BUILDING COMMUNITY SERVICE PROJECTS EFFECTIVELY

Lisa MacLean and Michael Werner
macleanl@wit.edu, wernerm@wit.edu
Department of Computer Science and Systems
Wentworth Institute of Technology
Boston, MA 02115

ABSTRACT

This paper presents an approach to using real projects for non-profit agencies in computer science courses. The goals are to deliver a useful project to our non-profit clients while providing a valuable learning experience to our students. The paper builds on failures and difficulties encountered in past community service projects. Risk factors are identified and measures devised to counter them. These measures include developing a systematic approach to managing such projects, enhancing course offerings to better prepare students for working on real projects, and building a repository of practical knowledge within the department. Proper communication between students within teams and between other teams, as well as timely feedback to faculty, and better preparation and communication in the community agencies is essential for effective projects. We also propose building an application framework, reusing proven design and implementation techniques to exploit the fact that many of our non-profit clients have similar requirements.

1. INTRODUCTION

In an increasingly competitive environment, students must be able to produce commercial quality applications before they graduate. Our college has taken a multi-pronged approach to this: two mandatory cooperative placements; a required laboratory session for each course so that principles and concepts in the lecture can be applied; and a senior capstone course where each student is expected to independently conceive, design, write and document a project of significant quality.

To give our students the edge when seeking employment we decided that real projects, or service learning, would be injected into all courses that could benefit by this addition. A new computer networking and database laboratory was built with one grant, and an office of Service Learning and Civic Engagement was created by another grant. There were successes and failures by students, faculty and organizations chosen for the service learning paradigm. Despite initial difficulties, the school remains resolute in its commitment to providing live projects for local needy non-profit organizations, and to students willing to take on the extra work.

Our computer science department currently offers two programs leading to the BS degree: BCOS (Computer Science) and BCNS (Computer Network and Information Systems). This paper relates the department’s recent experience in service learning.

While enjoyable and valuable to our students, the community projects have had their frustrations. In part this was due to the severe financial and time constraints, which are the norm in this type of work, but some frustrations were a product of our own inexperience, particularly regarding project management issues of planning, time estimation, coordination and control. In the end, almost all of the projects were completed; our customers are satisfied and are using the systems we developed. Credit for this goes to the dedication and perseverance of our students, who selflessly put in many hours of their own time.

We systematically reexamined our approach to building these projects. We needed to find what delayed projects, and devise means to bring them to successful completion. By studying past projects we have identified a number of risks. Some reflected weaknesses in our curriculum and even in the faculty's preparation. Others were attributed to failures in communication and project management. We identified opportunities for improved approaches by reusing successful techniques from earlier projects. Ultimately, we are working to extract core design and implementation templates into an *application framework*, based on the idea that many of the requirements in non-profits are similar in nature.

1.1 History at our college

In recent years we have undertaken several successful community service projects including[1]:

- A database system designed to track and report on the delivery of services to homeless people by a local shelter (*St. Francis House*)
- An online application process for students seeking training at *The Nelson Mandela Training center.*
- A database of available day care slots for the *Parker Hill/Fenway ABCD.*
- A web-enabled application for Christmas gift sponsorship of children for the *New England Home for Little Wanderers*.
- A registration system for courses and other services offered by the *Urban League of Eastern Massachusetts.*

1.2 Benefits of Community Service Projects

The community service projects have been extremely motivating to students who usually put in extensive time on them. The projects are harder than toy problems, almost always involving unforeseen complexity. They are sizable, requiring solution by one or more groups, which must coordinate their efforts. Finally, the work is by its nature, original for our students.

Projects done for non-profit organizations can be used to satisfy the community service requirement that many colleges have for graduation. Participation in the work of non-profits makes students more aware of the social and ethical implications of computing. Such understanding is now mandated by the ABET, CAC accreditation criteria.[2.] Whereas some other schools offer a separate course in service learning[3], our model is to incorporate it into the project portion of existing courses.

