Department of Computer Science,
University of Otago

Technical Report OUCS-2004-15

Talking Past Each Other – Student And Staff Reflection In Undergraduate Software Projects

Author:

Stewart T. Fleming
Department of Computer Science, University of Otago

Status: Submitted to SIGCSE 2005 Technical Symposium on Computer Science Education. St Louis, Missouri, February 23-27, 2005

Department of Computer Science,
University of Otago, PO Box 56, Dunedin, Otago, New Zealand

http://www.cs.otago.ac.nz/research/techreports.html

Talking Past Each Other – Student And Staff Reflection In Undergraduate Software Projects

Stewart T. Fleming
Department of Computer Science,
University of Otago,
Dunedin, New Zealand
stf@cs.otago.ac.nz

ABSTRACT

Group projects are an important part of Software Engineering education. However, conflicts that arise from group work can affect overall class learning and performance. It can be difficult for teachers to fully understand the social context of these issues.

We explore the nature of self-, peer- and staff-reflection to identify and mediate issues within a class. We have used a protocol that encourages reflection to explore conflicts that arise from group work in a Software Engineering paper.

We have found a way to explore and mediate student impressions and expectations and to identify conflicts with staff expectations and course objectives. We present a lightweight and flexible approach for such investigations.

Categories and Subject Descriptors

Programming teams, Computer Science education, Collaborative learning.

General Terms

Human Factors, Management, Measurement

Keywords

Group Work, social learning, reflection.

1. INTRODUCTION

The importance of a substantial Software Engineering project executed by groups of undergraduate Computer Science students is widely recognized [5, 6, 7, 9, 12]. With group work, there are many issues relating to group dynamics, organisation and conflict that arise that have an effect on project outcomes. It can often be difficult for staff to penetrate the social context that groups create for themselves in order to understand the issues that groups face during their projects. We focus on the issues relating to group projects and the role of reflection in discovering the true nature of project circumstances. We discuss a protocol that we used to guide reflective activity in our teaching.

In our institution, we face a continual problem in retaining students at postgraduate level in Computer Science. Our undergraduate students are motivated to complete their studies as quickly as possible and find work. At the same time, we are finding that more and more educational initiatives within New Zealand are encouraging us to work more closely with industry. In teaching Software Engineering, we find a natural focus and influence on our teaching to account for both of these trends. We view a course in Software Engineering as involving more *learning* than *teaching*. We view student involvement in such a course as "…learning to be an engineer, by practising to be an engineer…" and for an engineer to "[walk] the bridge between science and technology" [2], balancing engineering activity with the application of strong scientific principles [16].

Providing a simulation of the "real world" of Software Engineering is again a common approach throughout Software Engineering Education. This simulated project environment provides the grounding in real tasks that seems to be important for adult learning [15]. There is a strong resonance here with constructivist theories of learning, where we find an emphasis on learning by developing knowledge through practice and working collaboratively to solve problems [11].

Group work encourages the development of personal skills and responsibility and the development of inter-personal relationships [4, 13]. Practical experience in engineering and project management can only be acquired in a large project setting [9]. The question arises as to whether academic software projects really are simulating the real-world environment [8]. That is, when we engage in this process of simulation, do we provide an environment in which real-world practice can take place, or do we provide a simulacrum in which activities and outcomes relating to Higher Education principles related to group work occur and emerge? We lean more towards this latter approach in our course designs and delivery.

1.1 Course Organisation

We teach Software Engineering as a full-year paper, involving 26 weeks of full-time study with two formal lectures each week. At our institution, the study year is divided into two semesters of 13 weeks each, with a 1-week break in the middle of each semester and a 4-week break between semesters. Institutional guidelines to students indicate that they are expected to spend approximately 6

hours per week on this paper, including formal class time, preparation for class and project work. Assessment for the paper is divided into 60% for a 3-hour final examination and 40% for the full-year project.

Our philosophy on teaching this paper is based on the notion that simulating the engineering environment provides good preparation for real engineering – we strongly believe that students should be able to do things in a practical way and hence they can learn the theoretical underpinnings in a stronger sense through this practice. We arrange course material in both semesters using a "just in time" strategy to cover topics that are directly relevant to imminent project activities.

The selection of project topics is done to emphasise useful work. We attempt to make topics as meaningful as possible, selecting an application area for which there is a direct need either within the institution or, more commonly in recent years, in an external/industrial context. By the time they enrol in this paper, the majority of students will have engaged in group work in at least one other Computer Science paper.

