AC 2011-221: IMPROVING EFFICACY OF PEER-EVALUATION IN TEAM PROJECT SCENARIOS

Eckehard Doerry, Northern Arizona University

Eck Doerry is an associate professor of Computer Science at Northern Arizona University. His research interests fall within the broad area on "Groupware support for Online Groups", with active research in portal-based tools to support distributed scientific communities, groupware tools to support small, distributed engineering design teams, and distance education tools and environments. He has been a long-time advocate of realistic, interdisciplinary team design projects as a key element in engineering education, and has been managing advanced project teams in the Design4Practice program at NAU for nearly ten years.

James Dean Palmer, Northern Arizona University

Dr. Palmer is an assistant professor at Northern Arizona University where his research interests include undergraduate computer science education, language design, and computational storytelling.

Improving Efficacy of Peer-Evaluation in Team Project Scenarios

Abstract

A vast majority of engineering programs now include one or more project courses in the curricular core, in which students work in teams to solve realistic engineering challenges. Although team projects undoubtedly have strong potential for developing critical teaming skills, *individual performance in these skill areas is not easily accessible to the instructor*; monitoring and documenting the extent to which individual learning *actually occurs* in team project contexts represents a difficult challenge. In this paper, we describe an effective system we have developed to gain insight into internal team dynamics and individual performance based on an integration of structured task reports and anonymous peer evaluations. Benefits include early detection of internal team difficulties, an effective system for documenting individual contributions in team contexts, and a strong mechanism for differentiating student grades based on individual performance. Despite the minimalist philosophy shaping the evolution of our approach, truly effective use of the system developed requires substantial time investment by the instructor; we close by outlining an online system we are developing to largely automate the team management process. Benefits of automation include real-time feedback to instructor and team members, automated flagging of potential trouble, and automatic documentation of contributions/performance for individual team members.

1.0 Introduction

The past decades have brought a growing awareness of the value of integrating training in a broad range of "soft skills" – including teaming, project management, and oral and written communication – into the modern engineering curriculum. As a result, the vast majority of engineering programs now include one or more project courses in the curricular core, in which students work in teams to solve realistic engineering challenges.

This emphasis on team-based project work, along with recent ABET guidelines requiring programs to document learning outcomes at the individual level, has exacerbated an age-old problem with team-based projects: how to reliably measure individual contributions to group performance in team-based project scenarios. In the worst case ("coat-tailing"), one or more team members contribute little or nothing, forcing their teammates to pick up the slack — and yet all team members end up receiving the same "team grade" when the project concludes. This not only erodes the integrity of evaluation within an academic program, but also has serious consequences for the team as well: experience shows that simply having to cope with a non-performing team member can actually result in more stress and effort for the team than if that team member were not present at all. Ultimately, nothing erodes team morale faster than working overtime to make up for a poorly performing teammate, only to receive exactly the same evaluation. Thus, development of reliable mechanisms for (a) quickly detecting and dealing with internal team problems, and (b) adjusting individual credit given for team products based on actual effort invested by each individual is crucial in any team project context.

A fundamental obstacle in evaluating individual performance in teams is that, due to the very nature of the teaming experience, instructors inherently have limited insight into detailed internal

team dynamics. The whole point of teaming is to emphasize cooperative interaction between team members in pursuit of a common goal, resulting in team (rather than individual) accountability to the instructor; picking apart the relative value of individual contributions towards the common team product thus represents a difficult challenge.

As realistic practice-oriented team projects have become a fixture in engineering education over the past 15 years, a growing body of literature has developed around best practices in organizing and managing such teams, including the critical issue of evaluating team deliverables and performance [1-3]. Of the variety of evaluative mechanisms and schemas that have been developed, *peer evaluations* have been found to provide excellent insight into individual contributions and behavior in a team [4], [5]. A variety of peer evaluation schemas, in which team members evaluate each other's contribution to team outcomes using some scale or metric have been developed, but metrics can generally be divided into two groups:

- Qualitative approaches that emphasize "team citizenship", using qualitative ratings of professionalism, seriousness of effort, listening ability, and other collaborative or social skills [3], [6], [4], [1], [2].
- Quantitative approaches that focus more on quantifying the contribution or effort invested by each team member towards the production of team products [5], [7], [6], [1], [8].

While most peer evaluation schemas reported in the literature incorporate both quantitative and qualitative elements, there is generally an emphasis towards one or the other. A slight bias appears to exist towards qualitative approaches based on the argument that being a good team citizen should ultimately count more than mere technical skills [6], [3]. Concerns about inherent subjectivity and unfairness in peer evaluation schemas [8] have been shown to be largely groundless, as several studies have demonstrated the internal and external consistency of peer evaluations [6], [4]. It is evident, however, that the *efficacy* of peer evaluation approaches – defined as how accurately the schema tracks individual contributions and gives insights into internal team function – varies widely depending on the detailed design of the evaluation schema. Generally speaking, the efficacy of peer evaluation schemas appears to increase in proportion to the number of evaluative mechanisms deployed and the evaluative integration of those mechanisms [9]. Unfortunately, increasing the number and complexity of evaluative mechanisms also increases the overhead associated with deploying the schema, not only for team members, but also (and particularly) for the instructor. As a result, our experience is that most instructors settle for an imperfect but practically manageable approach.

