

## THE CHANGING ROLE OF COMPUTING EDUCATION: FOSTERING COLLABORATION

Tyson R. Henry, California State University, Chico, [tyson@ecst.csuchico.edu](mailto:tyson@ecst.csuchico.edu)  
Janine LaFrance, California State University, Chico, [janine@ecst.csuchico.edu](mailto:janine@ecst.csuchico.edu)

---

### ABSTRACT

*In all computing fields (computer engineering, computer science, software engineering, information technology, information systems) nearly every aspect of software development and evolution is performed by groups. Industry advisory boards, national curriculum committees, industry recruiters, researchers, and industry authors have been telling the education community for years that students need to learn the collaborative skills that allow them to be productive in a group. The education community has taken this advice seriously. For many years students in all computing fields spend some time working in a group.*

*While this approach has improved the situation, it has not solved it. Industry representatives still complain that new graduates do not have the appropriate skills to make them successful group participants. There are three problem areas: student culture that prevents successful group work, a lack of the collaborative skills necessary for making groups productive, and projects that are not structured appropriately for a group. Now that we understand it is not enough to simply tell students to work in groups, the academic community must assume responsibility for making student groups successful. This paper describes the problems that prevent student group success and presents suggestions on how to mitigate these problems and subsequently improve the collaboration skills of new graduates.*

**Keywords:** Student Groups, Collaboration, Active Learning, Communication Skills, Course Pedagogy

### INTRODUCTION

The academic computing community has responded to the computing industry's demand for collaborative skills by having students work in groups. This action is based on the assumption that students will learn the collaborative skills necessary for group success simply by experiencing group work. After many years of this approach, the complaints from industry have not changed. In fact, they have increased. Industry advisory boards and recruiters have gone so

far as to say that the ability to work in a group is more important than technical ability. Some recruiters are even willing to overlook technical shortcomings for students who have particularly strong interpersonal skills.

The current approach to group projects can worsen the problem. Negative group experiences reinforce students' negative attitudes toward group projects. These negative attitudes become a serious liability when job hunting. Recruiters will use a candidate's description of group experiences to measure the candidate's ability to work in a group. If all or most of the experiences are reported as negative, the recruiter will often assume the student is unable to work in a group.

It is thus very important for instructors to understand the problems that arise in student group projects and to actively work to ensure the success of student groups. The next section describes the barriers to group success. Then solutions are presented.

### BARRIERS TO PRODUCTIVE GROUPS

The problems students encounter when working in a group can be categorized in three broad areas: student culture that prevents group success, students not having the skills necessary for productive group work, and group projects that are not structured well for group success. This section addresses each of these areas. The next section provides methods to help mitigate these problems.

#### Student Culture

Waite *et al.* performed an extensive study of student culture looking for factors that inhibit group success [12]. They used communication researchers trained in ethnographic observations and in interviewing to study computer science students over three years. They observed 12 classes and performed over 130 student interviews. The interviews elicited information about students' background in the field, industry experience, approaches to solving class problems, and their attitudes toward group work. Their findings are summarized in Table 1.

**Table 1.** Student Culture

Cultural Observation	Explanation/Ramification
Students prefer to work alone	Students want to bear sole responsibility and get sole credit.
Procrastination	Virtually every student told stories of procrastination which hinders collaboration.
Experimentation	Students often would rather solve a problem using experimentation (trial and error) than gain an understanding of the problem and then develop the solution.
Disregard for process	In general students ignored process.
Combativeness	Most students boldly display their own opinions and disqualify the opinions of peers.
Unwillingness to support others	Some students told stories of not getting support from peers and/or professors. Other students told stories of not wanting to support other students.
Absence of passion	Without passion shared among the group it is hard to overcome the problems inherent in group work.

Clearly, students arrive in the classroom with beliefs and preferences that inhibit or even prevent successful group work. These beliefs and preferences are pervasive and create a student culture that does not support group work. Thus when students are told to work in a group, they often devise mechanisms that allow them to work independently. Waite *et al.* describe three tactics students use to avoid working collaboratively [12].

- Sequential segmentation: Students take turns working on the system
- Parallel segmentation: Students break the system into pieces and work independently.
- Natural selection: All students do the entire project and the best project is chosen of the best student is chosen.

The need is not for students to learn how to work independently while appearing to work in a group. The need is for students to learn how to work collaboratively with others. These cultural barriers must be eliminated before students can work effectively in groups so that they will be prepared for a career based on working in groups.

**Lack of Collaborative Skills**

There is no universal list of the collaborative skills necessary for productive group work. The terms soft skills and interpersonal skills are often used to categorize this skill set. However, these terms do not capture all aspects of collaborative skills. The following list contains skills commonly referenced in the computing and education literature[1, 9] :

- Communication (speaking, writing, listening, and presentations)

- Organizational (personal and group)
- Decision making
- Conflict resolution
- Leadership
- Defining goals
- Time management
- Motivational skills (self motivation and motivating others)
- Managing quality

Many instructors do not address any these skills; they simply tell students to work in a group. Instructors must find ways to train students in these skills. While it would be impossible to address every skill directly, the goal is to move in the right direction by addressing some of them.

