A Software Project Management Course Role-Play Team-Project Approach Emphasizing Written and Oral Communication Skills

Dr. Sarah L. Sullivan
Department of Computer Science
Indiana University - Purdue University at Fort Wayne (IPFW)
Fort Wayne, IN 46805
sullivan@cvax.ipfw.indiana.edu

SUBJECT AREAS

* Software Project Management.
* Team Project Experience.
* Example of Integrating Written and Oral Communication Skills into a Computer Science Course.

ABSTRACT

This paper describes a role-play team-project approach that provides students with practical software project management experience through team management of a fictitious software project. Under the direction of the Vice President of Software Systems (professor), management teams (students) complete software project management tasks (course assignments) on their own team's software project. These management tasks reinforce many of the concepts studied in the classroom while also cultivating written and oral communication skills.

INTRODUCTION

No two software projects are alike. Innumerable factors contribute to the inherent uniqueness of each software project. Consequently, each project becomes a unique management adventure.

In navigating a software project through unchartered terrain while the project and its environment evolve [Floyd 1988], the software project manager functions as a boundary spanner in an interpersonal communication network that includes groups of subordinates, users, consultants, management peers, vendors, and corporate upper management. Effective communication is essential for software project success. The team role-play approach presented here is rich with communication skills development opportunities.

ACM-24thCSE-2/93-IN,USA

The role-play is integrated into a one-semester capstone course for senior computer science and information systems majors. It takes students on a manager's journey through the software development life cycle from project conception through project post mortem. This journey integrates the student's prerequisite analysis, design, and programming course experiences which serve as a basis for the management experience. Classroom presentations of issues and techniques are sequenced to support this life-cycle management journey. Through the role-play, each team produces its own case study of what is involved in managing a software project. The end-of-the-semester project presentations further reinforce the core concepts while exemplifying the management challenges encountered and the fictitious actions taken in meeting those challenges.

In addition to covering techniques for planning, estimating, organizing, and controlling a software project, a software project management course should give students practice in applying these techniques to a specific software project. At the same time, it should cultivate effective communication skills that are necessary for software project success. The approach described in this paper accomplishes these goals. It has been successively refined through five course offerings since 1988.

Interest in integrating writing into computer science courses has been evident at SIGCSE for the past several years [Falconer & Katz 1992; Pesante 1991; and Jackowitz et. al. 1990]. Hartman and White [1990] also emphasize the need for oral communication skills. The purpose of this paper is to review an example of integrating both written and oral communication skills into a computer science course in the hope that others might consider incorporating similar communication skill development into their pedagogy [See also Sullivan 1990].

SOFTWARE PROJECT MANAGER DELIVERABLES

The lectures and reading assignments are given in a sequence that support the management tasks of a typical software project. The course material is outlined below through an overview of the team-project deliverables.

Project Proposal

Selling a software project to management involves showing that one has a workable, cost-effective, software systems solution to a current or future problem. Teams are instructed that their one-page project proposal should also include a discussion of pertinent technical, management, and financial considerations as well as an assessment of the proposed system's potential for gaining and maintaining competitive advantage (Figure 1). To assist students in defining projects of reasonable scope for the course, they are advised that Progressive Corporation's Board of Directors tend to approve projects that require three subordinates per supervisor and that span between one and two calendar years.

Project Requirements

Before a non-generic project plan can be drafted, a more detailed description is needed. Such detail includes: problem statement, project goals, system's major functions, general inputs, general outputs, standards, performance, growth, operation, environment, compatibility, interfaces with other systems, reliability, availability, human interface, organizational impact, maintenance, support, documentation, training, and terms and conditions of payment. A project requirements document (4-6 pages) is requested via a memo that congratulates teams for receiving approval on their proposed projects.