In this paper, we describe the peer-evaluation approach that we have developed at Northern Arizona University over more than a decade of teaching courses in a team-project-based curriculum. Driven by resource limitations and expediency, our approach has been shaped by a minimalistic philosophy: how can we achieve maximum efficacy with minimal overhead for students and instructors? We begin with a brief description of our Capstone course and the evolution of the efficient peer evaluation schema we have developed. We then present our ongoing efforts to increase the efficacy of our peer-evaluation system while managing overhead, by integrating the key elements into a flexible automated system for supporting team-based projects.

2.0 Evaluating Individual Performance in Student Design Teams

As awareness of the value of integrated training in a broad range of "soft skills", including teaming, project management, and oral and written communication into the modern engineering curriculum in engineering education has grown in the past decades, many engineering programs were redesigned to include one or more project courses in the curricular core. The engineering programs at Northern Arizona University were among the first to embrace this development fully, deploying the innovative Design4Practice (D4P) curriculum in 1996 [10-12]. This practice-oriented engineering curriculum, crafted with extensive input from industry, was designed to provide students with hands-on learning experiences and continuous practice of a broad set of professional skills in order to better prepare them for careers as engineering practitioners. Central features of the program include:

- A four-year sequence of increasingly challenging team-based design projects.
- Interdisciplinary teaming in one or more team design projects.
- Cross-disciplinary collaboration in sequenced courses.
- Active participation of practicing engineers from industry through teaching, program evaluation and project sponsorships.
- A required core spanning all engineering disciplines, incorporating the complete design cycle within simulated industrial product development contexts.

The strategy behind the Design4Practice sequence is straightforward: to introduce the students to the design process early in their college careers, and maintain a constant rate of increasing complexity throughout the four years of study. By graduation, the students are well versed in the design process, oral and verbal communication, and key teaming skills, and hence are immediately able to contribute productively in their first professional employment.

2.1 Evaluation of Team Members in Senior Capstone Design

Although peer evaluation is incorporated in every team-based design course within the program, we focus here on our experience with the final course in the D4P sequence for Computer Science, called Senior Capstone Design. In this course, teams of students work on real corporate software projects solicited from our industrial partners, acting as semi-independent software developers for the industry client. The high level of independence is critical, and marks this as a true capstone course, requiring students to integrate and apply everything they have learned in the program to create real software to address a real world problem for a real client. Each team is assigned a faculty mentor, who meets regularly with the team to review progress and provide guidance when problems are encountered; such problems generally center around development strategy, client interaction, and project management rather than technical issues (which students are expected to solve on their own).

The semi-independent nature of the Capstone Design course makes it particularly interesting and relevant to our discussion of evaluating individual performance in teams. Because there are no lectures, exams, homework, or other materials to evaluate, the entire course grade is based on team deliverables. Instructor insight into internal team dynamics is more restricted than in a conventional team project-based course because of the limited interaction with the student team and the focus on independent student engagement with an outside project. In this context, the

integrity of the entire course rests on the efficacy of the evaluation schema used to map team performance to individual student grades.

Developing an effective solution to this problem has proved a tremendous challenge for faculty mentors in our Design4Practice program, and the solution we have evolved at this point has been shaped by many failures. In the early years, the course developed a reputation of being "impossible to fail", due primarily to ineffectual peer evaluation schemas that failed to effectively identify or punish non-performing team members. The following sections describe the evolution of our current evaluation schema, using the failures along the way to identify and extract a set of best practices for effective team evaluation. As a foundation for this discussion, we begin with a brief outline of our team evaluation and disciplinary framework.

2.2 Basic Evaluation Philosophy and Framework

Our overall approach to team evaluation is straightforward, based on the general philosophy that the instructor should evaluate team deliverables based on professional standards of technical quality, clarity, and presentation, and that this team score should be somehow modulated by the peer evaluation system to arrive at individual scores. This overall model is common, and a broad variety of schemas based on it have been developed in the past decade, as project-based courses have increasingly become central features of modern engineering curricula. The central goal of all such evaluation schema is ultimately to preserve academic integrity and fairness by assigning credit for team products to team members based on their individual contributions and, in particular, ensuring that non-performing team members don't receive the same scores as those who do all the work. Many of these schemas trace their roots to the early work of Millis and Cottell [1] and Brown [5], which outline several quantitative and qualitative models for evaluating individual performance in teams. Our overall approach is based loosely on Brown's Autorating system, and consists of the following steps:

1. Team deliverables are evaluated by the instructor, using a scale based directly on the quality of deliverables expected in real-world consulting contexts.
2. Students provide peer evaluations to the instructor in the required form, usually via email. The instructor enters these into a spreadsheet and, at the appropriate point, calculates a peer evaluation score for each student.
3. The peer evaluation scores are used to augment or modify the scores for team deliverables to arrive at individual scores.