**Project Structure**

The structure of a group project influences the chance of student success. The project structure aspects that contribute to the success or failure of a group project include grading (the method used to determine a grade and grading emphasis given to group projects), well defined milestones and deliverables, definition of student roles, methods used to form groups, rules used to run groups, project choice, and provided collaboration tools

**BREAKING THE BARRIERS:  
FOSTERING COLLABORATION**

This section provides a compilation of ways to foster collaboration by changing student cultures, teaching collaboration skills, and restructuring projects.

## Changing Student Culture

In order to improve collaboration skills, the characteristics of student culture that inhibit collaborative work must be changed. Student culture cannot be changed by lectures on group work or by telling students to work in groups. Instructors must establish conditions that favor the development of a new collaborative culture. Ultimately, students must be shown that the advantages of collaboration outweigh their perceived cost [12].

The most direct way to change student culture is to provide positive group experiences. Instructors must develop processes and establish conditions that favor development of a collaborative culture and improve the chances of success. A student's entire view of group work can be significantly changed by a single positive group experience.

An in class exercise can also be used to change student culture. Waite *et al.* describe a decision-making exercise that addresses the combative nature of student culture [12]. Students are asked to solve a problem with non-trivial solutions and a high degree of variability. Then the entire class discusses the problem and develops a single solution by consensus.

## Teaching Collaborative Skills

Collaboration skills cannot be taught by lecture. Students must actively engage in exercises that demonstrate the skills and provide evidence that the skills are worthwhile. These exercises must allow students to participate via individual activities and must give them the opportunity to interact with other students [8]. This section presents several exercises and approaches that can be used to build collaboration skills.

In-class tutorials can be used to teach collaborative skills. Tutorial topics include how to run a meeting, using specialization as a technique for a group to accomplish a task, setting early goals, stages of group development (forming, storming, norming, performing), decision making, managing conflict, and managing quality [1, 9]. To be effective, a tutorial must provide exercises that students complete while working with other students. Without active participation, the tutorial becomes a lecture and most likely will not engage students.

Exercises that get students to reflect on their experiences are a powerful teaching tool [6]. For example, in-class role-play exercises can be used to introduce students to many of the aspects of group

work [7]. However, when students are asked to reflect on their participation in a role-play exercise, they often gain significant insight into the nature of group work and into the need for collaborative behavior. It is the reflective process that makes role-play a powerful mechanism for teaching collaborative skills.

Ludi *et al.* describe several exercises in which groups of students solve problems during lecture. These exercises include case studies, writing requirements, designing a system component, and developing a prototype. Each exercise is designed so that each group must reach a consensus and develop a single deliverable [8].

Garrett *et al.* use in-class debate to teach communication skills. They argue that debate stimulates critical thinking, increases confidence and speaking ability, helps students examine their own perspective of issues, increases students' awareness of conflicting points of view, exposes students to divergent points of view, and allows students to practice communication skills [5].

Another way to teach collaborative behavior is for the instructor to exhibit collaborative behavior and become a role model. For example, this can be achieved by replacing lectures with discussions. When an instructor presents class material by facilitating a discussion, he yields his authority to the students [12]. Student group projects sometimes fail because the group members are not willing to yield any authority, and they spend all their time fighting for authority.

## Structuring Group Projects for Success

Group projects must be designed to help students overcome the inherent problems with group work. For example, the way groups are formed, the deliverables, the schedule, the tools provided, and the grading mechanism can all effect project success. This section provides suggestions for structuring group projects so that they help the students succeed.

One problem students face is a lack of understanding of the different roles in a group. Students must be provided a clear description of the common roles: supervisor, client, programmer, tool assistant, etc. [1, 9]. Each group must agree upon and then explicitly document the roles and responsibilities of each member. This explicit documentation allows the group to form a convenience that the group can follow. Having a convenience reduces task

assignment ambiguities and reduces task assignment difficulties [11].

Group composition is crucial to their success. Well-formed groups tend to be exceptionally productive. Poorly formed groups often become so preoccupied with group issues that they are not at all productive. Negative group experiences tend to further polarize students against group work. Many instructors allow students to form their own groups or assign groups randomly. It is possible to do a much better job forming groups. When forming groups, student preferences, skills, personality, work habits, motivation, desired grade, and schedule should all be considered [1]. While it is difficult to create well formed groups, the payoffs are extraordinary.

The choice of projects can have a significant impact on student motivation and ultimately on group success. It is important to provide challenging and novel projects from a domain that students find interesting [3]. Open-ended projects give students the choice to excel. Students will often work harder than necessary on open-ended projects. It might be that they don't know when to stop working, or it might be that they want to demonstrate their abilities to their peers. Allowing a group to pick their project gives students the chance to exercise their creativity. It is usually the case that a class full of students is more creative than a single instructor. Students tend to have a greater sense of ownership and thus be more motivated when they are allowed to invent their own project. Churcher and Cockburn have found that students are more motivated to work on projects that have a "real world" flavor [2]. They suggest that projects should be modeled after industry practices. For example, reverse-engineering of existing systems, extensive code re-use, GUI development, interviews with customers, and oral presentation.

