paradigm uses a measure of observed exchange of transactive discussion [2, **?**] as an estimate of pairwise collaboration potential between students and then groups teams within a class in such a way as to maximise the estimated pairwise collaboration potential within teams across the class as a whole [9]. Transactivity as a conversational construct can

potentially interface well with the hypothesized benefits of OMP including more even distribution of work and more substantive discussions. Thus, **Hypothesis 2** - Groups formed transactively will demonstrate higher compliance with OMP practices that will then be associated with an intensification of the observed benefits of OMP compliance.

In order to test the two above hypotheses, we experimentally contrast the mob programming scaffold in randomly formed and transactively formed groups against individual programming in a 3 (Condition) x 3 (Programming Assignment) Latin Square between-subjects design embedded within a completely online, graduate Cloud Computing course offered to the students of Carnegie Mellon University and its campuses worldwide. Students first participate in a training activity with random groups. They are then assigned to one of three tracks within which they are assigned to individual, transactively formed, and random teams with 4-5 students per team such that no two students who have worked together in one group are together in another (including the training session). The conditions are then counterbalanced across tracks in order to prevent ordering effects. A total of 120 students took the course allowing 40 students to be assigned to each condition for each exercise. Each activity, including a training activity in the first week of the course, lasted 80 minutes with 10 minutes reserved for introductions and wrap-up and roles switching every 7 minutes. The role-switching was kept relatively frequent in order to promote observation of the problem from multiple perspectives.

3 Results and Discussion

Code contributions, chat logs, grades, post-assignment and post-course survey data was collected in all conditions to facilitate our analyses. In order to quantify whether students complied with the structure suggested by the OMP activity, we calculated a *Compliance Score* which was measured as the ratio of code contributions by the driver to the average code contributions by the other team members. Since the driver is expected to make all of the code contributions, a higher compliance score constitutes more compliance with the OMP structure. We further calculated the percentage of code contributions made by each group member which was used to compute an *Evenness Deviation* score, which measured the difference between this percentage and what percentage would be observed for the group if work was distributed evenly. Finally, in order to quantify the extent of activity-relevant *Conceptual Content* being discussed in the chat, we measured the vector similarity of the topic representation of the chat of a student with the primer corresponding to that activity using a latent semantic indexing model with the number of topics set to 5. A higher document similarity score meant that more of the conceptual content from the primer was discussed in the chat.

We begin by checking to ensure that students in all three conditions achieved equivalent grades on the assignments, and indeed, there was no significant difference. In order to test the extent to which the reward structure substantially does

encourage an uneven distribution of labor, we computed an ANCOVA model with a three-way split on the Evenness Deviation variable (Top Quartile, Middle, Lower Quartile) as the independent variable, average grade prior to the experiment as Covariate, Assignment time point as a random variable, and Grade at time point as the Dependent variable. The upper quartile had deviation scores of higher than .33, and the lower quartile had deviation scores less than .08. The median deviation score was .2. In 3 percent of teams, a single member did all of the work. There was no significant effect of the deviation variable on grade, though the trend was in the expected direction both for the auto graded and manually graded portions of the assignment. Thus, students may falsely believe it is necessary to deviate from an even distribution of labor when in fact it does not help their grade.

We first tested for an association between Compliance scores with Conceptual Content scores and found the association to be highly significant ($R=.26$, $p < .005$) such that students in more compliant groups focused more on conceptual content in their chats. Then we tested for an association between Compliance scores and Evenness Deviation scores. In this model, there was no main effect of condition, and in Random teams, there was no effect of Compliance, but in Transactive teams, there was a marginal effect such that more compliant teams had a lower Evenness Deviation score $F(1,149) = 1.93$, $p = .055$). We also tested for an effect of compliance on grade and there was no significant or marginal effect of Compliance in either condition. Thus, we have correlational, though not causal, evidence to support the first hypothesis that Mob practices are associated with more conceptual focus and more even distribution of labor without harm to grade on assignment.

To test the second hypothesis we computed an ANOVA model with Condition as the independent variable and Assignment time point as a random variable. Compliance score was the dependent variable. Here we found a trend consistent with the hypothesis, but it was not significant. Thus, Hypothesis 2 is not supported, though above we indicated that Transactivity maximized teams showed a more even distribution of labor when they complied, whereas the Random assigned teams did not. It is possible that the OMP structure acts as its own scaffold for idea exchange, which might explain why the primary enhancement we observe of the Transactivity maximization condition is that students in that condition cooperated better in terms of division of labor.

Overall, this study provides correlational evidence in favor of OMP as a set of team practices that might serve to counter the performance focus of team behavior in project courses and instead encourage teamwork and reflection on project relevant concepts. One limitation of the current study is that all of the teams were trained and supported in OMP practices. Without manipulating whether OMP was encouraged, we lack causal evidence in favor of the value of OMP. Thus, a follow-up study is necessary to obtain such evidence.

This research was funded in part by the National Science Foundation grants ACI-1443068, IIS 1546393, and IIS 1822831 as well an AWS Educate Grant, Microsoft Azure Educator Grant Award and a Google Cloud Platform Grant.

References

1. Adamson, D., Dyke, G., Jang, H., Rosé, C.P.: Towards an agile approach to adapting dynamic collaboration support to student needs. International Journal of Artificial Intelligence in Education **24**(1), 92–124 (2014)
2. Azmitia, M., Montgomery, R.: Friendship, transactive dialogues, and the development of scientific reasoning. Social development **2**(3), 202–221 (1993)
3. Buchan, J., Pearl, M.: Leveraging the mob mentality: An experience report on mob programming. In: Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018. pp. 199–204. ACM (2018)
4. Kattan, H.M., Oliveira, F., Goldman, A., Yoder, J.W.: Mob programming: The state of the art and three case studies of open source software. In: Brazilian Workshop on Agile Methods. pp. 146–160. Springer (2017)
5. Kumar, R., Rosé, C.P., Wang, Y.C., Joshi, M., Robinson, A.: Tutorial dialogue as adaptive collaborative learning support. Frontiers in artificial intelligence and applications **158**, 383 (2007)
6. Malmgren, U.: Remote mob programming set up (Jan 2017), https://natooktesting.wordpress.com/2017/01/30/remote-mob-programming-set-up/
7. Rosé, C.P., Ferschke, O.: Technology support for discussion based learning: From computer supported collaborative learning to the future of massive open online courses. International Journal of Artificial Intelligence in Education **26**(2), 660–678 (2016)
8. Sankaranarayanan, S., Dashti, C., Bogart, C., Wang, X., Sakr, M., Rosé, C.P.: When optimal team formation is a choice-self-selection versus intelligent team formation strategies in a large online project-based course. In: International Conference on Artificial Intelligence in Education. pp. 518–531. Springer (2018)
9. Wen, M., Maki, K., Dow, S., Herbsleb, J.D., Rose, C.: Supporting virtual team formation through community-wide deliberation. Proceedings of the ACM on Human-Computer Interaction **1**(CSCW) (2018)
10. Wilson, A.: Mob programming-what works, what doesnt. In: International Conference on Agile Software Development. pp. 319–325. Springer (2015)
11. Zuill, W., Meadows, K.: Mob programming: A whole team approach. In: Agile 2014 Conference, Orlando, Florida (2016)