The exact nature and ultimate efficacy of this mapping process, of course, depends on the detailed implementation of the last two steps: what metrics are used as the basis of peer evaluation, what are the scoring scales for those metrics, how are ratings by various team members combined, and how are peer evaluation scores used in determining the final individual grade? Our exploration of possible permutations along the dimensions outlined by these questions provides the grist for the discussion presented in Section 3.

3.0 Evolution of our Evaluation Schema

Because the Senior Capstone Design course is already one of the most time-intensive elements (for both students and faculty mentors) in our curriculum, overall development of our schema for team evaluation has been shaped by a philosophy of minimalism: how can we attain the best possible insights into individual effort contributions to team products with the minimum of

overhead for instructors and students? We therefore began with a very simple system, and then augmented this incrementally in response to observed shortcomings. The following subsections describe the evolution of this system into its current form, which has been stable for nearly four years. For each iteration, we first describe the schema, and then discuss its evaluative efficacy, with an emphasis on shortcomings leading to the next iteration.

3.1 Version 1: Qualitative Evaluation, Team Reports

In the initial version of our team evaluation model, peer evaluations were required just once, at the completion of the project. Peer evaluations were in the form of a questionnaire in which each person rated teammates on a 10-point scale across a variety of qualitative categories, including meeting attendance, professionalism, overall reliability, and so on; free-form comments and suggestions for each teammate were solicited as well. The questionnaires were compiled into a broader "Teamwork Report", in which the team was asked to reflect on their teaming success. This "open evaluation" model is similar to Clark [6], who advocates for open discussions of peer evaluations as a basis learning and improvement. Scores from the Teamwork Report were counted as 10% of the final course score.

Version 1: Evaluation

This approach at first appeared to be successful, with insightful narratives of successful teamwork appearing in Teamwork Reports. It soon became apparent, however, that the generally positive reviews appearing in the reports often did not match up with reality. In many cases, serious dissatisfactions with teammates revealed in private office consultations with instructors never appeared in the peer ratings or, if they did, then in much milder form. Even when poor performance was endemic, students simply did not feel comfortable publicly confronting teammates with poor ratings in the report, worrying about hard feelings and retribution from slighted teammates. Another drawback was the considerable effort required to produce the report, coming precisely at the crux of team effort, where teams were struggling to finish up projects at term's end.

3.2 Version 2: Anonymous Evaluation, Individual Reports

In the next iteration of our system, the peer evaluations were made anonymous, and restricted to the one-page questionnaire of ratings and comments; no Teamwork Report was required. Students were explicitly promised anonymity of peer evaluations in the course syllabus, and peer evaluations were emailed directly to the instructor by individuals. To provide an additional sampling point, two evaluation (midterm and final) were required.

Version 2: Evaluation

This modification produced a dramatic shift in the scoring and tone of evaluations, as students felt free to vent their frustrations with teammates; students clearly felt more comfortable being honest under these conditions. Overhead was reduced dramatically as well. In comparing scores with the accompanying commentary, however, broad inconsistency in scoring was apparent. Some students gave very harsh ratings for relatively minor shortcomings in teammates, while others remained relatively generous in the face of similar behavior. Moreover, we began to question the practical relevance of some rated categories to outcomes: should professionalism or being on time to meetings really receive the same weight as completing tasks on time?

3.3 Version 3: Zero-Sum Model, Strong Focus on Task Completion

Our next iteration was quite dramatic, driven by the practical outcomes-oriented philosophy of the Design4Practice program: we discarded the more subjective soft skills categories (professionalism, listening skills, communication, etc.) and focused solely on contribution to work outcomes. Using a quantitative approach similar to the Work Product Pay Packet [7], each team member was given (N x 100) points to distribute across all teammates, including him or herself, where N is the total number of team members. Thus, each member of a three-person team would have 300 points to distribute across the three teammates, representing contributions by each teammate to the team deliverable(s); in a well-balanced team, each team member would simply receive 100 points. To guard against vindictive or wildly unfair ratings, detailed commentaries justifying each rating in terms of tasks assigned and completed was required as well, and students were informed that they might be contacted by the instructor for clarification in extreme cases. Team members emailed their ratings to the instructor, who averaged the ratings received for each team member (including the self-rating) to arrive at an overall peer rating; this rating was then counted towards the 10% peer evaluation portion of the course grade.