Students tend to procrastinate. If a semester project is due at the end of the semester, most students won't start the project in earnest until a few weeks before the deadline. Regular deadlines and milestones motivate students to be productive throughout the semester [2, 9]. For example, projects in a software engineering class often have documents due throughout the semester, but the software is not due until the end of the semester. Requiring students to give a software demonstration to the class mid semester forces them to start on the software early. Regular status reports and contribution logs also help keep students on track. If each group is required to regularly turn in a one page status report (every two weeks works well) it makes it clear how much work has been accomplished and students will motivate

themselves each week so that the status report is not empty. Requiring students to regularly turn in a log of individual contributions to the project keeps students on task. Even the most diehard procrastinator does not like to admit to procrastination in writing, and slackers do not like documentation of their slacking.

Waite *et al.* argue that group assignments should not be a large fraction of the course grade [12]. They believe that if a group project is a large portion of the course grade then slackers can pass the class on the coattails of productive group members. On the other hand, if more emphasis is placed on test grades, slackers will not be able to pass the class. A third alternative is to assign a relatively small project grade to each project and a contribution grade to each group member. This increases the assessment burden but reduces a slacker's chance of slipping through the cracks. The method also alleviates tension between non-slackers and slackers; the non-slackers know the slacker will be graded fairly. Individual contributions can be measured by requiring students to submit contribution logs and by requiring students to rate the performance of their groupmates.

Port *et al.* [10] emphasize risk management in their student groups. Each week the groups monitor the risks associated with their project and produce a report documenting the current assessment of each risk and the progress toward resolution. They also have the groups pay particular attention to those features of the project that simplified or complicated development tasks for the group. When a student group focuses on risk management, they communicate about their project on a different level. Students are often surprised to learn what other group members think are risks. The task of prioritizing risks requires students to articulate why a risk is important, debate group members' opinions, and finally to agree on a ranking. Building an awareness of risks increases the potential for success.

The final suggestion is to provide software support that promotes collaboration. The instructor should provide students with an on-line mechanism for storing and accessing documents (a version control system), communication mechanisms (mailing groups, bulletin boards, a chat program), and web pages for storing and accessing documents [4,9].

## CONCLUSION

For many years the computing industry has requested that educators teach students the skills necessary to work in a productive group. The academic

community has responded seriously to this request. Most students graduate with significant experience working in groups on class projects. However, years of this approach has not solved the problem, industry is still complaining. It has become apparent that having students work in groups does not ensure they will learn the skills necessary to be a productive member of a group. A proactive approach is needed. Computing faculty must accept the responsibility for group success. They must actively work to teach the collaborative skills necessary and must take steps to ensure student projects are successful. This paper provides a description of the problems inherent in group work and a summary of the most promising approaches for teaching collaborative skills.

### REFERENCES

1. Brown, J. & Bobbie, G. (1999). Supporting and evaluating team dynamics in group projects. *Proceedings of SigCSE*.
2. Churcher, N., & Cockburn, A. (1997). An immersion model for software engineering projects. *Proceedings of the 2nd Australasian Conference on Computer Science Education*.
3. Daniels, M., Faulkner, X., & Newman, I. (2002). Open ended group projects, motivating students and preparing them for the 'real world.' *Proceedings of the 15th Conference on Software Engineering Education and Training*, 115-126.
4. Diwan, A., Waite, W., & Jackson, M. (2002). An infrastructure for teaching skills for group decision making and problem solving in programming projects, *SIGCSE 2002*.
5. Garrett, M., Schoener, L., & Hood, L. (1996). Debate: A teaching strategy to improve verbal communication and critical-thinking skills, *Nursing Educator*, 20(4), 37-40.
6. Hazzan, O., & Tomayko, J. (2004). Reflection processes in the teaching and learning of human aspects of software engineering, *Proceedings of the 17th Conference on Software Engineering Education and Training*, 32-38.
7. Henry, T., & LaFrance, J. (in press). Integrating role-play into software engineering courses. *Journal of Computing Sciences in Colleges*.
8. Ludi, S., Swaminathan, N., & Reichlmayr. (2005). An introductory software engineering course that facilitates active learning. *ACM SIGCSE Bulletin*, 37(1), 302-306.
9. Mennecke, B., Bradley, J., & McLeod, M. (1998). Making project groups work II: The impact of group process training and role assignment on the performance and perception of information systems project teams. *Proceedings of the 13th Annual Conference of the International Academy for Information Management*.
10. Port, D., & Klappholz, D. (2004). Empirical research in the software engineering classroom, *Proceedings of the 17th Conference on Software Engineering Education and Training*, 132-137.
11. Towns, M. (1998). How do I get my students to work together? Getting cooperative learning started. *Journal of Chemical Education*, 75(1), 67.
12. Waite, W., & Leonardi, P. (2004). Student culture vs group work in computer science, *SIGCSE2004*.