Project Plan

The project plan deliverable covers the entire software development life cycle from system conception through retirement/replacement. The major components of the plan include: work breakdown structure, project milestones, project schedule, project resource needs, type and frequency of meetings and reports, change control procedure, and assumptions. Each task in the work breakdown structure must be assignable, estimatable, schedulable, and completable. Also, the quality and completion criteria for all tasks and milestones must be stated in measurable terms. The project plan must have a title page, a table of contents, and an executive summary. The preferred organization presents the material in a terse content-packed style with supporting detail placed in appendices.

Staffing Profile

The staffing profiles serve to document the characteristics, skills, project tasks, job design, training needs, and career path potential of the each fictitious staff member recently hired by the project supervisors. This assignment follows approval of the staffing levels justified by the team's project plan. It reinforces lectures on software project team models, personality characteristics typical of software professionals, and job design.

Status Report

Each team presentation reports status from a different point in the software life cycle. The presentations begin with a review of the project's previous life cycle phases, followed by a detailed status report of the phase just being completed, and ending with an overview of the remaining phases. Through these presentations, teams share their case study of what is involved in managing a software project. Students are often surprised by the range of management challenges covered in the presentations.

The detailed status report portion of the presentation focuses on a) plan vs. actual, b) turnover, c) major problems encountered and action taken, and d) pleasant surprises. Each successive presentation reports status at a later point in the life cycle than all the previous presentations. Through this sequencing the presentations double as a review of course content.

Post Mortem

Reflecting on the course content, each student drafts a paper that includes:

a) successes
   i.e. What did you do well?;
b) hindsights
   i.e. Given what you know now, what would you have done differently?;
c) manager insights
   i.e. As a supervisor, what could you do on your next project that would substantially contribute to project success?; and
d) worker insights
   i.e. As a software professional, what could you do on your next project that would substantially contribute to project success?

The industry practice of conducting post morteums helps organizations identify which aspects of the software process to retain and which aspects to change on future software projects.

Peer Evaluation

Each class member drafts a one-page evaluation of the performance of each team member including his or her self. Each evaluation must contain the roles, responsibilities, and contributions of that individual. Each evaluation must also identify at least one strength and at least one area for improvement. Since software project supervisors must evaluate their subordinates to shape behavior and to reward performance with salary increments (decrements) and/or promotions (demotions), this assignment serves to provide practice in giving and receiving such feedback.

TYPES OF PROJECTS

The following are examples of projects that have been "managed" through the role-play:

* an automated home system;
* a cardiology expert system;
* a credit management system;
* a grocery store customer order and delivery system;
* a millennium cruiser that travels back and forth in time through "worm holes" in space;
* an office android;
* a 3-dimensional virtual-reality architect system that lets home buyers "walk" through their custom home before it is built;
* adding code generation capabilities to a front-end CASE tool; and
* the software systems in KIT, the high-tech, talking, self-driving car on the television program Knight Rider.

COMMUNICATION SKILLS PEDAGOGY

Team Role-Play

At the first class session, the roles and structure of the role-play are discussed to set the stage for the team project. The Software Project Supervisors (students) are welcomed to the newly formed Software Systems Division of Progressive Corporation (fictitious company) by the Vice President of Software Systems (professor). The VP informs the supervisors that their first order of business is to propose and get corporate approval for the software projects that they will supervise. Hiring of subordinates will be delayed until after the supervisory team's project plan receives funding. The VP promises to provide guidelines for key management tasks in the form of memos. The VP also promises to meet with each management team on a periodic basis. Then, supervisors are forewarned that they will be presenting their status reports directly to Progressive's VPs and CEO at quarterly project status meetings.

Brainstorming

Group brainstorming is demonstrated as the class dreams up possible projects. This brainstorming exercise gets the creative juices flowing and has resulted in some very innovative project proposals. Some teams have later been observed using the brainstorming technique on their own. After the brainstorming demonstration, the supervisors establish three-person supervisory teams.

Teamwork

Most software projects require a group effort. Indeed, both the ACM [Tucker et. al. 1991] and the DPMA [DPMA 1991] curriculum guidelines emphasize group projects. Students need to develop the interpersonal skills required for group efforts to be successful. Having the students work in semester-long teams incorporates group decision making, group planning, negotiation, coordination of schedules, and peer review into the role-play. Through the project, the team building phases of forming, storming, norming, and performing [Licker 1985 p. 121-124] are both studied and experienced.