This very outcomes-oriented, quantitative approach goes against some existing philosophies [3] that argue that this approach demoralizes students with lower technical skills and creates a competitive team environment. Our motivation was strictly practical: in engineering practice, what matters most is producing quality products on time, and in that context, those students who are more productive (due to technical skill, organization, or whatever reason) *should* receive more credit for team products. In addition, our aim was to reduce the subjectivity (and thus variability) in ratings by giving students maximally objective criteria to focus on in rating their teammates.

Version 3: Evaluation

This approach simplified and focused the peer rating process considerably, and the zero-sum model focused simply on contributed effort was enthusiastically accepted by students. Students in well-functioning teams complained that the required commentaries were pointless (since everything was going well), so a small change was made: detailed justifying commentaries were required only in cases where ratings were over 120 or under 80. In this fashion, "normal" variations in effort invested could pass without further comment, while extreme ratings would require justification and draw instructor attention. In accordance with other findings [7], [8], our initial concerns about collusion or ganging up under this more quantitative model were unfounded. The zero-sum model tends to discourage collusion, because heavy contributors are unlikely to give up an improved score in order to bump up the score of a weak teammate; ganging up is discouraged because of the requirement to rationalize severe ratings and instructor oversight.

While generally satisfying, several weaknesses remained in this system. First, the model of counting peer evaluations as simply a component of the course grade (e.g. 10% of final grade) was simply not consequential enough. Even the worst performers receiving, say, a 50% peer rating would lose only 5% on the overall course grade, meaning that in many cases they received the same grade as top performing teammates. Even increasing the peer evaluation component of the final grade to 20% had no strong effect on final grade. A second problem was reliance on just a midterm and final peer evaluation. It is not uncommon for student effort investment in the project to vary somewhat over the term, e.g., due to time pressures in other courses or at work.

This is normal, and should result in local variations in team participation. Examining the written commentaries revealed that, in a system with two semester evaluations, such temporary fluctuations often resulted in a "negative carry-over" effect: the sour taste of poor performance of a teammate, even for just one week or on one deliverable, could lower the rating for the entire half of the term.

3.4 Version 4: Peer Evaluations as Deliverable-Specific Score Modifiers

In this iteration, the fundamental model for how peer evaluations factor into individual course grades was changed: rather than counting as some percentage of the overall course grade, peer evaluation scores were used as direct factors to adjust the credit received for deliverables, and peer evaluations were required for each major deliverable or milestone of the project; the average peer evaluation score was treated as a weighting percentage to map team grades to individual scores. For instance, if a particular deliverable is awarded a score of 85/100 by the instructor then:

- in a perfectly balanced team, all team members receive an average peer evaluation of 100; this amounts to an adjustment factor of 1, and all team members simply receive the 85 as an individual score.

- in a team with some fairly substantial variation, Student A may have received an average rating of 108, while two other team members received an average rating of 96. Thus, Student A would receive an individual score of 91.8 for the deliverable, while the other two receive scores of 81.6.

In essence, this modification merely brings the model for integrating peer evaluations in the grading schema directly into line with the outcome-oriented philosophy that motivated Version 3: team members should be rewarded in direct proportion to effort invested. The number of peer evaluations were increased as well ,with a separate peer evaluation attached to each major milestone or deliverable, and students were instructed to evaluate peers strictly on contributions to that specific deliverable.

Version 4: Evaluation

This iteration has been very successful, and has been the basis for our peer evaluations for the past five years. Our initial concerns that students would vengefully give punitively harsh ratings for minor infractions have proved to be largely unfounded. Students appear to appreciate the significant but localized effect of this rating model, as evidenced by the following characteristics in well-functioning teams:

- Peer ratings are generally internally consistent across all team members, even when there are variations in effort invested. This means that even those who invested less effort tend to award themselves fewer points, consistent with and in direct proportion to ratings by other team members.

- In many cases, poor performers actually rate themselves *lower* than the ratings they received from teammates, often including apologetic commentary about a temporary overload creating a drop in contribution for that deliverable, and expressing that teammates "deserve" credit for picking up the slack.

- Ratings often varied significantly with each deliverable. Teammates that received low ratings for one deliverable often receive normal or higher ratings with the next deliverable. The direct connection of peer evaluations to specific deliverables limits the consequences of a short period of poor performance to the specific deliverable affected and seems to avoid poisoning attitudes toward a teammate for the entire term. Conversely, temporary poor performers are motivated to improve (i.e., "the next deliverable is a new ballgame"), versus discouraged (i.e., "I've blown my peer evaluation for the term, so what's the point?").

Note that all of the above characteristics apply to *well-functioning* teams, i.e., teams where there is some normal give and take, but are generally able to work things out. In our experience, however, approximately one in five teams are "dysfunctional", with one or more team members substantially failing to function effectively, seriously hampering team success. We discuss our model for dealing with this situation in detail in Section 4; the interesting point for the moment is that our peer evaluation model is very useful in early detection of such trouble, allowing aggressive intervention early in the term. In particular, we have noted that dysfunctional teams exhibit peer ratings profiles that are not only internally inconsistent, but nearly always exhibit one particular profile: The contributing team members rate the non-performing member significantly lower, distributing the points between contributors, while the non-performing member gives everyone (including him or herself) an even 100 rating. This pattern has been so consistent that it has become common knowledge in our faculty, and instructors regularly use it to detect trouble early, by briefly scanning incoming ratings. When such a "red flag" is noted, the instructor can immediately call the team into conference, and actively work to resolve the problem.

