Student Culture vs Group Work in Computer Science *

William M. Waite
Michele H. Jackson
Amer Diwan
University of Colorado
Boulder, CO 80309

Paul M. Leonardi
Stanford University
Stanford, CA 94305
Leonardi@Stanford.edu

{William.Waite,Michele.Jackson,Amer.Diwan}@Colorado.edu

ABSTRACT

Our industrial advisory boards tell us that our students are well prepared technically, but they lack important group work skills. Simply adding project courses and requiring that assignments be done in groups has not improved the situation. A careful study of student culture in Computer Science has uncovered barriers to collaboration, which can be overcome only by pervasive changes in the way we approach our curriculum.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computers and Information Science Education

General Terms

Design, experimentation, performance, human factors

Keywords

Curriculum issues, course pedagogy, classroom management, CS educational research, communication skills

1. INTRODUCTION

For a number of years, companies have told us that, although well prepared technically, our graduates lack skills needed to work in groups. At first we attempted to respond to these criticisms by introducing project courses and requiring group work on assignments in other courses. Unfortunately, these changes didn't solve the problem. Not only do we hear the same refrain from industry, but we also see a decrease in technical competence as some students coast through courses on the efforts of others in their groups.

*This work was supported by the National Science Foundation under grant EIA 008625. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

We initially proposed a technical solution in the form of a system to support development of the basic skills important to effective group work [8]. Our idea was to make it possible to place students into a simulated large project in order to control their dependence on their peers. Initially, each student would develop a distinct module and combine it with code developed by the instructor. Later, the instructor's code would become inaccessible and students would have to rely on modules developed by other students.

In order to obtain a baseline against which to measure the effect of this system, we began a process of ethnographic observation and in-depth interviews of students in three undergraduate courses (Data Structures, Principles of Programming Languages, and Introduction to Compiler Construction). What we learned from this process caused us to question whether a technical solution to the problem was appropriate: The students had an inherent bias against collaboration, and this bias was reinforced by the way in which assignments were posed and assessed. In order to improve the students' group work skills, we needed to avoid that reinforcement and instead reduce the bias.

In this paper we first make some observations on the nature of group work, arguing that collaborative skill is the ingredient our advisory boards find inadequate. We then present our findings about the existing student culture. These show a strong bias against collaboration, which cannot be overcome simply by providing opportunities for group projects. Finally, we present three interventions that we have used successfully to improve collaboration by modifying aspects of the academic culture.

2. TACTICS FOR GROUP WORK

When a group of people is given a task, there are four obvious tactics to try:

Sequential segmentation: I work on it for a while, then pass it along to you.

Parallel segmentation: We break it up and everyone does a piece.

Natural selection: We each carry it out and then choose the best result, or we choose the best person and let them do it.

Collaboration: We interact closely during the task.

In each of the first three cases, the members of the group are effectively working alone. Working alone is our students' preference (Section 3.1), and therefore whenever they are

given a task they actively employ one or more of those three tactics regardless of the intent of the professor. After all, they reason, who cares how it gets done? The solution is *graded* according to how well it solves the problem!

When we were explaining this to one of our colleagues, his comment was "In my experience industry uses parallel segmentation, so why are you worried about improving collaboration?" For all tactics except collaboration, technical skill will carry the day. According to our advisory boards, our students are well-prepared for technical challenges. Thus, if our colleague's comment were correct, we would not be hearing complaints from industry about students' ability to work in groups. We have to conclude that *collaboration* is the area of the deficiency; the results presented in Section 3 show the nature of this deficiency.

The reason that a deficiency in collaborative skills shows up as an "inability to work in groups" is that collaboration is crucial to a group's success no matter which of the four tactics that group claims to use. In each of the first three cases, the group has to come to some agreement about (or some authority has to dictate) how the members will work alone. Without that agreement (or dictate) the tactic cannot work.

Another reason that collaboration is important is that it avoids the third order of ignorance [2]: It is a process by which people can uncover things that they don't know and don't know that they don't know. Because it is impossible to specify a project completely, one member of the group will stumble over some questionable points when working alone, while another will stumble over different ones. Collaboration is required to share these experiences, ensuring that everyone understands that there are holes and how those holes will be filled.

Participation in collaborative discussion groups also exposes the *process* of thinking about course material. Members see how others think in addition to being exposed to a variety of possible conclusions. They learn how to make useful judgments more quickly than do people studying the same material individually, and the judgments of the group are usually better than those of the individual [1].