1.3 Difficulties and Constraints

Our semesters last 15 weeks. Time is needed at the beginning of the semester to organize the student groups and interview the clients about their requirements. The requirements document must be validated by the client. This leaves little time for design and implementation. Only small projects are feasible for completion in one semester.

The students are undergraduates with limited experience, if any, in building software systems. Although they have already taken programming, database and systems analysis courses, for the most part they are learning as they go along. A few students are able to use their cooperative education (coop) experience to advantage; this input has proven invaluable in bringing projects to a successful conclusion.

Cooperative education is one of the distinguishing features of our college. Students are required to complete two 4-month jobs in industry in order to graduate. Junior students go on coop in the spring, seniors in the fall. However, significant community service projects necessarily span more than one semester. With coop, projects begun by one set of students are continued by an entirely different set the next semester.

2. A PROJECT AT RISK – ULEM

In 2003 we undertook a community service project which serves to illustrate some of the difficulties these projects often engender. The project was for the Urban League of Eastern Massachusetts (ULEM), an interracial, non-profit, community-based organization, which provides programs of service and advocacy in the areas of education, career/personal development and employment for African Americans and other residents of color[4]. Currently, ULEM's primary focus is Roxbury, a traditionally low-income section of Boston. Community residents are provided with job training and professional skills to start them on a career path**.**

ULEM depends on financing from a mix of government agencies, foundations and private donations. The varying reporting requirements of its fund sources require ULEM to carefully track its programs, capturing needed data[5]. Additional data is needed to support day-to-day operations such as enrolling students into courses and tracking attendance and progress. The aim of the project was to replace the former paper-based system with a database system using a web-based interface.

2.1 Project Initiation – Spring 2003

The project was initiated in January 2003 at ULEM's request. ULEM had learned of our community service program from the director of the Nelson Mandela Learning Center. For the Mandela project students had programmed five different applications against a large, newly designed relational database[6].

Two faculty members, one a specialist in database design, and the other a specialist in software design, agreed to take on the ULEM project in collaboration. Unlike the Nelson Mandela project, the project would be very large in scale and tie together several different departments. The back-end database design was to fall on students from a sophomore-level database management class, while the front-end web interface design, over-all analysis and design tasks were undertaken by a group of six students in a senior-level software design and development class.

It was assumed that students would sit in on other team meetings and informally coordinate, with the two professors providing overall coordination. In fact, communications between teams proved to be a problem. Many of our students work outside jobs to pay for college, and the classes' lecture and laboratory dates and times did not allow the two course sections to meet during school hours.

There was also a complete lack of overall project management. The front-end team produced a Gantt chart as part of their course requirements, but this was more a formality than a working document, and was never shared with the other teams. The database teams often disagreed on the design making it difficult to produce an integrated database.

The two faculty members took different approaches towards incorporating the projects into their curriculums. The database professor was interested in involving every student in the class in the database design. This way, the project could be used as a running example in the lectures and demonstrate the difficulty of large-scale database design. In the software design class, by contrast, this project was only one of a number of projects taken on by various student teams. Some other projects, for example, involved building games and programming handheld devices such as cell phones. This professor's goal was to show that common software design techniques could be used in building different kinds of applications. He devoted little lecture time to discussing the project, leaving it to the student team to present their work during laboratory periods. The team made three formal presentations as well as several informal progress reports.

To establish requirements, several meetings were held at ULEM's headquarters in January and February 2003. Primarily front-end team members attended these meetings, with some representation from back-end teams. On the ULEM side were those most familiar with the programs to be automated and with their existing computer systems. These meetings, though short, were fruitful. Students came prepared with lists of questions as well as use case diagrams and prototype screens. The ULEM staff supplied existing paper forms and answered questions. Eventually, ULEM identified six programs to be automated:

1. EPST (Employment and Professional Skills Training Program)
2. SCSP (Seniors in Community Service Program)
3. Parent Involvement Program
4. Youth Program
5. Technology Training Program
6. Volunteer Program

The key programs were the training programs. To manage these, ULEM employs four levels of personnel:

1. **Front-Line Staff** – Clerical staff helping walk-ins fill out request forms for entry to the programs.
2. **Coordinators** – Managing scheduling and administration of courses, approving enrollment, and recording student attendance and progress.