3.5 Final Schema: Augmenting Peer Evaluation with Task Articulation

We have found that the previous iteration (Version 4) largely meets our key criteria for a peer evaluation schema: it provides a reliable, minimalist, low overhead (for students, at least) mechanism for mapping team outcomes to individual scores based on individual level of contribution. We have added one further element, however, to help increase the efficacy of the system, based on the following observations:

- When a dysfunctional team was detected, resulting in instructor intervention, the most common reason given was miscommunication: the non-contributing member claims not to have realized that a task had been assigned, or that a specific due date had been set. Moreover, weekly meetings with nominally well-functioning teams showed that, even in these teams, there was often some confusion about task assignments.

- From an administrative standpoint, substantial documentation is required by the university to justify "firing" (the assignment of a failing grade) a team member. A mechanism was needed to create a paper trail starting from the first day of class to justify such drastic measures.

- In remediating a dysfunctional team situation, the instructor often had no evidence to go on, with one party claiming deadlines had been missed by a teammate, and the accused party claiming that no task assignment or deadline had ever been set.

Our solution has been to formalize the initial informal steps in the conflict resolution process, which are centered around increasing task documentation (Section 4), in the form of a weekly "task report". A template for this report is provided to teams (Figure 1).

<team Logo>	**Weekly Team Task and Status Meeting Report**	<report #>

Team: <Your team name>	**Meeting Date:**
Project Title: <your project title>	
Team members present: <list all member first names; I'll circle who's there>	**On-time:** <list all member first names; I'll circle who's on time>

Progress of Team Meetings

a) last meeting(s):
b) next meeting(s):
c) Communication Problems?

Last week's Tasks: Status

Task Title: <short title. Used as label for the task → easy to track from week to week.>	**Task Initiation:** <date the task started>	**Orig. Due Date:** <date task was initially due>	**Status:** <current status of task as of today. Could be "on-time", "complete", "overdue+expected due date"
Who (%): <who's responsible for it; if multiple persons, then to what percentage>			
Description: <2-3 sentences to describe what involved with the task and its goals in more detail. Include succinct statement of what shall be delivered!>			
Outcome/Progress: <Concise statement of exactly what is currently completed. If done, it describes outcome. >			
Task Eval: < Self-eval of team performance on the task. If satisfactory, say so. Else critique outcome or the team's process and state how it might be improved. Shows that team is striving for perfection.>			

<please leave at least one blank line between tasks for readability>

This week's Tasks: Work plan

Task Title:	**Task Initiation:**	**Orig. Due Date:**	**Status:**
Who (%):			
Description:			

Upcoming Tasks: Planning

Task Title:	**Who (%):**	**Rough Due Date:**
Description: <just give light description here. Idea is to show that you've thought about what's coming up>		

Timeline check:

a) Slippage? Up-to-date?
b) Notebook Status:
c) Website Status:

Other Problems / Other Issues:

Figure 1: Task report template provided to teams.

Task reports are filled out weekly by each team, and must be presented at the start of weekly team meetings with the instructor. As indicated in Figure 1, the task report provides running documentation of the team's current focus of effort and distribution of tasks. Each task is briefly outlined, with emphasis on status, due date, and team members assigned to it. Task "roll

through" the three sections over the course of several weeks: they are first roughly formulated as upcoming tasks, then are fully specified and assigned as active tasks for the upcoming week, and then migrate to last week's tasks as they are completed. This "dynamic task report" concept has proven effective for several reasons:

- it provides detailed insight into the internal workings of the team for the instructor, and provides a solid basis for weekly meetings with the team. Rather than discussing vaguely "how things are going", discussion can focus on what tasks were assigned, what tasks were completed, and if not, why those tasks are slipping.

- it creates solid documentation of team and individual performance for the instructor, providing an objective basis for instructor evaluation of the team, and for disciplinary action, should this become necessary.

- it is a valuable tool for *avoiding* trouble in the team by clearly delineating assigned tasks, and providing clear documentation of due dates and expected task products. This removes the ambiguity that poor performers typically use as concealment.

- it promotes team progress and learning of proactive project management skills by forcing teams to think explicitly and in detail about where they are, what needs to be done next, and how personnel resources should be deployed.

Note that task reports are not a replacement for the team's overall project plan, which teams are required to maintain as well (usually using project management software), but rather a complementary articulation of detailed weekly assignments that move the team towards completing the broader tasks or milestones shown in the project plan.

