Managing Expectations: Suggested Guidelines for Student Development of Software for Non-Programming Staff in the University Setting

Caroline Lange
Computer Science Department
University of Northern Iowa
langec2816@uni.edu

Abstract

An on-campus student-programming job for the summer was going to be the perfect opportunity to enhance classroom knowledge with actual experience. Instead, it became the inspiration for a thesis.

Universities depend on students guided by non-programming staff to develop software for university use. The success of these projects is important for the students, staff, and the university, yet these projects are often viewed as failures because they come up very short of the goals. Why? The expectations of the staff "client/user" and the student "developers" are unrealistically high. While client/user involvement is identified as a key success factor in professional projects, it is overlooked for student development teams, and raises the importance of appropriate expectations. This paper is part of a larger body of work, which is still in progress at the time of the proceedings MICS 2001.

Student developers in Project Management and Information Systems Development and their staff "clients" were surveyed to identify their expectations for their software project and to identify what they think are the top three factors affecting software project success. Results of the first survey will be presented. At the conclusion of the project, it is expected that students and clients of teams that log more hours of client interaction will perceive their projects as being more successful than those teams logging fewer hours and their expectations will become more realistic.

Introduction

An on-campus student-programming job for the summer was going to be the perfect opportunity to enhance classroom knowledge with practical experience. Instead, it became the inspiration for my thesis.

The task of updating a university-critical database had actually been started several years ago. (Why was there no documentation?) In fact, two newer versions of Microsoft Access had been released since the project was started and a third version was on the way. (Why was it taking so long?) The Y2K non-compliance of the existing Paradox-based system necessitated the change. (Will it be ready in time?) By May 1999, the project was *already* many months behind. You can probably guess the rest of the story. There were no statements of requirements or specifications, nor any data flow diagrams or state charts. Version control? (What version control?) There was only a brief set of notes from a meeting over a year ago, a pile of reports from the current system, some sample data, and the need to get the job done.

That is not how they taught us to do projects!

Explanation of the Problem

Universities often depend on the work of students along with non-programming staff (those not hired specifically to do programming) to develop software for university use. This work may be purely departmental in nature or may come as part of a computer science or information systems project management course. Either way, these projects often fail or come up very short. Why? Expectations of the staff client/user and the student developers are unrealistically high [2], [5]. Both the students and the staff have to deal with many shortcomings of software projects, as identified in Table 1.

Table 1: Shortcomings of Student-Staff Software Projects.

Student	Staff	Shortcomings of Project
•	•	Lack of experience with software projects/project management
•	•	Requirements not clear, documented or managed
	•	Project is addition to work load
•	•	Project grows beyond scope of available time
•		No one person has the role of project manager
•		Lack of familiarity with development software/platform
•		Lack of development resources (hardware, software, modeling tools, etc.)
•	•	Inadequate delivery of product (partially working, some project deliverables not ready)

Software engineering course work tends to consider success in terms of the technical benchmarks used in industry. As Parker, Holcombe and Bell [7] state: "We tend to ask our students to work for two different masters: lecturers and customers. To some extent we ask them to work on two different agendas: software engineering practice as we would like it to be . . . and software development as it frequently is in reality, with good practice sacrificed to time schedules and financial limitations" (p. 238). In the case of my own summer programming job, my expectations were clearly focused on the former agenda, while reality dictated the latter agenda. While learning did take place, I often felt less than successful about my work.

The expectations for my student programming job experience had been developed through software engineering course work. What forms the basis of client expectations? They may not have a background in software engineering. If my expectations, (formed through coursework) are high, how high might the client's expectations be?

Generally, too much is expected from software [2], [5]. Software projects are expected to be successful, yet project "success" is often relative to our expectations of the project. To be fair, the success of a project must be defined relative to the scope of the project being assessed. If our expectations are too high, i.e., unrealistic, success of the project relative to our expectations may not be possible – regardless of the technical achievements reached.