Thus it behooves us to improve our students' collaboration skills even in the absence of pressure from our industrial advisory boards. Listen to one of the students in Introduction to Compiler Construction:

> "I think that I have a good understanding of the material because I've worked with other compiler tools before, but as far as the tool that helps me pass the class, it's the other people in the class. I absolutely rely on the four or five people that you can always count on to do an assignment together. I couldn't do well in the course without them."

3. STUDENT CULTURE

An *occupational community* has been defined as "a group of people who consider themselves to be engaged in the same sort of work; whose identity is drawn from the work; who share with one another a set of values, norms and perspectives that apply to but extend beyond work related matters; and whose social relationships meld work and leisure" [13]. Clearly, university students belong to an occupational community according to this definition.

Occupational communities create and sustain work cultures involving task rituals, standards for behavior, and routine work practices. If the culture of computer science students fostered the ability to work in groups, then we would not be hearing that our graduates are deficient in that area; obviously there is a cultural mismatch [5].

In order to address this mismatch effectively, we need to understand the culture that characterizes the students' occupational community [15]. During the course of our three-year project, we have placed trained ethnographic observers in two classes each semester, and carried out more than 130 student interviews. Research assistants in the Department of Communication who have no connection with the CS Department were responsible for all of these tasks, and participants were assured of anonymity.

Our interviewers followed a protocol that was designed to elicit information about the students' background in computer science, experience in industry (if any), attitude toward group work, and approach to their classes. Questions were open-ended, and the interviewers encouraged the students to talk freely. Interviews lasted from 45 minutes to 2 hours. Analysis followed a modified version of Bantz' [3] procedure.

The observations and interviews have helped us to understand the behaviors underlying the students' occupational community. The remainder of this section summarizes the more important findings; see [12] for a complete analysis.

3.1 Preference for working alone

Participants in our study expressed an overwhelming preference for working alone. Their primary justification was that they wanted to bear the sole responsibility and get the sole credit for their work. This is important because they consider every assignment to be a "product" [6]. In other words, it is an opportunity to demonstrate to the professor that you can get the "right" answer and thus be rewarded with a good grade.

The preference for working alone was independent of prior experience in group work, and also independent of industrial experience. Participants mentioned a number of other common reasons for avoiding group work, such as not wanting to deal with interpersonal problems and having to pull along less competent group members, but they were all subordinate to the need to control one's own work.

This behavior is not surprising, given the importance of a student's grade-point average in campus life and its use as a first filter for employers seeking interviewees.

3.2 Procrastination

Virtually every student told stories about procrastinating on assignments. This ritual was often cast as a calculated risk, in which the participant knowingly put off starting the assignment but was able to complete it successfully because of their ability. Tying procrastination to confidence was a common theme, but in some stories the outcome was not successful because the student had overrated their competence or underestimated the magnitude of the task.

Participants did not provide any clear explanation for why they procrastinated, although there can be benefits other than the boost to one's ego from "pulling it off": More information about the assignment may become available from peers who have tried it, or the professor may give some last-minute hints in class in answer to questions. Changes may

be made to the assignment itself, thus invalidating early efforts or making them redundant.

Procrastination hinders collaboration because it does not leave time for discussion.

3.3 Experimentation

Another common set of stories concerned the students' approach to assignments. Although faculty members stressed the need to understand a problem before trying to solve it, most students chose to ignore this advice. "Diving in" and "tinkering" were very common words in the vocabulary used by the students to describe the way they tackled their assignments.

Careful study of programmers at all levels of development [10] shows that the distinguishing factor is the number of "patterns" a person has at their disposal. Thus an expert might recognize a particular problem as a sorting task, where a novice sees only a collection of data that must be summarized. Seen in this light, the "tinkering" ritual is consistent with most homework assignments: Because the intent of an assignment is to help the student to solidify their knowledge, it is unlikely that they will already have applicable patterns. Because of their lack of patterns, the only way that they can really *understand* the problem is to *solve* it!

Group discussion is a strategy that one can use to increase the number of patterns at one's disposal, because each member of the group comes to the discussion with a different set of patterns. The shared experience of the group can help to direct the experimentation process so that it is more efficient.