The integration of weekly task reports has proven to be a worthwhile improvement in team evaluation, providing valuable insight into individual contributions and internal dynamics of teams. The drawback, of course, is the overhead it creates: teams must create a task report before each weekly meeting with their mentor, and instructors must archive task reports, then reconcile them with peer evaluations when calculating grades. More generally, the overhead for team evaluation overall has increased substantially since Version 1: one project peer evaluation has expanded into 5-8 (depending on deliverables), plus the weekly task reports. Practically speaking, this inevitably reduces the efficacy of the system, as both students and instructors cut corners to reduce the workload. Our current effort to reduce this overhead while further enhancing the efficacy of the schema by embedded team evaluation in a sophisticated automated system is outlined in Section 5.

4.0 A Formal Policy for Dealing with Chronic Poor Performers

In the previous section, we emphasized that a good team evaluation schema must motivate team members to contribute their fair share of work, while supporting early detection of under-performing team members. But what should be done when a problem is detected? As a first step, of course, teams should be mentored in solving internal problems themselves [13]; conflict resolution is an important learning goal of the teaming model. If problems persist, however, our finding is that explicitly raising the issue of workload distribution and task completion with respect to the under-performing team member openly, during a weekly team meeting with the mentor, can be enough to resolve the problem. Once poor performers realize that their non-

performance will be quickly evident and that they are accountable, they often pick up the pace. The anonymity of peer evaluations can easily be preserved in this discussion by staying focused on the task reports, and using these to question contributions and task completion by the alleged non-performer; the students need never know that this focused probing was engendered by recent peer evaluations.

There are some cases, however, where poor performers are stubbornly unwilling contribute their fair share, and here it is important to have a formal process in place for "firing" the chronic under-performer. One might argue that this is overly harsh, and that the peer evaluation system itself (if properly designed) will ensure that non-contributing team members get a failing grade in the end. This may technically be the case, but our experience has shown that this passive approach is often a disaster for the team, as other team members become increasingly frustrated, resentful, and fearful that the weak contributions of the non-performing team member will drag down the quality of deliverables for the whole team. Having to deal continually with the uncertainty of a poorly performing team member – i.e., assigning tasks, then having to scramble to fill the gap when no results materialize – can actually *create more effort than having to complete the project with one less person*. In any case, having a formal expulsion mechanism makes an important symbolic point: this is a real project for a real client, and as in the real world, non-performance will lead to dismissal. As in any corporate setting, it is important to have a well-defined policy for documenting non-performance and escalating disciplinary action. Our process is adapted from industry models, and consists of the following incremental escalations:

Level 1: Increased internal structure. The team leader should attempt to resolve the problem by sending written task reminders, precisely detailing deadlines and descriptions of expected task deliverables in the task report, and posting written meeting minutes. This also generates clear documentation of non-performance.

Level 2: Formal disciplinary memo. If poor performance continues, a formal memo to the offending member is drafted by the remainder of the team. The memo is completely professional and very succinctly and clearly describes the specific performance failures to date. The memo is CC'd to the faculty mentor. Upon receipt of the memo, the faculty mentor follows up with a note to the offending team member containing a link to the expulsion policy (posted on the course website), and emphasizing that the situation is serious.

Level 3: A formal contract for improvement. Within 24 hours, the offending team member must produce a formal response memo, in which he or she addresses each alleged instance of non-performance, explaining why the failure occurred, and what changes he or she is making to ensure that it will never happen again. This memo is reviewed for quality and completeness by the faculty mentor, and revisions are made as necessary. The response then becomes a formal contractual obligation to the team; it is clear that breaking this commitment will be grounds for dismissal from the course.

Level 4: Sanctions and Expulsion. If the disciplined team member fails to live up to one or more of the specific commitments made in the response memo, disciplinary action is initiated. Generally the department chair is invited to join a formal meeting, at which the situation is reviewed and appropriate disciplinary measures are decided, including dismissal from the course.

In the nearly 10 years since we established this formal policy, only two students have been dismissed from the course; only five further cases have reached the disciplinary memo stage. Simply having an explicit expulsion policy in place, along with a team evaluation schema that exposes and documents individual contributions, has reduced the incidence of non-performance in teams, and made it fairly straightforward to deal with when it does occur.

5.0 Future Work: Enhancing Team Evaluation with Automation

Despite our focus on a minimalist approach to building up an effective team evaluation schema, the overhead – in terms of effort required by students to produce peer evaluation and task tracking information, and effort required for the instructor to monitor team dynamics and calculate individual scores – has increased steadily with each evolution of our schema. In the final version, instructors must monitor peer evaluation emails, enter scores into complex spreadsheets, apply peer scores to generate individual grades for deliverables, and reconcile peer scores with task performance information drawn from weekly task reports. Student overhead for generating peer evaluations is low, but creating weekly task reports is arduous and error-prone. As a result, both students and instructors gradually disengage from the evaluation process, completing their responsibilities more superficially as stress increases: students begin to copy-paste task reports and skip peer evaluations, and instructors can only sporadically monitor incoming information for early signs of deteriorating internal team dynamics. What is needed is a system that supports more evaluation, better integration of task reporting and peer evaluations, yet reduces the amount of overhead.