Defining Expectations, Risk, and Success

Understanding how expectations are defined will move us toward the goal of setting and managing more realistic expectations for the outcomes of student programming projects to be used by university staff. In the context of this study, let **expectations** be defined as *assumptions of what will be true*. However, it is the expectations that are voiced nearer to a project's completion, or not voiced at all, that can cause the most difficulty.

Consider that expectations can be classified by four distinct categories: *time, skill, scope,* and *cost.* When expectations are not met, it is because of overestimation or underestimation of expectations in one or more of the categories

Expectations are fundamental to our perception of how successful software projects are. If there is a question whether or not an expectation can be met, it becomes a risk to the software project. In this context, let us define **risk** as *a factor that, when encountered, can threaten the positive outcome of the expectation*. **Success** can be defined simply as *a positive outcome*. The definition of what constitutes a successful outcome for software development projects in the university setting needs further evaluation.

Why These Projects are Important

Universities and students benefit from using students for campus software projects [4], and their success is important for the students, staff, and the university. Computer science and information systems students can benefit from experiential learning [4] and the opportunity

to succeed on a software development project during their academic career. University staff can directly benefit from the opportunity to participate in the development and acquisition of specialized software that would not be available otherwise [4]. The university itself benefits by providing an opportunity to its students, and the nominal cost of student software projects for university use reinforces the case [4].

In the University of Northern Iowa's Strategic Plan 2001-2006, two specific objectives support collaboration between students, faculty and staff with regard to technology projects. Objective 5.3 seeks to "enhance opportunities for mentoring and social interaction among students, faculty and staff," and objective 7.2 states a desire to "more fully integrate modern technology into the everyday lives of UNI students, faculty and staff" [10, online]. The prominent placement of technology issues in the context of this university's goals should be considered a mandate to encourage success when students and staff collaborate on development of software for the university's use.

There is clearly a benefit to making student and staff software projects more successful while at the same time evaluating what outcomes constitute success for software projects in the university setting.

Purpose of the Study

The purpose of this study is to examine the affect of client/user involvement on software project success in the university setting, as perceived by student software developers and their staff clients. The study also seeks to clarify the relationship between realistic expectations and perceived project success, for both the student developers and their staff clients. University of Northern Iowa student developers and their clients were surveyed twice during the term of their projects. The initial survey was conducted near the beginning of the projects to obtain demographics, indicators of expectations, and ask for participants to identify three of 10 factors they believe affects project success. General demographics include age range, gender, software project experience, work information, and for students, grade level and student classification (full-time or part-time). For the end of project survey, participants will be asked questions designed to measure their perception of their project's success. Students and clients will identify the resulting deliverables and assess their completeness. Students will be asked how much client interaction (in hours) they had, how successful they perceive their project to be and how successful they think their clients will rate the project. In turn, clients will be asked to rate the project's success according to an evaluative framework known as the AEIOU model (accountability, effectiveness, impact, organizational context, and unanticipated outcomes) [9]. This approach will allow clients to respond to the questions in a way that is meaningful to them.

Based on the data gathered from the initial survey, the following research questions will be discussed:

1. Generally, how high are the initial expectations of the students?
2. Does chronological age appear to reflect a trend toward more realistic student expectations in the initial survey?

3. Are the top three success factors selected by students consistent with the top three success factors for industry?

Because the evaluation of student expectations requires a comparison between the initial survey response and the end of project response, it is not reasonable to evaluate the current results beyond their face value. It is expected that both students and clients will be characterized as having high expectations for their projects, as evidenced by a high number of deliverables with high percentages of anticipated completion in the initial survey and respond to the final survey with fewer deliverables and or lower percentages of anticipated completion. Expectations are characterized as more realistic when the responses from the beginning and end of project surveys have minimal changes. Students who are chronologically older are expected to have more realistic expectations.

Overview of the Literature

Industry Overview

In a global sense, the software industry has a bad record of accomplishment of implementing the right software on time and under budget. Public perception of the software industry is negative; the products are slow to market, overpriced, faulty, and aren't "good enough any way" (p. 89) [2]. Perhaps providing a guided opportunity for success in an academic setting is the first important step to correcting deficiencies in the industry by "growing" better software engineers, who in turn "grow" better software.