3. **Managers** – Responsible for reporting on courses to funding sources.
4. **Database Administrators** – Responsible for maintaining the integrity and availability of the database and other computer resources.

The software needed to limit access on a need to know basis. This concerned both the back-end teams which needed to limit access to the database tables, and also the single front-end team, which needed to limit access to web pages.

The front-end team produced a requirements document in February, 2003 containing use case diagrams, scenarios and a diagram outlining the web pages to be built and showing navigation between them. This document was made available to the back-end teams and to Urban League staff. This back-end database was to be Microsoft's SQL Server®, and the front-end web pages would be built using Active Server Pages (ASP) scripting.

Meanwhile the back-end teams were dividing the applications up into logical units based on ULEM's 6 programs. The designs for the smaller units would later be merged to create the global database model. They were also creating data field names, determining data types and lengths. An Entity-Relationship diagram (ERD), was produced, depicting over 200 tables.

The front-end team produced a design document in March, 2003. Among other things, this document spelled out the logic of the processes of requesting entrance to a course, being approved for enrollment, having progress notes added in, and so forth. Appropriate web forms were designed for accomplishing these tasks. The team was interested in prototyping the system, but unfortunately the back-end database was not yet available, so testing a prototype would require the front-end team to create a mock database themselves.

The team had another problem – lack of a suitable testing environment, namely a machine running the SQL Server database as well as the web server. Although the department had obtained machines and software for this purpose, there were issues involving allowing students access to them. The students resorted to an improvised testing environment using a trial version of SQL Server and an IIS web server set up in a dorm room.

The back-end teams decided to use stored procedures to limit database access. Stored procedures are very secure since they are stored on the centralized back-end, and rights to stored procedures can be granted to users even when the user does not have rights to the underlying tables being accessed[7]. This simplified the front-end task; they could call the stored procedures even while the underlying database was in flux. But the back-end team provided only a few stored procedures. The semester was in its final two weeks so the front –end had to mock up a database to make its final presentation.

The Project Continues - Summer 2003

One member of the spring 2003 front-end team, agreed to continue the project during the summer as his senior project. He concentrated on the most important program offered by ULEM, namely the EPST. He wrote and tested ASP scripts to support this program. However, the stored procedures he used were sometimes unreliable, and indicated weaknesses in the design of the base tables.

The Project Continues -Fall 2003

A former member of the back-end team continued the project. His main goal was to redesign the back-end database, which at this point still had almost 200 tables and a great deal of redundancy due to conflict in the database class teams. Of course this also meant redoing many of the stored procedures.

The Project Continues -Spring 2004

A new front-end team was constituted in the software design class. This team was provided with the existing code and documentation, in particular, the ASP scripts and the database design. However the scripts covered only the first level of intake forms, ULEM actually had three stages in their intake process, each with their own forms. So, significant additional scripting was required. Also, the database had changed since the scripts were written.

This new team had greater expertise in using PHP than ASP. PHP is a free open-source scripting language that is very popular with students[8]. Since the scripts needed to be reworked anyway the team switched to PHP. By the end of the spring 2004 semester, they had produced a working prototype. In March of 2004 it was installed on ULEM's server for on-site testing. ULEM staff now began to provide bug reports on roughly a weekly basis.

However, ULEM now requested a change in the method for authenticating users and assigning privileges to them. Since the project was initiated ULEM had changed its systems to use Microsoft’s Active Directory. Users within the ULEM building are authenticated when they log on to their computers. The team was requested to conform its system to Active Directory to avoid the need for a double log-on.