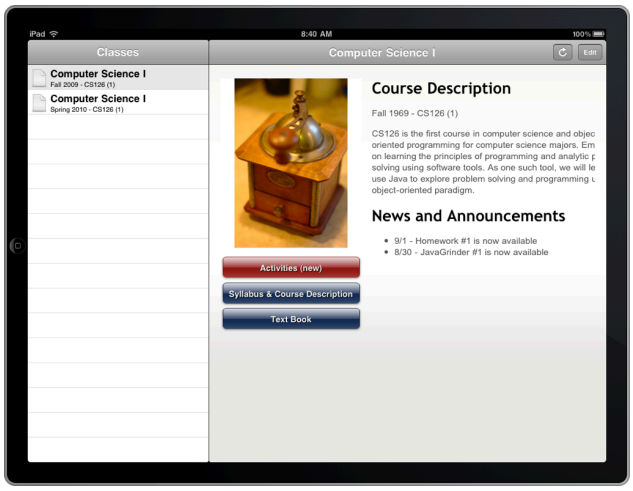

Figure 2. JavaGrinder supports many different services and clients including iOS pictured here.

To address this need, we are integrating extensive support for team evaluation into Javagrinder, a sophisticated system for improving student success in computer science programs being developed with support from the National Science Foundation. JavaGrinder lets students work on programming projects individually or in teams, while automatically evaluating and providing feedback on their work. The overall goal of the JavaGrinder project is to emphasize problem solving, solid software engineering practices, and teamwork, while presenting computer science as an exciting multidisciplinary field rather than an abstract world of syntax and arcane codes. Currently two clients exist for the Javagrinder system: one written in JavaScript that provides the primary JavaGrinder experience on conventional platforms, and a second written in Objective-C that provides an iOS-based client for the iPad.

As part of its core functionality of training students to be highly-functional programmers and software engineers, Javagrinder provides built-in support for a variety of evaluative instruments, including true/false, multiple choice, multiple answer and short answer activities; where possible submissions are evaluated automatically. Teaming is a key activity supported by Javagrinder, it

is in this context that we are exploring automation of the team evaluation schema described in earlier sections. Specific team evaluation features being built into Javagrinder include:

- Teams can be defined by the instructor, or automatically created by the Javagrinder system as students are assigned to programming projects defined in the system.
- All deliverables and deadlines for a project will appear as milestones in an overall project plan established by the instructor.
- Individual teams may flesh out the overall project plan, refining the project phases defined by deliverables with specific tasks related to completing that deliverable. Hardcopy weekly task reports are no longer necessary because task, task assignments, and task completion are recorded continually by the system. A "team status" view will be provided as a snapshot of current status, showing currently open tasks, their assignment to team members, due dates, and completion status.
- Teammates update status of assigned tasks continually, and the system tracks and highlights tasks that become overdue, generating a running rating of performance for each team member.
- Peer evaluations are solicited automatically as deliverables come due. Responses are securely collected online, with combined scores automatically calculated and used to modulate individual scores for deliverables.
- Current course performance information for individual students is continuously available to the instructor, as team evaluations and other evaluative results are recorded by the system.

Because all aspects of the student evaluation system are automated, Javagrinder will be able to help provide dynamic monitoring of teams and individuals for signs of developing trouble. Instructors can be immediately alerted by automated email if a peer evaluation indicates that a team may be dysfunctional (as described in Section 3.4), or if an individual student's current course grade falls below a pre-defined mark.

We expect that automating the process will allow our team evaluation schema to reach its full potential: faculty can include more frequent peer evaluations, maintain strong awareness of internal team dynamics, and automatically generate documentation of individual student contributions and performance (e.g. for ABET), while minimizing the overhead on themselves and students. Automating peer evaluation may also allow us to explore novel ideas. For instance, one interesting possibility is to provide some qualitative feedback (e.g. temperature gauge, smiley/grumpy face) to indicate the general satisfaction of teammates with one's work, without compromising the overall anonymity of peer evaluations.

