

Peer Evaluations for Team Projects in Computer Science Courses¹

David L. Coleman
Henry C. Thibault

Abstract

Academic and industrial situations often differ in the emphasis placed on working together—students must be evaluated individually; employees work together toward a common goal. College students need to experience group dynamics to learn the benefits and pitfalls of working in a team environment. Students that are not contributing to their team's efforts are plagiarizing the work of other team members. To mitigate and penalize the effects of non-contributing members, each team member's overall project score is augmented with a peer evaluation. This paper presents several problems and benefits associated with peer evaluations. Two example peer evaluation techniques are presented. In addition, the paper presents several team formation guidelines that can minimize team conflicts.

Introduction

Academic and industrial situations often differ in the emphasis placed on working together. In an academic setting, we must evaluate students as individuals to distinguish between students who get A's and those who get B's, etc. A student who claims credit for the superior work of another student would unfairly earn a higher grade. However, in an industrial setting, teamwork is the norm. Backstabbing while climbing the corporate ladder aside, the general goal is to work as a team—building on each other's work—to get the job done.

College students need to experience group dynamics and to learn the benefits and pitfalls of working in a team environment. The benefits include division of labor and alternate viewpoints on how to complete a project. The pitfalls include the overhead of team communication and coordination, and the potential for some team members to

¹ Republished with permission of Small College Computing Symposium, Inc. from Proceedings of the 28th Annual Small College Computing Symposium, p.34-42. April 21-22, 1995, Augustana College, Sioux Falls, SD.

not contribute their fair share of the work on the project. Deliberately non-contributing team members are guilty of a form of plagiarism. They claim part of their teammates' work as their own by receiving the same overall score for the project. In fact, their failure to contribute often results in undeservedly low scores for their more industrious teammates.

However, under-contribution may not be deliberate. Several factors may combine to make it impossible for a given student to contribute a fair share. Sometimes students form team with schedule conflicts that prevented one or more members from contributing as effectively as they might have. Also, controlling personalities occasionally made it difficult for others to contribute. In Section 2, we discuss several factors that should be considered when forming teams.

To mitigate the effects of non-contributing team members and controlling personalities on the overall score of the team, and to penalize the offending team members, each team member's overall project score is augmented with a peer evaluation score. In a Software Engineering course, Coleman varied the frequency of evaluation over a period of three semesters—at the end of course, at the end of each project phase, or every two weeks. Thibault, in other courses with smaller projects, administered peer evaluations at the end of the projects. However, he applied various techniques during the team selection phase to reduce the likelihood of difficulties in group dynamics. Section 3 describes the peer evaluation techniques we use in various computer science courses. Included are examples of the peer evaluation forms used.

In Section 4, we discuss team performance and team dynamics as reflected in peer evaluations. Several successful team leadership styles manifest themselves. However, pathological dynamics also occur, and we note ways of avoiding trouble. Section 5 presents and discusses benefits that we have observed, both for students and instructors.

Team Selection

Team formation can be accomplished in a variety of ways (Scott, Tichenor, Bisland & Cross, 1994). The major division of these techniques is based on whether the instructor chooses the teams or if students self-select them.

An instructor could assign students to teams randomly. Random assignment does not necessarily mean the use of a random number generator, but could simply be based on position of the student in the class roster—every Nth student is assigned to team N or the first M students are team one, next M students team two, etc. This is clearly the easiest method for the instructor, but is fraught with problems for the students.

Factors an instructor should consider in team formation include:

Related Course Work: For upper division courses, students do not necessarily have a homogeneous background of prior courses. Teams could be formed with the goal of providing teams with a homogeneous background of prior courses. A measure of homogeneity used to form teams could also be weighted on the grade

received in prior courses. Thus a student on one team that received an A grade in course X might be balanced by two students on another team that each received a C grade in course X.

A potential pitfall of team homogeneity occurs when a particularly strong team member limits the academic growth of weaker team members. This could be an argument for forming team from homogeneous members, resulting in heterogeneous teams. In this case the instructor must be sure to evaluate the team in a heterogeneous fashion.

