Identifying instructor interventions to improve student group dynamics in long-term group projects

Discussion leaders: Dr. Adam J. Lee and Bill Garrison, Computer Science.

Dr. Lee and Adam discussed the progress on and ideas about their project for which they received a dB-SERC Mentor-Mentee Award which involves analysis of course repository metadata to identify productive group dynamics. The goal of the project is to determine what instructor interventions can help improve the quality and outcomes student group work on long-term projects.

The project is being carried out for a junior-senior level course in network systems and cryptography. Course background:

- Heavy project-based course
- In-class lectures serve to provide students with information and skills required to carry out various aspects of the project
- Students work in groups of 2-3 on a joint project to propose their own solutions to various security threats that can arise in a data sharing environment
- Student groups:
    - Form early: the first group assignment is given on the 2nd lecture
    - Students are allowed to choose their own groups
        - Usually, ~75% of students find a group immediately – mostly based on friendships formed in previous classes
        - The rest will either find a group in the 2nd class, or get assigned a partner if they cannot.
    - Anecdotally, some groups are homogeneous, others are not, possibly because they are mostly formed between students who know each other
- This course is one of the first electives computer science students take, and it is usually the first time students are required to work in groups in a significant fashion.
    - Students may have some experience studying together or helping each other out with various projects, but they have little to no experience working together on huge projects, editing online documents and code etc.
- A big part of the instructor effort is to ensure that students work well in groups.

The first time Dr. Lee taught the course, he realized that groups are not working well together at all, which resulted in very poor end-of-semester projects. Therefore, in the next offering (which was taught by Bill under the supervision of Dr. Lee), students were forced to complete their projects in an online data repository. The project is mainly focused on analyzing repository data to identify patterns of successful groups which can be used to inform future implementations of the course, for example by introducing incentives for students to engage in the productive habits of successful groups.

Project:

- Develop a secure file-sharing system
- Students are given 5000 lines of code to start with and first they must make it work without ensuring that security threats are taken into account.
- In the next phases of the project, students are introduced to various threats, and they have to take steps to ensure that the file-sharing system is secure.
- For each phase, the instructor meets with students in each group. Students have to:
    - Submit a 3-5 page write-up which describes:
        - The threat
        - Possible solutions
        - Which solution was chosen and why
    - The primary purpose of this meeting is for the instructor to get a pulse on who is doing how much of the work in each group.
    - A secondary purpose is to ensure that students have thought about designing a defense against the security threat carefully before implementing it.
- Throughout the semester, students are also given various homework assignments designed to help them develop the skills required to complete the project.

Several problems have been identified in the past:

- Uneven work
    - Sometimes because one student is lazy, but other times there is a student who wants to take charge, which can be problematic for other students in that group who want to be included.
- Lack of communication
    - Students don't always work well in teams. Sometimes, their work overlaps in non-synergistic ways, for example, when one student overwrites another students' code, and the first student operates under the assumption that his code is the same.
- Procrastination
    - Students start with 5,000 lines of code, and often, introducing additional subroutines to take care of various security threats does not involve a whole lot of code. However, a lot of thinking has to go behind these additions, and often students overlook this ("well, it's just gonna be another 200 lines of code, how long is that going to take"), and attempt to do it last minute.
- Juggling design and code
    - Related to the previous issue: students must be very meticulous about the design of the various additions they will put in before coding, and students need to closely communicate about all aspects of the project.
- Difficulty integrating code written by different people
    - Students have little experience doing this – they are used to working on their own projects.
- Design and code splitting

    - Sometimes students divide labor as one student works on the design and the other works on the coding, but in the past this has just not worked. Both students need to understand both aspects in order for the project to be successful.

In Spring 2015, students were forced to use a software version control in which the code was housed in an online repository. While working on the project, students make various modifications and then submit a commit which includes a log of what changes were made.

- This makes students more reflective about the changes they are making, and can help ensure that they make 'minimally useful commits' – i.e., work on a small enough part that they can do in a reasonable amount of time, but the part is important enough to push the project forward
    - Before you commit, you must think carefully about what you have done.
- In addition, the online repository provides a lot of rich data which can be used to identify which students did what/when. It essentially provides the instructor (who has access to the repository) a history of how the project evolved.
- All this data can be used to identify patterns of successful groups, which in turn can be used to identify instructor interventions to improve group dynamics.

During semester:

- Repository data can be used to identify problems early, for example, one student may be doing most of the work. Then, the instructor can meet with the group to determine why that may be. Perhaps they are splitting labor in unproductive ways (e.g., one student doing the design, the other the coding).

Between semesters:

- Identify interventions to improve group dynamics
    - For example, perhaps successful groups use a lot of comments in their code (to clarify how it works for someone who is unfamiliar with it). Then, a future offering of the course can require students to use comments frequently and their projects can be assessed also on their clarity for someone who did not work on that project.

Dr. Lee and Bill then gave several examples of how the data collected can be used:

- In the first example (slide 9), the third student had the lowest stability (% of code which remains unmodified). This could indicate that this student did the initial modifications and the other two students finished up.
- Third example (slide 11), one student did the coding and the other completed the write-up required by the instructor. The natural question can arise – did student 2 understand the coding in detail? When they meet with the instructor, he/she can ask this student to explain various parts of the code or subroutines.

Regarding instilling productive group work, one idea discussed was to give students roles that can rotate. For example, one student can be given the role of “reflector” – how well did our group work on this phase of the project? How often did we meet, and were those meetings productive? What lessons can we learn to improve how we work in the future? Etc.

- Students are required already to do “self-assessments” in which they assess the contribution of each group member, but perhaps these can be expanded and done by one student for each phase (i.e., role rotates).
- Students can also be given organizational, or oversight roles which require very little oversight from the instructor. For example, one student can be tasked with organizing group meetings and another with ensuring that a certain amount of work is done each week in order for the project to be moving forward.