6.0 Conclusion

Over the past 15 years, team-based design experiences have become a central feature of engineering programs across the United States. Our experience over the past 15 years of offering the Design4Practice curriculum at Northern Arizona University has shown that team-based design is valuable and effective at teaching engineers critical "soft skills" – project management, teaming, interdisciplinary design, and professional communication – that are increasingly important in modern engineering practice. Although many approaches have been tried over the years, accurately distilling individual performance evaluations out of overall team performance

has proven to be a difficult and stubborn problem. In this paper, we have traced the evolution of the team evaluation schema used in our Design4Practice curriculum, explicitly highlighting missteps and shortcomings along the way. Importantly, our schema was developed using a conservative minimalist philosophy of starting with a very simple system, and incrementally increasing complexity only as needed to address specific shortcomings observed. Our goal was to develop a system with high efficacy and fidelity in mapping team performance to individual performance, while keeping overhead for instructor and students to a minimum. The best practices distilled from this experience are summarized in the following points:

- Focus on quantitative measures. Quantitative measures of effort invested are easiest for team members to assess objectively, and can be directly used to modify deliverable scores.
- Anonymity of peer evaluations. Peer evaluations must be anonymous to encourage openness and honesty. There may be a middle ground (explored in our ongoing automation project), however, that allows students to roughly gauge team member satisfaction without breaching confidentiality of evaluations.
- Association with specific deliverables. Peer evaluation should be associated with specific deliverables rather than focused generally on the course or project phases. This allows for accurate portrayal of normal variations in student performance over the course of the project.
- Peer evaluations should be used as direct adjustment *factors* in calculating grades. Peer evaluations must have substantial effect on individual scores to provide meaningful motivation; directly using numerical peer evaluations to apportion credit for team scores on deliverables to individuals can achieve this goal.
- Integration of multiple measures. The use of multiple mechanisms of insight into individual contributions to team deliverables is valuable; adding weekly task reports allowed instructors to effectively monitor and rationalize peer evaluation scores.
- Low overhead for students and instructor. Created a system with low overhead for both instructor and students is critical. If overhead of evaluation is too high, efficacy of the system suffers as student evaluations become hurried and superficial, and instructors have no time to properly monitor incoming results.

Based on these insights, our current initiative to integrate peer evaluation and task tracking in an automated team management system shows strong potential for reducing the overhead of managing the team evaluation process for students and faculty alike, allowing detailed tracking of team dynamics and automated, continuous computation of individual scores from team outcomes. The Javagrinder system promises to integrate performance tracking and other learning support tools used by teams in a simplified project management system used across the curriculum. Development of this system is ongoing, supported by a National Science Foundation grant for developing innovative computational support for STEM education; we look forward to reporting on the complete functionality and efficacy of the Javagrinder system after the system has been fully implemented and evaluated.

Bibliography

[1] B. J. Millis and P. G. Cottell, *Cooperative Learning for Higher Education Faculty*. Phoenix, AZ: Oryx Press, 1997.

[2] R. F. Stein and S. Hurd, *Using Student Teams in the Classroom: A Faculty Guide.* Anker Publishing Company, Inc., 2000.

[3] B. Oakley, R. M. Felder, R. Brent, and I. Elhajj, "Turning Student Groups into EffectiveTeams," *Journal of Student Centered Learning*, vol. 2, no. 1, pp. 9-34, 2004.

[4] D. B. Kaufman, R. M. Felder, and H. Fuller, "Accounting for Individual Effort in Cooperative Learning Teams," *Journal of Engineering Education*, vol. 89, no. 2, pp. 133-140, 2000.

[5] R. W. Brown, "Autorating: getting individual marks from team marks and enhancing teamwork," in *Proceedings of the Frontiers in Education Conference*, vol. 2, 1995.

[6] N. Clark, "Evaluating student teams developing unique industry projects," in *Proceeding of the 2005 Australasian Computing Education Conference*, 2005.

[7] N. Herbert, "Quantitative peer assessment: can students be objective?," in *Proceedings of the ninth Australasian conference on Computing education - Volume 66*, pp. 63–71, 2007.

[8] J. H. Hayes, T. C. Lethbridge, and D. Port, "Evaluating individual contribution toward group software engineering projects," in *Proceedings of the 25th International Conference on Software Engineering*, pp. 622–627, 2003.

[9] D. E. Wilkins and P. B. Lawhead, "Evaluating individuals in team projects," in *Proceedings of the thirty-first SIGCSE technical symposium on Computer science education*, pp. 172–175, 2000.

[10] K. Collier, J. Hatfield, S. Howell, and D. Larson, "A Multi-Disciplinary Model for Teaching the Engineering Product Realization Process," in *1996 Frontiers in Education Conference*, 1996.

[11] S. Howell, D. B. Larson, J. Hatfield, K. Collier, G. Hoyle, and G. Thomas, "An Integrated Engineering Design Experience: Freshman to Senior Level," in *ASEE 1995 Conference Proceedings*, 1995.

[12] E. Doerry, B. N. Bero, D. Larson, and J. Hatfield, "Northern Arizona University's Design4Practice Sequence: Interdisciplinary Training in Engineering Design for the Global Era," in *3rd Workshop on Global Engineering Education (GEE3)*, 2000.

[13] B. Oakley, "It takes two to tango: How 'good' students enable problematic behavior in teams," *Journal of Student Centered Learning*, vol. 1, no. 1, pp. 19-27, 2002.