In setting projects, we look for topics that are realistic, doable within the resources available and institutional guidelines and which do not require specialised knowledge beyond programming and data structures that we assume students to have as prerequisites. Strictly speaking, we select topics that are "not quite doable" with the resources available. We do this deliberately so that students can encounter time-related constraints and evolve risk-management strategies to suit. Groups are made up of 4-6 students, selected randomly, and we make no attempt to balance groups based on gender, age or previous experience.

We select a combination of "simulated customer" or a "simulated project manager" techniques to assist students with the management of group projects, depending on the nature of groups themselves or the project at hand.

Simulated customer – an academic staff member acts in the role of the customer, meeting with groups, setting requirements for the project, receiving and testing deliverables, entering into different types of communication with the group, causing various events and risks to occur in the project.

Simulated project manager – an academic staff member acts in the role of "boss", meeting with groups to discuss project schedules, goals and activities. They act as manager for the group and can cause variation to goals, schedules and planned milestones. This arrangement is much in the style proposed by Ford & Morice [8].

With the institutional guidelines as above, we expect each student to undertake 3 hours per week on the project, for a year-long total of 84 hours. We believe that this is on the low side; in New Zealand, Brown [3] provides examples of workload ranging from 10-40 hours per week for student projects and comparing internationally we find estimates of project work ranging up to 500 hours of work [5] per semester per student.

1.2 Conflicts of Expectations

Through evaluations of the paper as a whole and student-based evaluation of staff using standardised questionnaires, project meetings and Departmental-level student representations over the last few semesters, we have become aware of a number of conflicts between staff and student expectations.

First, while we may believe that the expected workload is realistic and indeed lower than that set in other institutions, clearly our students do not. High or excessive workload has been the commonest cause of complaint for this paper in the last few years. Teaching staff find this frustrating since they feel that they have delegated control over workload to students within the group project setting.

The second major conflict that we have encountered is with our expectation of project feasibility. Our point of view is that not all of the project requirements will be met and we are generally not disappointed when project groups fail to meet them. That is, we *expect* failure to meet all of the objectives, but we also expect groups to specify which ones they *intend* to achieve. On the other hand, students appear to believe that all project requirements must be met and generally apply one of two strategies for coping – dropping the paper or doing lots of work. Neither are intended outcomes from the teaching viewpoint.

The third conflict that we encounter is with students' expectations regarding assessment. We adopt a flexible strategy to assessment of project work, based on the principle that we expect engineering activities to be performed in the course of the project. Our primary form of assessment is whether or not such activities were performed and how they were done. Students are not sure about assessment; they want clearer guidelines and marking schemes.

It is important to note that we do not regard conflict as something negative and to be avoided. Rather we look at conflict as an opportunity to investigate our teaching practice further, to make adjustments and refinements where appropriate, but more so as a spur to seek better communication with students so as to find a better match between our expectations and theirs.

1.3 Contributions of this paper

One reaction would be simply to blame the students for misaligned expectations. Such an approach is not very satisfactory, since it damages morale and tends to lead to students dropping the paper, hence missing out on a valuable educational experience. A better outcome, one that is intended to improve teaching practice, reflects more deeply on student activities and difficulties and looks at how their activities can be better aligned with the objectives of the course.

Facilitating understanding – influencing project outcomes by encouraging students to learn is a role that we can perform as teachers. If students are not engaging in engineering activities, then we need to be asking why. *"Why aren't they learning? How can I get them to be relevantly active?"* [1, p12].

What we set out to do was to find better ways of investigating the issues and improving communication with students. Instead of recommending any one management style, we regard the facilitation of a discourse with students as more important. We report here on the use of a technique of self-, peer- and staff-reflection that gives a clearer picture of the actual factors at work and lets us focus on managing the real conflicts that we found.

1.4 Related Work

Since we regard dialogue with students as an important factor in the overall effectiveness of teaching, we wanted to find techniques with which to investigate student attitudes and experiences.

Upchurch & Sims-Knight [18] have proposed that a curriculum model based around a portfolio of student work and the promotion of self-reflection by students on learning provides a valuable way for students to organise and maintain their learning.

We believe that a constructivist approach will be fruitful in a social, group-oriented course. In this approach, the notion of self-assessment on the part of students and their peers becomes central to their development. We find a role for critical reflection at the heart of the curriculum [17, 10] and use a technique [14] that encourages the important issues to arise from the individual students and to be qualified by students and staff as a group. Note that we are using reflection here for evaluation and not as part of assessment for coursework [4].