Group Dynamics: Individuals in a work environment can be classified into three types (Sommerville 89):

1. Task-oriented: Individual is motivated by the work itself.
2. Self-oriented: Individual is motivated by personal success.
3. Interaction-oriented: Individual is motivated by presence and actions of co-workers.

These classifications are not rigid, nor is a single individual confined to a particular class. Teams made up of members that are all task-oriented or all self-oriented tend to suffer from an overabundance of leaders. Teams with a task-oriented leader and a mix of other classifications may be the most successful, however, programmers tend to be task-oriented individuals.

Prior Knowledge: The instructor may already be familiar with some of the students from other courses. Knowledge about students' strengths and weakness, both academically and personality wise, could be used to guide team formation.

Student Schedules: An overriding concern in team formation is scheduling of work time. Projects that require teams to work together outside of class necessitate coordination of students' schedules. The scheduling factor is probably the most important factor to be considered in team formation, whether formation is random, instructor assigned, or self-selected by the students. In a real-world environment, employees' schedules are relatively similar and are under the control of management. Students' schedules are obviously dependent on a variety of factors that differ from student to student and cannot be controlled by the instructor.

The most likely scenario is that students self-select teams. However, students should be made aware of the above factors, especially scheduling outside class meeting times for the team. Thibault provides students with the list of guidelines from Figure 1. An auxiliary advantage to having students self-select teams is that the instructor is less likely to be blamed when a team fails to meet deadlines due to internal conflict on the team.

(figure 1 here)

Evaluation Techniques

The goal of the evaluation process is to mitigate the effects of poor team formation. Evaluations allow students to express grievances, reward hard work, and penalize slackers. This section presents Coleman's and Thibault's evaluation processes and rationales.

Coleman's Evaluation Process

Coleman has varied the use of team evaluations in the last two semesters he has taught Software Engineering. In prior years, team evaluations were administered only at the end of the semester. During the last semester, team evaluations were required at the end of each project phase. In the previous semester, team evaluations were required at the end of every two weeks.

The goal of the increased frequency was twofold. First, end-of-year evaluations tended to reflect the performance of team members only during the final crunch phase of the project. Team members who contributed substantial work during earlier project phases could be unfairly penalized for smaller contributions at the end of the semester. Second, end-of-year evaluations did little at the beginning of the semester to motivate indolent team members to contribute to the project.

Coleman's team evaluation form is given in Figure 2. Coleman's evaluation is based on the principle that team members should not evaluate themselves. This works well only if team size is sufficiently large to prevent one or two members from ganging up on another member. Team sizes for Coleman's Software Engineering course were fixed at four or five students per team. In addition to an evaluation of their teammates, each team member submitted a time sheet describing their time expenditures on the project. Although the time sheets were not used for grading, they could be used to authenticate peer evaluations under the assumption that time spent measures contributed effort. However, advertising this use results in inflated times and invalidates the assumption.

(figure 2 here)

To calculate a final project score each team member received the average of the scores submitted by their teammates for each evaluation period. The final evaluation score was the sum of these averages. The final evaluation score contributed 25% to the total points received for the term project. The other 75% of the points on the team project would be the same for each member of a particular team.

Thibault's Evaluation Process

Figure 3 is Thibault's Peer Evaluation Form. Here are some important points about the form and its administration.

- Students receive the form at the beginning of the project. Thus, they are aware of peer evaluation standards throughout the project, and strive to earn the approbation of their teammates.
- Students have the option to not submit the form. They thereby implicitly acknowledge the truth of comments made by teammates, receiving the team's project grade modified by their peer rating. In each course, a few teams usually submit no forms, thus agreeing to equal scores.
- Students decide what project phases to list. The rationale behind this is to increase student acceptance of the form's fairness, and to reveal the student's perception of the project.
- Students say as much as they like on the back of the form. Besides giving additional information not otherwise available, this seems to defuse some problems by serving as a vent for frustrations.
- Students evaluate themselves as well as their teammates. It was initially a pleasant surprise to see that students regularly assign higher marks to some teammates than to themselves, and that there is much agreement within teams as to members' performance.
- The evaluation process necessarily incorporates subjective elements. How consistent are replies from all members of the team? If there are significant differences, (a) did they have any bearing on the quality of the work of individuals? Mere disagreement does not justify adjusting grades. (b) What is likely to be the truth? What do we know about each student from past experience? What is consistent with other evidence?
- Students whose performance is clearly outstanding, especially if acknowledged by teammates, get bonuses, perhaps 5% to 20% of the project value depending on the extent of their extra contribution. Other team members receive no penalties, unless it seems clear they had a good opportunity to contribute but did significantly less than others.