A review of literature regarding software development/engineering reveals that most of the material is written by the industry for the industry. The scale of the projects is large and while the suggestions for improvement are valid, they do not directly relate to the scope of student projects. The fact that programming projects tend to fail is no real surprise when considered in light of the software engineering industry's reports on successful project completion. According to research published by the Standish Group [10], nearly a third of the planned software development projects in the United States get canceled before completion and 52.7% of projects that aren't canceled are 189% over budget. Clearly, these figures are cause for concern.

Success Factors

Many studies regarding factors for successful projects have been published. Pinto and Slevin noted a need for identification of critical success factors and their interrelationships [8]. They defined critical success as "factors which, if addressed, [would] significantly improve project implementation chances" (p. 22) [8]. They found critical success factors for projects to be "inherently more 'organizational and behavioral' than technical" (p. 22) and focused on the project manager as the person charged with intuitively recognizing when a project is not going well either technically or organizationally [8].

Pinto and Slevin sought to define critical success factors and develop a potential instrument to measure the factors, thereby enabling early diagnosis of problems [8]. Their meta analysis of previous studies identified nine success factors that were common to at least two studies, but the studies were general in nature and the success factors identified were based in theory [8].

To determine if the critical success factors that were supported by existing literature could be empirically proven, Pinto and Slevin's own study utilized a procedure known as Project Echo (developed by A. Bavelas). Reflective responses were elicited from MBA students with project implementation experience regarding steps they could take as project managers to ensure their project's success. The study resulted in a list of 10 critical success factors, which form the basis for Pinto & Slevin's Project Implementation Profile model [8]. An instrument of the same name was developed in an attempt to create an empirical method of assessing the status of any project [8].

While Pinto and Slevin did not intend to determine the strength with which the success factors affect project success [8], the Standish Group's "Chaos " report [10] did just that.

To identify and weigh success factors for software projects in industry, the Standish Group queried 365 Information Technology (IT) professionals. The top ten success factors are shown in Table 3. Consistent with Pinto and Slevin's observations, many of the factors reported are not technical in nature.

Table 2: Standish Group's Success Criteria [10].

Success Criteria
1. User Involvement
2. Executive Management Support
3. Clear Statement of Requirements
4. Proper Planning
5. Realistic Expectations
6. Smaller Project Milestones
7. Competent Staff
8. Ownership
9. Clear Vision & Objectives
10. Hard-Working, Focused Staff

Student Projects

Research conducted at universities in the United Kingdom and the United States tends to focus on student/staff software projects with respect to interventions of project management education, process improvement, and role assignments. Groups who received project management training do better than those who do not receive the training [6], [3]. Those who receive team role assignments again outperform the control group and also those groups that received only project management training [6].

An important success factor that seems to have been ignored in the current literature is the involvement of the client with the student development team. In a student project, the "client" is often a user or domain expert. The interventions tested in the current literature focus specifically on the student developers and exclude the client [7]. Since user involvement is a key success factor in industry [10], it would follow that such user involvement would also be a key success factor for student software development projects. Therefore, clients need to be involved as much as possible throughout student software projects [7].

How Software Projects Differ from Other Types of Projects

Perhaps the best way to discuss the unique difficulties software projects face is to compare software development to construction projects. Many authors have made this comparison, including the Standish Group [10] and consultant/author Steve McConnell [5]. Construction is based on sound engineering principles learned over centuries [10]. If, for example, a bridge collapses, there is an investigation to determine why the failure occurred. Such is not the case with software development [10]. Pinto and Slevin noted in 1987 that project managers face the challenges of "increasing uncertainties in technology, budgets, and development processes" (p. 22), [8].

These same challenges continue today. McConnell notes in software projects, as in homebuilding, it is more important to get the client to consider what they want versus what they need [5]. He goes on to state that "part of our jobs as software developers is to educate our customers so that they understand software development better, which makes it possible for us to live up to their expectations" (p. 244), [5]. Parker, Holcomb, and Bell echo this sentiment by their suggestion that instructors must carefully select clients, help set their expectations, and help clients assess the final product [7].