2. THE ROLE OF REFLECTION

As teachers, we are particularly interested in evaluation statements that relate to teaching objectives and/or administration.

1. Simulated engineering environment.
2. Flexible assessment based on engineering activity.
3. Doable, relevant group projects.
4. "Just In Time" delivery of project-related material.

We were also interested in discovering whether or not workload issues were as serious as indicated through other channels. However, we wanted to understand the issues as expressed by students and to avoid the situation where one vocal member of a class could affect our impression of the problem.

2.1 Protocol

We adopted a protocol for course evaluation suggested by Rowland [14] that encourages self-reflection on the part of students and provides validation through the use of peer- and staff-reflection. This technique has the potential to discover situations where the expectations of the student and the teacher come into conflict.

In this protocol, the evaluation statements arise from the students themselves in a process of self-reflection. Their peers then have an opportunity to evaluate the statements made by others and a consensus (or otherwise) emerges. Finally, the evaluator has a role to play in stimulating a discussion to investigate the critical issues that emerge.

This is a student-centred approach to evaluation. This circumvents a problem found in conventional student evaluation questionnaires, where the evaluator can never be sure that the statements with which students are expected to agree or disagree are actually meaningful to them. By allowing students to define the agenda for evaluation, issues can emerge that the teaching staff never considered at the outset of the course.

2.2 Administration of the protocol

We chose a normal lecture session towards the middle of the course year for the evaluation. A small group of 11 students (approximately 30% of the class) attended this session.

1. Students were provided with a handout to explain the intentions of the evaluation and three blank index cards.

2. Students were then given 10 minutes to write three statements about the course: one sentence saying something they liked about the course; one sentence saying something they didn't like about it; and a third sentence making some comment (positive or negative) about their project work.

3. After all students had completed the cards, the cards were distributed to all members of the class and they were provided with the opportunity to evaluate all statements (including their own) on a 4-point scale with no neutral point. On each card they were to score the statement as a 1 if they strongly agreed with it; 2 if they mildly agreed; 3 if they mildly disagreed; and 4 if they strongly disagreed with the statement. 15 minutes were set aside for this activity.

4. The cards were gathered together and sorted into three groups: positive evaluation statements where there was general agreement; positive or critical evaluations where there was general disagreement; and critical evaluations where there was general agreement.

The evaluator then conducted a discussion with the students arranged around the groups of statements. Firstly, the positive statements with general agreement were read out and students provided with an opportunity to expand on what they enjoyed about the course. Secondly, the contentious issues were read out and the discussion that ensued was structured with the aim of resolving differences in perceptions. Then the critical evaluations were read out and the discussion was aimed at expanding on the reasons why the critical attitudes existed and what could be done to improve the paper in future.

After the discussion session, a summary report was written by the evaluator to classify the statements and summarize the discussions. This was circulated to all members of the class and to the rest of the teaching team, to act as a record of the discussion and to encourage further action.

2.3 Evaluation outcomes

Students wrote a total of 31 statements during the evaluation session. Statements were coded N (critical), P (positive) and numbered in each classification (*e.g.* N01, P02) for reference. A weighted sum was calculated to score each statement, using a weight of 2 for strong agreement, 1 for mild agreement, -1 for mild disagreement and -2 for strong disagreement.

When assessing statements, we looked at the degree of agreement across the class (the score for each statement) to decide whether or not the class found that a statement was a valid opinion or not. A positive *outcome* was where there was a positive statement with general agreement, or a negative statement with general disagreement. Conversely, a *critical* outcome was where there was a critical statement with general agreement or a positive statement with general disagreement (Table 1).

Table 1. Evaluation statements by type and outcome.

Type	No. of statements
Critical statements	20
Positive statements	11
Critical outcomes	18
Positive outcomes	8
Neutral outcomes	5

On conducting more detailed staff reflection after the discussion session, we classified the statements against the course objectives to which they referred. The balance between positive and negative outcomes (Table 2) indicates where problems lie in terms of conflicts between staff and student expectations.

Table 2. Staff/student conflicts of expectations.