(figure 3 here)

There is no substitute for an instructor's judgment and knowledge of students. However, peer evaluations provide hard evidence in case a student challenges a grade. This has not occurred in eighteen years of teaching, involving thousands of team projects at two universities. A student occasionally requests clarification about a grade. It is often difficult to reveal all computations without violating teammate confidentiality, especially in two-person teams, so students learn in advance that they will not see many scores in heavily project-oriented courses. In some cases, students are told little beyond overall numeric scores and letter grades for the course. Revealing even non-peer-influenced grade components sometimes can enable a student to "reverse-engineer" peer scores from final averages.

Observations About Team Performance

We have observed several team dynamics, usually reported by team members on evaluation forms. Some occur in successful teams; others in problem teams. The type of team dynamic is usually the result of a tacit agreement between the students.

It is important to know that there are several successful ways for teams to operate. If instructors and students know that there are several possible paths to success, teams will be more likely to operate according to a model appropriate for the personalities involved.

There are also pathological team dynamics. Knowing that a peer evaluation awaits is a deterrent to many of these undesirable team styles, but all of them still occasionally occur. We describe them here so they can be anticipated and avoided. We also recommend that instructors conduct informal reviews to detect and correct these problems before their consequences become severe.

Successful Team Dynamics

Single Leader: In this type of team, one strong team leader emerges to control the project, making most decisions and assigning tasks. This is usually a natural consequence of team personalities, but sometimes a person who is normally a leader chooses to be a follower, perhaps because of other commitments.

Joint leaders: This is rare, but sometimes a decision-making group forms to direct team activities. Joint leadership would probably emerge more often with larger teams.

Phase leaders: Students often report, for instance, that one person led the design phase, then another directed implementation. This can be an especially effective style, utilizing varied expertise without placing excessive demands on one or two leaders.

Leaderless consensus: Some successful teams say they worked together on all phases without any single leader emerging. We doubt if the consensus is always as complete as reported; perhaps teams wish not to wash dirty linen in view of the instructor. However, this style is frequently associated with success.

Pathological Team Dynamics

The team formation guidelines in Figure 1 warn students of the existence and causes of pathological team dynamics. It also clearly establishes student responsibility for finding compatible teammates. This, along with informal monitoring reviews, prevents many of the problems listed below.

Control Freak: This perversion of the Single Leader style exhibits an unwillingness to delegate tasks, or an inability to accept work done by others. It results in surprisingly good final products if the Control Freak has the ability and time to do the whole project alone. However, there will be hard feelings. This style can sometimes arise if individuals with greatly differing goals and standards form a team, forcing the more ambitious member(s) to take over the whole project if they want it done well enough to satisfy them.

Freeloaders: Actual freeloaders, who have no intention of contributing to the team effort, are rare. The certainty of censure during peer evaluations usually prevents this. What does sometimes occur is that students may be hindered from contributing by schedule problems that make it impossible to meet often enough with teammates, or a Control Freak leader may prevent one or more teammates from contributing.

War: Sometimes, two or more strong individuals or groups have very different visions of how to do the project. Instructors need to interview teams to detect and resolve the misunderstandings that underlie such situations.

Conclusions: Effects and Advantages

We do not claim that our students achieve perfect group dynamics. We have, however, noticed a number of benefits:

1. Teams almost never break down.
2. More teams function smoothly, with far less need for "refereeing" by instructors.
3. Difficulties are easier to resolve when they occur.
4. Project results are of higher quality
5. Students are more interested and able to work in teams, bringing lifelong career benefits.

The methods we suggest are not difficult. In fact, they take less time than struggling with disputes and poor outcomes of team projects administered without them. Most significantly, though, the students achieve more, both in the short term and in career situations.