3.4 Disregard for process

In general, the students ignored process. They regarded all process as being laid down by the faculty, rather than engendered by particular tasks. Thus they felt quite safe in ignoring it, stating that as individuals they had more flexibility in their approach to the problem at hand. They believed that this flexibility allowed them to complete the assignment in the most efficient way and get the best possible grade.

Unfortunately, the students are right in their observation that most of the tasks they are assigned can be carried out in complete ignorance of process. They might take longer to complete, and the results might not be of the highest quality, but often the differences are small enough that the effort needed to master a process does not appear to pay off. This is an interesting consequence of the fact that students see assignments as products rather than opportunities to learn.

Collaboration is difficult when everyone has their own idea of the most effective way to work.

3.5 Combativeness

Most computer science students boldly display their own opinions, and disqualify the opinions of peers. The stories they tell show that they believe that the material in the curriculum is easier for them to master than it is for others. When describing particular assignments that they have completed, they use words like "challenging". In contrast, they use words like "difficult" to describe their peers' opinion of those same assignments.

Such behavior is often a consequence of working alone: The world appears to be simple and understandable, because one's own view is rather restricted. Ideas that are unfamiliar, and not immediately clear, seem to have little relevance. Thus collaboration is difficult because there is little common ground on which to build shared understanding; discussions are reminiscent of the six blind men with the elephant.

3.6 Unwillingness to support others

Many participants described situations in which they were unable to obtain technical and emotional support from their peers or professors; others told of refusing to provide such support themselves. Interestingly, we noted no gender difference in this behavior.

Refusal of support is rationalized by statements along the lines of "if I help them, they won't get the benefit of working it out for themselves". It is promoted by explicit plagiarism policies that try to set boundaries on cooperation and enforce them with draconian penalties. Our data indicates that such policies are also important in shaping the preference for working alone.

Supportive behavior is a very important factor for team success [11], and therefore a refusal to provide or allow it is damaging to collaboration.

3.7 Absence of passion

Our data failed to uncover any particular passion in the participants, although this may be due to the selection of courses that we chose to study. Data Structures is a required second-year course, regarded by the faculty as an opportunity to instill further programming skills and familiarize the student with common data structures like stacks, queues, and trees. Principles of Programming Languages is also required, and is intended to give the student a basic understanding of concepts like structure, binding, parameter passing, and type systems with the goal of easing the task of learning new languages or selecting an appropriate language for a particular problem. Introduction to Compiler Construction is a fourth-year elective in which the students apply language concepts using tools to build a compiler. None of these courses seems to hold great appeal for the students, who regard them as chores to get through on the way to a degree.

Collaboration, especially at first, can be hard work. Many students told stories of interpersonal problems that derailed their group projects. Without shared passion, it is very hard for a group to get beyond such problems.

4. FOSTERING COLLABORATION

In order to improve the students' collaborative skills, we need to change some of the characteristics of their occupational community. This cannot be done by teaching a course in group work or telling them to work in groups to solve a problem. It has to be done by understanding the enculturation process, and establishing conditions that favor development of a collaborative culture.

This section describes three specific interventions that we have used successfully to influence the students' enculturation.

4.1 The conversational classroom

As noted above, the students overwhelmingly preferred to work alone because collaboration forces them to yield authority over their work to others. Bruffee [4] discusses

that aspect of collaboration, and points out that collaboration also forces one to *accept* authority over the work of others. Students must be shown that collaboration offers advantages over working alone that outweigh the perceived disadvantages of yielding authority and accepting it.

Professors' behavior is an important component of the enculturation process, and thus professors ought to use a collaborative approach to the curriculum if they want the students to take collaboration seriously. The *conversational classroom* [14] is a strategy for demonstrating the advantages of collaboration.

In the conversational classroom, the professor facilitates a discussion of the material relevant to a particular class session instead of giving a lecture over that material. Thus the professor is yielding authority over the flow of information to the students. They must accept this authority by preparing carefully, contributing to the discussion, determining how deeply to explore each issue, and deciding when to switch topics.

We have used the conversational classroom technique very successfully with classes of 80-120 students. Relative to traditional lectures, it has improved students' performance as well as improving their collaborative skills. We have observed an increase in effective use of study groups by participants, and increased willingness to take part in classroom interaction in other classes.

4.2 Group decision-making

One of the recurring stories told by students to illustrate why they don't want to work in groups is the "failed decision": The group talks around and around a question without coming to any conclusion and finally the narrator gives up on the group and solves the problem independently.