Objective	Positive outcomes	Critical outcomes
Flexibility of assessment	2	4
Group work provides a rich learning experience	4	5
Workload is reasonable and under control of groups	0	2
Course content arranged around broad issues generally relevant to projects.	3	3
Selection of projects is intended to be reasonable and feasible.	2	4

By looking at the actual nature of the statements and from the discussion related to these statements, we discovered that workload issues were actually related to the management of group conflict, an issue that had not arisen in our consideration of other forms of feedback from students. In this case, we found the activities associated with reflection to be an effective method of discovering a real issue with the student learning.

Peer review is an excellent method of validating statements and encouraging further reflection on the part of a student who made a statement. Since the process is anonymous, the discussion can allow issues between students to be resolved, as well as differences between staff and student expectations. Some evaluations were phrased in robust terms that would be difficult to imagine emerging spontaneously from a face-to-face discussion – *e.g.* "*N03 I did not enjoy the group work. There are too many parasites who sit and do nothing…"*

Opinions are mediated across the class by the process of peer-evaluation of statements. For example, the following statement (commenting on project work) "*N19 Because there are no internal assessments, students tend to spend little time on this course*" found the strongest disagreement of all statements across the class, with various students highlighting during the discussion exactly how much time they spent on project work and what their expectations were of other group members.

Two critical evaluations that conflict with course objective 2 (flexible assessment) found broad agreement: "*N01 Project assessment could have been better defined*" and "*N04 Would prefer more regular assessment, especially on theoretical material not related to current project*". Since we believe that the nature of the assessment is appropriate given our constructivist approach, we see the opportunity here to engage in dialogue with students to explain the assessment principles and project topics.

On the other hand, positive outcomes can be found where evaluation statements reinforce course objectives. "*P03 Meeting new people – software engineering is a very social course…",* "*P08 Projects give good practical experience*" (both positive consensus). Where such evaluations arise, the subsequent discussion is a good opportunity for the teaching team to add context to these principles, providing deeper explanation of a topic that has just received positive evaluation and hence likely to be more relevant to students.

Some statements contradict each other and present an ideal opportunity for an issue to be resolved by allowing students to identify the contradiction and to discuss potential explanations. For example, the two statements "*N15 The project is rather boring – perhaps a more interesting project idea could have been used*" and "*P09 I think that the project is an interesting one…" both* found broad agreement. It emerged through the discussion that a project *can* be both boring and interesting at the same time; but without the evaluations arising from the students, the visible contradiction and the ensuing discussion, the exact nature of student interest/disinterest would have been opaque to the teaching staff.

Interestingly, we found only two statements (both negative) regarding workload in the paper. We found more to suggest that group conflict was the more important issue. The "knot" of related issues entwining group work, conflict, group members not contributing equally suggests to us that we need to rethink the administration of project groups, mediating among groups more actively to address conflict as it arises.

We found a large number of statements (17 overall, but with no clear consensus) that related to the selection of project topics and the "just in time" strategy of delivery in lectures. From the statements and the discussion that followed, it became clear that students had been expecting specific instruction on how to do the project, rather than generic tools and techniques such as project management, configuration management and risk management.

2.4 Classification of evaluation outcomes

There are several possible outcomes from the evaluation process. Some of these can be handled during discussion immediately following peer evaluation; some require deeper reflection and further discussion; others represent a fundamental disconnect between teachers and students that require careful work to resolve.

1. Disagreement among class members with evaluation statements. This is validation through peer review of the various evaluations made by members of the class. This kind of control mechanism is useful to mediate the effects of "rogue" evaluations.

2. Contradictory evaluations of similar statements. This situation can indicate a common problem with evaluation – that those assessing the statements did not completely understand what it was that they were evaluating.

3. Contradictions between separate evaluation statements. Similar to the above classification, but the contradiction may not become apparent until all statements have been reviewed after the original discussion. Resolution of the conflict depends on a different form of presentation than that described in the protocol above, presenting both contradictory statements together and allowing a general consensus to emerge.

4. Alignment or conflict between evaluation outcomes and course objectives. Conflict can be detected by classifying negative outcomes according to course objectives; alignment by classifying positive ones. In both cases, a more engaging dialogue between staff and students is the key to resolving the conflicts and highlighting the alignments that may exist.

In deeper examination of this latter class of evaluation statements, we identified three main areas of conflict that we need to resolve in future iterations of this paper.

1. To make assessment principles and mechanisms clearly defined and cross-check with students that they are understood.

2. To make the relationship between lecture material and projects clear and find room in the class schedule for material that does directly relate to projects. We believe that the simulated customer approach is a valid vehicle for this latter form of delivery.