The egalitarian structure of the egoless programming team model [Weinberg 1971], which has been found to produce superior results compared to a hierarchical structure [Krasner et. al. 1987, p. 59], is encouraged in the management teams.

Research on gender communication differences [Tannen 1990] reveals that this egalitarian structure is typically familiar to women while typically being totally foreign to most men. Experiencing a truely egalitarian structure is especially important for our male dominated student body and faculty. Also, since mixed gender interactions tend to follow a hieriarchical structure [Tannen 1990], techniques that encourage an egalitarian structure in both all-male and mixed-gender teams are needed. Such techniques are currently being developed [Sullivan 1992].

Business Memos

All assignments are distributed in the form of business memos (Figure 1). The deliverables requested in these memos are to be attached to a cover memo that responds to the assignment memo. This technique gives students practice in reading memos, practice in asking clarification questions to understand what a business memo really requests, and practice writing politically correct memos.

SOFTWARE SYSTEMS DIVISION, PROGRESSIVE CORP., INNOVATION, IN

TO: Software Project Supervisors
FROM: T. H. E. Boss
DATE: January 13, 1992
SUBJ: Project Proposals

Please prepare a one page project proposal for review at the January 27th Board of Directors meeting.

In your proposal, state the problem your proposed system will solve. Show that 1) you understand the problem, 2) have a solution, and 3) the solution will work.

Discuss competitive advantage and include pertinent technical, management, and financial considerations.

Figure 1: Example Assignment Memo.

Meetings with the Vice President

Several class sessions are set aside for team meetings with the Vice President (professor). These meetings provide a forum for team members to ask specific questions regarding the requested management deliverables. In addition to supporting the students' learning of how to read and produce what was asked for in a memo, these meetings provide 1) practice preparing for an effective meeting, 2) practice conducting a brief focused meeting, and 3) practice working with a corporate superior. These meetings also provide opportunities for individual attention and management coaching.

Software Project Management Documents

Much of software project management involves technical writing to communicate with superiors, users, peers, and subordinates. The course assignments provide opportunities to practice conveying ideas clearly and succinctly to these audiences. In addition to encouraging a terse yet powerful natural language style, both the one-page rule and document organization are emphasized.

Simply stated, the one-page rule is:

> If it fits on one page,
> it is more likely to get read.

A one-page document has a better chance of being read to completion between interruptions. It can readily be posted, written on and forwarded, attached to a cover memo, FAXed, or quickly transformed into electronic mail and distributed. Thus, adherence to the one-page rule increases the chances that information will actually get to the personnel who need that information to do their piece(s) of the project.

Whenever possible, the one-page rule should be applied to small documents as well as to sections of large documents. Large documents should be organized with a title page, table of contents, and executive summary. The reaction that "All this technical writing is harder than programming!" underlines the importance of providing opportunities to practice technical writing skills.

Presentations

The presentations (see Status Report above) must be oriented toward an audience of VPs and a CEO. When available, the class presentations are given in a room that is furnished like a corporate board room.

Visual aids are required. Most students use transparencies. Some teams have run their presentations from a PC with the screen displayed via projector.

Presentations are 25-30 minutes. Members of a 3-person team must each speak for a minimum of 5 minutes of air time. Members of a 2-person team must each speak for a minimum of 10 minutes of air time. A one-page handout is required. The handout should be designed to serve as a presentation outline, a crib sheet of important points to remember, a place for the audience to jot down notes or questions to ask later, and an attachment to the meeting minutes.

STUDENT'S EVALUATION

Student grades are based on performance in three areas: examinations (40%), team project (50%), and team presentation (10%).