Two characteristics of the students' occupational community, their bold display of their own opinions and their willful disregard of process, play into this scenario. Because each participant is advocating their solution and disqualifying the solutions of others, little consideration is given to the criteria on which the decision should be based. Processes for effective group decision making are available, but the students don't believe that process is necessary. We need to change that perception.

After several unsuccessful attempts, we have developed a viable group decision making exercise illustrating the effective use of process in a situation where that process is dictated by the problem. This exercise helps the students to move away from their insistence on individual freedom in all cases.

To be a compelling example, the problem must be complex enough to present nontrivial choices and a rich set of criteria. It is also vital that the decision be *real* in the sense that it must strongly affect the students' future in the course. Finally, students should come into the exercise with some specific opinion. (This enables us to contrast a focus on criteria with the usual focus on outcomes discussed in Section 3.5.)

The problem we chose was selection of an abstract syntax tree (AST) for a compiler project. That tree is the central data structure on which all of the computations implementing semantic analysis and translation are carried out. Its structure determines the complexity of those computations, and the ease with which standard modules can be applied. All of the students in the class were required to agree as a group on a single AST, which was then used as the basis for all of their future assignments.

After they had gained some experience with semantic analysis, each student was asked to submit a specification of an AST for a language they had not previously seen. These proposals were made available to all students. Each student was asked to select one of the proposed trees (their own or someone else's) and use tools [9] to generate a tree constructor. At this point the students had enough information to make an informed choice.

The next class period was devoted to a discussion, facilitated by an outsider, in which the class as a whole (including the professor as a participant) developed a set of criteria for choosing an AST. Notice that the discussion was *not* concerned directly with choice of a tree. The object of the exercise was to show that the most important aspect of the decision was the *criteria*. Those criteria were based on principles that the students had learned in the course, and that were relevant to selection of an AST.

After class, the facilitator entered the criteria developed during the discussion into a web-based decision support system [7]. Over the next two days, each student was required to enter a weight for each criterion reflecting their individual judgment of the importance of that criterion for the selection. They were also required to evaluate each of the proposed trees with respect to each of the criteria, and enter a number that reflected their judgment of how well that tree fit that criterion.

At the end of the evaluation period, the decision support system selected the highest-ranking tree as the basis for the compiler that the students would construct over the remainder of the semester. Changes to the tree were permitted in the light of subsequent experience, but *only* if everyone agreed and *only* if the change had the effect of raising the tree's score according to the agreed-upon set of criteria.

4.3 Assignment devaluation

There is always pressure from the students to increase the weight of the assignments, because the assignments "take so much time" to complete. If the weight of an activity in the final grade is proportional to the time a person spends in that activity, then certainly that activity must be considered as a product. Professors therefore reinforce the view of an assignment as a product by weighting assignments heavily in the final grade for the course.

When assignments are heavily weighted, there is a tendency on the part of the professor to actively discourage collaboration among the students. The fear is that heavily-weighted collaborative assignments will lead to an erosion of standards as unqualified people slide through on the efforts of their more-qualified collaborators.

We have confronted these problems directly in several courses by significantly decreasing the weighting of the assignments in the total grade. This change was *not* accompanied by any change in the assignments themselves. Instead, we have devoted time and effort in both the web material for the courses and the lectures to emphasize that the assignments are *not* regarded as products, and the only reason that they are given any weight at all is to provide an incentive to do them.

Each assignment includes a description of what we expect the student to learn, and a "background" section linking that assignment to appropriate readings and outlining any

process engendered by problems of this type. If we want the student to follow a specific process, we include questions for the student to answer about intermediate results. We limit procrastination by having assignments due weekly. (This allows the students to establish a rhythm for scheduling their time.) Longer projects have deliverables at weekly intervals.

We explain the concept of the orders of ignorance [2], and argue that assignments are a mechanism to expose issues that the student doesn't understand and doesn't know they don't understand. Students are encouraged to work collaboratively, but we usually do not require them to do so (in some courses lack of specialized equipment forces collaboration). We talk about the low weight of the assignments, relative to the time the students spend doing them, and liken that time to time spent reading, attending class, or visiting the professor in their office. The low weight means that students can try unfamiliar techniques, and if they don't work out then little is lost.