References

Scott, T. J., Tichenor, Jr., L. H., Bisland, R. B., and Cross II, J. H. (1994). Team dynamics in student programming projects. Twenty-Fifth SIGCSE Technical Symposium on Computer Science Education, 111-115.

Sommerville, I. (1989). Software Engineering. Wokingham, England. Addison-Wesley.

David L. Coleman, Assistant Professor
Henry C. Thibault, Associate Professor

Department of Computer Science
University of Wisconsin-La Crosse
La Crosse, Wisconsin 54601
(608) 785-6805
coleman(Ocsfac.uwlax.edu
thibault@csfa,c.uwlax.edu

Finding Good Teammates

You can save yourself much grief in this team project by being careful when forming teams. Please find teammates who are compatible with you in the following areas:

Schedule: If you can't find times when the whole team can meet, you will be miserable.

Interests: If you can't get interested in the same topics, you'll build a piece of junk.

Attitudes: If all of you want to be the boss, you'll eat each other alive.

Work Habits: If highly organized people try to work with "whenever" people, everybody goes nuts.

Goals for this Course: If you want to take a C and run, don't team up with perfectionists.

Note also:

It is better to have a mix of talents on your team than for all teammates to have similar strengths and weaknesses.

Figure 1: Thibault's Team Formation Guidelines

C-S 341 Log Sheets & Team Evaluations

Keep track of all time spent on the term project according to the following three disjoint activities. Record time in hours rounded to the nearest 1/4 of an hour for each activity. The amount of time you report will not be used for grading purposes, however, not reporting your times will count against you.

1. How much time did you spend in team meetings (not counting time spent reviewing)? This could include planning future meetings, dividing up the work load, or brain storming sessions. Call this "meeting time."
2. How much time did you spend working on tasks assigned to you? This could include joint tasks worked on with another team member, but not the whole team. Call this "working time."
3. How much time did you spend reviewing? This could include reviewing your work with other team members or reviewing work produced by another team member either individually or with other team members (including the whole team). Call this "reviewing time." Note, if you review your own work by yourself, count that as "work time."

Based on your own observations, evaluate the performance of the other members of your team during the last evaluation period. You do not evaluate yourself. Assume an average team member would receive a score of 20 points. Top performers would receive more, sliders less. There are two constraints your point distribution must satisfy:

1. The total points distributed to your team members equals 20 times the team size minus one (i.e., yourself),
2. The maximum score you can assign to any one person is 30 points.

For example, suppose you were on a 5 person team with the following other members: Sue Superstar, Ole Outstanding, Annie Average, and Ned Neverthere. You would have 80 points (4 times 20) to distribute. Your distribution might look like the following:

Sue Superstar	23
Ole Outstanding	23
Annie Average	20
Ned Neverthere	14
	<hr/>
	80 total

You should reevaluate your team members for each evaluation period and base your evaluation only on performance since the last evaluation. These scores will be used for grading. Be honest and fair as you would hope others will be.

Figure 2: Coleman's Team Evaluation Form

C-S 390 Teammate Peer Evaluation

Instructions: Evaluate each team member including yourself. Percentages should add to 100 across rows. Scores should not measure time and effort, but should consider the effect of the member's work. In other words, you are evaluating output, not input. This is also true in weak areas; high percentages apportion blame for bad work as well as giving credit for a fine job. Be fair—this evaluation is important to each team member's grade (and you might need teammates for other courses!), so be sure that both praise and blame are justly given. Also, I am suspicious of evaluations that are entirely positive, entirely negative, or without thoughtful added comments and explanations (use back of page).

Percentage Contributions:

Name: _____
(you)

Project Phase:

_____	____%	_____	____%	_____	____%
_____	____%	_____	____%	_____	____%
_____	____%	_____	____%	_____	____%
_____	____%	_____	____%	_____	____%
_____	____%	_____	____%	_____	____%

Overall: _____%

Like to work with again? (0=never; 5=eagerly) _____

Letter grade: _____

Letter grade for entire team: _____

Who, if anyone, emerged as team leader?

Use the back of this form to answer the following questions: What improvements would you like to make in your project? What features of your project are especially good? Is there anything else you want to say about the project?

Figure 3: Thibault's Team Evaluation Form