3. To continually revisit group conflict and take a more active role in mediating conflicts among groups.

3. CONCLUSIONS

The protocol for evaluation develops the notion of a "critical community" in terms of professional development where reflection can take place and be made widely known. Participants view themselves as enquirers who are prepared to share reflections in a supportive environment.

In the study described here, the protocol was used in an immediate context – the evaluator classified the statements in the same session as the evaluations were completed by students. While this provides immediacy, deeper reflection is often needed to allow staff to investigate an issue in greater detail. This is particularly important for contradictory statements, or for statements with strong consensus that conflict with course objectives.

The protocol we have used performs the important role of mediating opinions. Statements made by individuals are considered by the group and by teaching staff. A strong opinion held by one student may not find support across the class. However, a statement made by one student may cause another to think more deeply about their own experience and be able to discuss it. This protocol provides a valuable and useful form of feedback to staff and also encourages students to reflect more deeply on what has happened and what they have actually learned.

4. REFERENCES

1. Biggs, J. *Teaching for Quality Learning at University*. Society for Research into Higher Education and Open University Press, Buckingham, 1999.

2. Bjorner, D. Some Thoughts on Teaching Software Engineering - Central Rôles of Semantics. in Boer, F.d., Heijden, M.v.d., Klint, P. and Rutten, J. eds. *Liber Amicorum: Professor Jaco de Bakker*, Stichting Centrum voor Wiskunde en Informatica, Amsterdam, The Netherlands, 2002, 27-45.

3. Brown, J., Bloodshot eyes: workload issues in computer science courses. in *Seventh Asia-Pacific Software Engineering Conference*, (Singapore, 2000), 46-53.

4. Butcher, A.C., Stefani, L.A.J. and Tariq, V.N. Analysis of peer-, self- and staff-assessment in group project work. *Assessment in Education*, *2* (2). 165-186.

5. Daniels, M., Faulkner, X. and Newman, I., Open Ended Group Projects, Motivating Students and Preparing them for the "Real World". in *15th Conference on Software Engineering Education and Training (CSEET'02)*, (Covington, Kentucky, 2002), 128-139.

6. Fink, F. Integration of engineering education and productive engineering, Finnish Higher Education Evaluation Council, 2000.

7. Flener, P. Realism in project-Based Software Engineering Courses: Rewards, Risks and Recommendations, Uppsala University, Uppsala, Sweden, 2003.

8. Ford, M. and Morice, J., Using Micro Management Techniques to Overcome Problems in Group Assignments. in *Informing Science and Information Technology Education Joint Conference*, (Pori, Finland, 2003), 1311-1321.

9. Gehrke, M., Giese, H., Nickel, U., Niere, J., Tichy, M., Wadsack, J.P. and Zündorf, A., Reporting about industrial strength software engineering courses for undergraduates. in *Proceedings of the 22rd International Conference on Software Engineering*, (Orlando, Florida, USA, 2002), ACM, 395-405.

10. Graue, M.E. Integrating theory and practice through instructional assessment. *Educational Assessment*, *1*. 293-309.

11. Jacques, F. *Logical space of interlocution*. PUF, Paris, 1985.

12. Liu, J. and Stroulia, E., A Lightweight Project-Management Environment for Small Novice Teams. in *3rd International Workshop on Adoption-Centric Software Engineering*, (Portland, Oregon, 2003), 42-48.

13. Oldfield, K.A. and Macalpine, M.K. Peer- and self-assessment at tertiary level - an experiential report. *Journal of Further and Higher Education*, *20* (1). 125-132.

14. Rowland, S. *The enquiring tutor - exploring the process of professional learning*. Falmer Press, Lewes, 2000.

15. Savery, J.R. and Duffy, T.M. Problem based learning: An instructional model and its constructivist framework. *Educational Technology*, *35*. 31-38.

16. Shaw, M. and Tomayko, J.E., Models for Undergraduate Project Courses in Software Engineering. in *Fifth SEI Conference on Software Engineering Education*, (1991), Springer-Verlag, 33-71.

17. Stewart, S.R. and Richardson, B. Reflection and its place in the curriculum on an undergraduate course: should it be assessed? *Assessment & Evaluation in Higher Education*, *25* (4). 369-380.

18. Upchurch, R. and Sims-Knight, J., Portfolio Use in Software Engineering: An Experience Report. in *Frontiers in Education 2002*, (Boston, USA, 2002).