How Student Projects Differ from Industry Projects

Although financial concerns are not as critical in student projects (because there are generally no budgets), the problems of delivering a usable solution within a restrictive time frame is common to both student and industry projects [4]. Student projects may not have a guarantee of a usable product [3], and any usable product that is produced may not be maintained or developed further [4].

Student software projects are generally team or group oriented, which adds overhead to students, who must coordinate multiple schedules to meet at the same time and in the same place [6]. Group project commitments must be juggled along with other study and work obligations. This is less of a problem in industry projects where developers have their full time to devote to project work [4].

Students are also less likely to have a user or domain expert as a part of their development team, as recommended in rapid development or iterative prototyping methodologies used in

industry [7]. The benefit of involving a real client or user is that it requires the team to engage more fully in developing requirements [3], [7]. The time a university client spends with their development team is usually limited and typically in addition to their daily responsibilities.

A factor that is perhaps common to industrial and educational projects is known as scope or requirements "creep." The client's involvement can encourage students to put forth their best efforts [7]. However, tendencies of students to over-engineer and the client's desire for change can negatively impact a project. "Few students have the maturity of negotiating skills to manage this situation" and clients do not realize how changes impact their project's integrity (p. 237), [7]. Authors Capon [1], Heinrichs [4], and McConnell [5] provide some specific suggestions to address scope creep and other shortcomings that threaten project success.

Following this brief overview of the literature, readers should understand some of the challenges and limitations of software projects involving students and staff in the university setting. Previous studies have shown that chances for success improve when interventions of project management education, process improvement, and role assignments are imposed on student projects [6], [3]. Success factors have been researched [8], [10] and shared in industry. Yet, there is still trouble in the software industry, when "good practice [is] sacrificed to time schedules and financial limitations" (p. 238), [7].

Methods

The current study seeks to examine the affect of client/user involvement on software project success in the university setting, as perceived by two groups, student software developers and their staff clients. Additionally, the study seeks to clarify the relationship between realistic expectations and perceived project success for both groups.

Student participants are currently enrolled in either Project Management (course 810:173) from the Computer Science Department or one of three sections of Information Systems Development (150:125) from the Management Department. The staff participants are primarily employees of the University of Northern Iowa who are not computer programmers per their job function at the university.

A survey developed by the author was conducted near the beginning of the project work to obtain demographics, indicators of expectations, and ask for participants to identify three of 10 factors they believe affects project success. The surveys were distributed to students during their respective class sessions, along with a self-addressed envelope for confidential return via campus mail. Client surveys and return envelopes were distributed to the instructors of the participating class sections and delivered to the clients by either the development team or campus mail.

Identities of the respondents are unknown. Team numbers and class section responses are included for possible use in averaging the responses in aggregate by class and to identify any notable change within a team between the first and last surveys. Demographic data for the

students includes grade level, age, gender, classification (full-time or part-time student), employment status (full-time, part-time or not employed), whether or not their employment relates to their course of study, their class section, team number, and number of persons on the team. Demographics for the staff will include gender, employment status, and team number.

Results from the first student surveys are included and will be discussed. Return of the client surveys was slow. Follow-up email reminders sent to the clients resulted in a few more submissions, however not enough responses were received for consideration of discussion at this conference. It was decided that telephone contact with the clients to answer the final survey questions would yield better results.

Overview of the Projects and Teams

Classifying the projects in work for the Spring 2001 semester is difficult, as there are several instances of one client with a large project that has been divided into several smaller projects. Out of 19 projects, more than half involve development of websites and half of those involve database connectivity via the web.

Students in the Project Management class are allowed to indicate the students they would like as teammates, with the final team assignments determined by the instructor. The teams are then allowed to indicate their project preferences, again with the final assignments made by the instructor. Teams are allowed to decide individual responsibilities or roles for each member. Depending upon the size of the team, some individuals may have more than one responsibility to the team. Roles include project manager, software requirements lead, design lead, user or maintenance manual lead, and test document lead.