This strategy has been largely successful. Interviews show that students still spend a lot of time in the lab, but they describe it in terms of educational value rather than time versus credit. There is extensive collaboration, which spills over from assignments to study groups, and the aggregate grades are higher even though the lion's share of the assessment is an individual examination [14].

5. CONCLUSIONS

Both educational research and our industrial advisory boards encourage us to improve our students' collaborative skills. At first we attempted to do this by introducing group assignments and project courses, but we still heard complaints from those who hire our graduates.

Careful examination of the culture of our students' occupational community has revealed task rituals, behaviors, and work practices that inhibit collaboration. Normal classroom lecturing and assignments reinforce these inhibitions, making it very difficult to improve the situation.

In this paper we have suggested three general strategies for moving student enculturation toward increased collaboration:

- Replacing traditional lectures with more open discussion
- Incorporating collaborative processes into technical assignments
- Emphasizing the instructional nature of assignments (as opposed to treating them as products)

A key insight is that professors must relinquish their role as masters of all they survey. After providing a syllabus laying out the structure of the material and appropriate readings, and a set of assignments to give the students the necessary practice, a professor should act as a facilitator of discussion and a source of expertise in both factual and process areas.

The results of our longitudinal study show that, although these strategies may initially be strongly resisted by the students, they lead to better performance and increased student satisfaction. Over time, as students who have experienced them move through the curriculum, resistance decreases and the benefits begin to accrue earlier in a course due to carryover from previous courses.

6. REFERENCES

[1] M. L. Johnson Abercrombie. *The Anatomy of Judgement*. Hutchinson, London, 1960.

[2] P. G. Armour. The five orders of ignorance. *Communications of the ACM*, 43(10):17–20, 2000.

[3] C. R. Bantz. *Understanding Organizations: Interpreting Organizational Communication Cultures*. University of South Carolina Press, 1993.

[4] Kenneth A. Bruffee. The art of collaborative learning: Making the most of knowledgeable peers. *Change*, 26(3):39–44, 1994.

[5] Louis L. Bucciarelli and Sarah Kuhn. Engineering education and engineering practice: Improving the fit. In Stephen R. Barley and Julian E. Orr, editors, *Between Craft and Science*, Technology and Work, pages 210–256. Cornell University Press, 1997.

[6] G. Button and W. Sharrock. Project work: the organisation of collaborative design and development in software engineering. *Computer Supported Cooperative Work*, 5:369–386, 1996.

[7] Lynn Robert Carter. The personal engineering process. URL `http://deming.colorado.edu/`.

[8] Amer Diwan, William Waite, and Michele Jackson. An infrastructure for teaching skills for group decision making and problem solving in programming projects. In *Proceedings of the 33rd ACM Technical Symposium on Computer Science Education*, pages 276–280, New York, 2002. ACM Press.

[9] Robert W. Gray, Steven P. Levi, Vincent P. Heuring, Anthony M. Sloane, and William M. Waite. Eli: a complete, flexible compiler construction system. *Communications of the ACM*, 35(2):121–130, 1992. URL `http://eli-project.sourceforge.net/`.

[10] Robin Jeffries, Althea T. Turner, Peter G. Polson, and M. E. Atwood. The processes involved in software design. In J. R. Anderson, editor, *Cognitive Skills and their Acquisition*, pages 254–284. Lawrence Erlbaum Associates, Hillsdale, NJ, 1981.

[11] Frank M. J. LaFasto and Carl Larson. *When teams work best : 6,000 team members and leaders tell what it takes to succeed*. Sage Publications, Thousand Oaks, CA, 2001.

[12] Paul M. Leonardi. The mythos of engineering culture: A study of communicative performances and interaction. Master's thesis, University of Colorado, Boulder, 2003. URL `http://www.cs.colorado.edu/~pltools/pubs/Leonardi.pdf`.

[13] John Van Maanen and Stephen R. Barley. Occupational communities: Culture and control in organizations. In Barry M. Staw and L. L. Cummings, editors, *Research in Organizational Behavior*, volume 6, pages 287–365. JAI Press, 1984.

[14] William M. Waite, Michele H. Jackson, and Amer Diwan. The conversational classroom. In *Proceedings of the 34th ACM Technical Symposium on Computer Science Education*, pages 127–131, New York, 2003. ACM Press.

[15] K. Weick. *The social psychology of organizing*. Addison-Wesley, 1979.