In grading team projects, evaluation of the performance and contribution of each individual student can be difficult. The peer evaluation assignment normally provides a good indication of how actively each team member participated in the project. In addition, students are encouraged to maintain a CYA (Cover Your A__) log of who-agreed-to-do-what-by-when-and-who-really-did-what. All students have the option of submitting a CYA log to support a petition for distributing team grades among team members in a manner proportionate with each team member's contribution to the team effort. Unless the documentation in the CYA log provides overwhelming evidence to the contrary, all team members receive the same grade for work submitted by the team.

STUDENT RESPONSES

Although sometimes the students complain that there is too much work involved, they have consistently appreciated the application of theory to practice. They have especially appreciated both the opportunity to envision and "manage" an innovative software project.

COURSE TEXTBOOK

Finding a software project management text whose chapters supported the management tasks of managing a software project from womb-to-tomb, was surprisingly difficult. Initially, Fall 1988, a collection of professional papers was used. The following text is the only good match found thus far:

> Rakos, J. J. 1990. Software Project Management for Small to Medium Sized Projects, Prentice Hall.

Von Mayrhauser [1984] serves as a good library reserve supplement to support the material discussed above under staffing profile.

CONCLUSION

The software project management course described in this paper represents an attempt to provide students with a meaningful yet practical experience of what is involved in managing a software project. Its emphasis on both written and oral communication skills represents an attempt to convey the importance of these skills in software project management while providing course relevant opportunities to practice and apply these skills. It is hoped that this paper will provide ideas to those who are or will be teaching a software project management course. It is also hoped that this paper will stimulate experimentation with integrating written or oral communication opportunities into computer science courses.

REFERENCES

DPMA 1991. Information Systems: The DPMA Curriculum for a Four Year Undergraduate Degree. Data Processing Management Association, 505 Busse Highway, Park Ridge, IL 60068 USA.

Falconer, D. R. and Katz, M. 1992. "Building an Infrastructure to Support Writing Across the Computer Science Curriculum." SIGCSE Bulletin, 24(1), pp. 34-37.

Floyd, C. 1988. "Outline of a paradigm change in software engineering." Software Engineering Notes, 13(2), pp. 25-38.

Hartman, J. and White, C. M. 1990. ""Real World" Skills vs. "School Taught" Skills for the Undergraduate Computer Major." SIGCSE Bulletin, 22(1), pp. 216-218.

Jackowitz, P. M.; Plishka, R. M.; and Sidbury, J. R. 1990. "Teaching Writing and Research Skills in the Computer Science Curriculum." SIGCSE Bulletin, 22(1), pp. 212-215.

Krasner, H.; Curtis, B.; & Iscoe, N. 1987. "Communication Breakdowns and Boundary Spanning Activities on Large Programming Projects." Proceedings of the 2nd Workshop on Empirical Studies of Programmers. Ablex. pp. 47-64.

Licker, P. S. 1985. The Art of Managing Software Development People. Wiley.

Pesante, L. H. 1991. "Integrating Writing into Computer Science Courses." SIGCSE Bulletin, 23(1), pp. 205-209.

Rakos, J. J. 1990. Software Project Management for Small to Medium Sized Projects, Prentice Hall.

Sullivan, S. L. 1990. Assessing the Effect of an Instructional Intervention on the Communication Behaviors of Software Engineers. (Doctoral dissertation, Illinois Institute of Technology, University Microfilms International 9030480.

Sullivan, S. L. 1992. Integrating Gender Communication Awareness into a Team Role-Play Software Project Management Course Project. Working Paper.

Tannen, D. 1990. You Just Don't Understand: Women and Men in Conversation. Morrow.

Tucker et. at. 1991. "A Summary of the ACM/IEEE-CS Joint Curriculum Task Force Report: Computing Curricula 1991." Communications of the ACM 34(6), pp. 69-84.

von Maryhauser, A. 1984. "Selecting a software development team structure." Journal of Capacity Management, 2(3), pp. 207-225.

Weinberg, G. M. 1971. The Psychology of Computer Programming. New York: Van Nostrand Reinhold Company.