Students in the Information Systems Development course receive team and project assignments by their section's instructors. The instructors attempt to balance the teams for skills that are appropriate to the project. The instructor also assigns individual roles.

Discussion of Preliminary Results

Out of 98 student surveys circulated, 68 were returned, which is a 69% response rate. As previously mentioned, client surveys were not considered for presentation at this time. Table 3 contains the general demographic information. There are no unusual responses. The typical profile for a student in the study is: male, 20-22 years of age, full-time senior who works part time in a job that relates to his field of study.

Table3: Student Demographics.

Demographic	Category	Percentage
	(n = 68)	
Gender	Male	69.0
	Female	31.0
Age	20-22	54.0
	23-29	33.8
	30-39	4.0
	40+	5.8
Grade Level	Sophomore	0.0
	Junior	1.0
	Senior	85.0
	Graduate Student	8.0
	Other	3.0
Student Classification	Full-time	89.0
	Part-time	10.0
Work	Full-time	10.0
	Part-time	71.0
	No Job	19.0
	Work relates to study?	
	Yes	73.0
	No	27.0
Course Enrollment	Project Management (CS)	40.0
	Info Systems Development (MGT)	60.0
Average Size of Student Teams	3.79 Students	

Research question #1: Generally, how high are the initial expectations of the students?

Students were asked to identify their team's deliverables and indicate the percentage of completeness they expect to have from 0% (no deliverable of this type) to 100% (complete deliverable) with 25% intervals. It is possible but not required that all teams will have all eight of the identified deliverables. Identification of a high number of deliverables with high percentages of completion indicates high expectations about the amount of work to be produced. More realistic expectations are indicated by a lower number of deliverables with the percentage of completion between 50% and 100%.

Table 4 displays the percentages of all students indicating the deliverables and the respective completeness. Based on this data, it appears that on-line help, maintenance manuals, and installation instructions are less likely to be included as deliverables in student projects. These results, in their aggregate form and without a comparison to end of project results, are at best incomplete. At this point, the student's expectations appear to be more realistic than originally anticipated. It may not be feasible to measure student or client expectations for

this study with the existing survey. Such a determination cannot be made until the final survey results are tallied. However, if it is possible to evaluate expectations more conclusively by comparing the initial survey results to the final survey results, it is expected that there will be a general lowering of expectations.

Differences in terminology and requirements between the participating computer science and management course may have affected the student responses to the deliverables section of the survey. For example, some respondents who indicated they were delivering a web site did not indicate test cases or software requirements specifications or software itself as deliverables. In Project Management, a CS course, all of these deliverables would be expected.

Table 4: Proposed Deliverables

	Deliverables							
	Software	User Manual	On-Line Help	Maintenance Manual	SRS	Installation Instructions	Web Site	Test Cases
% Complete								
100%	60%	47%	10%	21%	51%	32%	50%	49%
75%	10%	6%	10%	10%	6%	1%	6%	7%
50%	7%	7%	3%	16%	7%	7%	7%	7%
25%	0%	3%	7%	9%	3%	6%	4%	3%
0%	22%	37%	69%	44%	32%	53%	32%	34%

Research question #2: Does chronological age appear to reflect a trend toward more realistic student expectations in the initial survey?

At this time, age does not appear to reflect a trend toward more realistic expectations for students. Again, a conclusion regarding this research question cannot be reached until the final survey results are available.

Research question #3: Are the top three success factors selected by students consistent with the top three success factors for industry?

Students were asked to select what they believe to be the top three project success factors from a list identified by the Standish Group (1995). Those who identify the top three factors as ranked according to the findings of the Chaos Report (Standish, 1995) will be categorized as having an awareness of what makes a project successful. Surprisingly, students did select 3 out of 4 of the top success factors, as shown in Table 5. Results from this survey data support the idea that the importance of client/user involvement in student software projects is comparable to its importance to industrial software projects.

Table 5: Initial Rankings of Success Factors by Students

Success Factors (listed in same order as on questionnaire)	# of Students Ranking Each	Avg. Student Ranking	Standish Group Ranking
Hard-working, focused staff	31	4	10
Clear statement of requirements	35	1	3
Ownership	0	9	8
Proper Planning	33	2	4
Client/User Involvement	32	3	1
Clear vision and objectives	26	5	9
Smaller Project Milestones	4	8	6
Realistic Expectations	16	6 (tie)	5
Competent Staff	16	6 (tie)	7
Management Support	8	7	2

Concluding Remarks

At the conclusion of the study, we'll be assessing how successful students and clients perceive their projects. We expect that students and clients of teams that log more hours of client interaction will perceive their projects as being more successful than those teams logging fewer hours. It is believed that more client interaction provides an opportunity for the teams to manage the expectations of the client.

An intention of the study is to formulate guidelines for student development of software for use in the university setting. These guidelines should consider typical shortcomings of student-staff software projects (as noted in Table 1) and the established success factors identified by the Standish Group [10] (as noted in Table 2). Authors Capon [1] Heinrichs [4], and McConnell [5] provide some specific suggestions to address these issues. We'll address one specific suggestion here.

The first two shortcomings listed in Table 1 are "lack of experience with software projects/project management" and "requirements not clear, documented, or managed." These items affect both students and staff. McConnell [10] suggests that it is our responsibility as software developers to educate the customer. In the university setting, we cannot rely on students, who are involved in their own education, to provide their clients with information that fully appreciates the client's role in the software development process. Client education that includes an overview of the software development lifecycle and addresses requirements discovery, ownership, and clear vision and objectives, would satisfy the two deficiencies and reinforces three success factors. As this study continues, more guidelines appropriate to student-staff software project shortcomings will be outlined to demonstrate how they reinforce respective success factors.

With the survey results available at this time, definitive conclusions to all the research questions posed here, and those to be contained in the larger work, cannot be made. However, it has been shown that the initial expectations of the students appear to be more realistic than originally anticipated. After the final survey, a comparison of the number of realized deliverables and their relative completeness would yield a more valid assessment of student expectations for their projects. The final survey will also assist us in formulating answers to other research questions.

Supporting the aforementioned finding, this study has also shown that students can identify success factors that correspond to the top two success factors for industry. This result further supports the notion that the student's expectations are not as unrealistic as originally thought.

References

1. Capon, P. (1999). Maximizing learning outcomes of computer science projects. Computer Science Education, 9(3), 184-199.

2. DeMarco, T. (1993). Why does software cost so much? IEEE Software 10(2), 89-90.

3. Hagan, D., Tucker, S., & Ceddia, J. (1999). Industrial experience projects: A balance of process and product. Computer Science Education, 9(3), 215-229.

4. Heinrichs, L. R. (1987). Meeting user needs through in-service student projects. ACM-SIGUCCS, XV, 261-256.

5. McConnell, S. (1996). Rapid development: Taming wild software schedules. Redmond, Washington: Microsoft Press.

6. Mennecke, B., Bradley, J., & McLeod, M. (December 1998). Making project groups work II: The Impact of group process training and role assignment on the performance and perception of information systems project teams. In, Proceedings of the 13th Annual Conference of the International Academy for Information Management, 110-121. Helsinki, Finland: IAIM. (ERIC Document Reporduction Service No. ED 431 423).

7. Parker, H., Holcombe, M., & Bell, A. (1999). Keeping our customers happy: Myths and management issues in "client-led" student software projects. Computer Science Education, 9(3), 230-241.

8. Pinto, J. K. & Slevin, D. P. (1987). Critical factors in successful project implementation. IEEE Transactions on Engineering Management, 34(1), 22-27.

9. Sorensen, C. K. (Fall 1997). Localizing national standards for evaluation of distance education: an example from a multistate project. New Directions for Community Colleges, (99), 53-62.

10. Standish Group, (1995). Chaos [On-line]. Available: http://www.standishgroup.com/visitor/chaos.htm.

11. University of Northern Iowa (July, 2000). "Focused on Excellence" University of Northern Iowa Strategic Plan 2001-2006 [On-line]. Available: http://www.uni.edu/pres/2001-2006strategicplan/.