The Project Concludes -Summer 2004

One of the members of the spring 2004 front-end team, volunteered to continue the work as his senior project. He handled the bug reports that had accumulated from ULEM. He also changed the authentication to align with Active Directory. This made it feasible for ULEM to successfully change over to the new system.

2.2 Risk Factors Identified

From our experience with the ULEM project as well as others, we have been able to identify a number of risk factors for community service software development.

- Communication can break down, even in a small organization.
- The development team may lack needed skills.
- Projects may fail to progress in a timely fashion.
- Project participants may have different goals and time schedules.
- Integrating work done by separate groups is problematic.
- If projects drag, clients are likely to change their requirements.

3.0 A BETTER APPROACH TO COMMUNITY PROJECTS

For a charity organization to get its software built by students is like a person going to a dental school to get her teeth fixed. The process may take a bit longer, she will come in contact with a number of students with varying degrees of experience and expertise, but she can expect that eventually the job will be done right. Best of all, it's for free. Still, we feel the process can be improved to both provide better service to our clients and a better learning experience to our students. This section outlines our current approach.

3.1 Working with Clients

An ongoing relationship must be maintained with clients starting from the initial contact and project proposal, continuing at least until the project is finished, installed and tested on the client's machines. Ideally the relationship should be maintained well after the project is deployed to handle newly discovered bugs and evolution of the client's requirements. Since students come and go, the primary responsibility for maintaining the client relationship necessarily falls on faculty and staff. We have found it useful to prepare a client contract specifying:

- The developer's commitment in terms of included features and time line.
- The client's commitment in terms of resources and availability of contact personnel.

The contract is in two phases: 1- A proposal sketching the nature of the work, an anticipated time line, and general language sketching the obligations of both parties. 2- A requirements specification containing a list of testable requirements, both functional and non-functional, as well as a proposed development plan.

3.2 Enhancing Course Offerings

Students need to know how to analyze problems, design solutions, and execute them using state-of-the-art programming languages and development environments. Fortunately our institute has always emphasized hands-on application of theoretical knowledge. Nevertheless, experience with outside projects has revealed some weaknesses in our curriculum. Hence, we are implementing the following changes:

- A new course in developing WWW applications. Students are taught the principles of client-side and server-side scripting[9].
- Changes to the software design and development course to emphasize multi-tiered architectures including layers for front-end web access, back-end database, and middle layer business logic.
- A new course in project management. (See below)
- Incorporation of project management techniques into all project courses.

3.3 Managing Projects

Much of the risk associated with community service software development can be mitigated using proper project management. Project management activities include:

- Estimating costs and benefits
- Work breakdown into phases and tasks
- Assigning personnel to tasks
- Coordinating
- Assessing risks
- Managing resources
- Monitoring progress
- Managing artifacts
- Tracking costs and resources

To this end we now require students in our BCNS program to take a course in Project Management. Students will learn the concepts, and also the use of tools such as Microsoft Project. In addition, all projects courses will require student teams to document their use of project management techniques.

3.4 Faculty Preparation and In-House Expertise

Faculty members play key roles in community service projects:

- They need to make sure they are equipping the students with necessary skills when they plan curricula, devise syllabi, choose textbooks, etc.
- They directly supervise the projects providing over-all project management. They assess the difficulty of proposed projects, match projects to student capabilities, monitor progress, review documents, enforce deadlines and provide feedback
- They act to resolve disputes between team members, and between teams and the organization being serviced.
- They provide continuity for projects that span multiple semesters and multiple course offerings.
- They maintain contacts with clients and work with professionals and staff at their institution charged with fostering community service.

To carry out their roles, faculty involved in community service projects may need to upgrade their own skills. For example, in:

- Current software design techniques.
- UML conventions for modeling systems.
- Scripting and database languages such as SQL, HTML, Javascript and Perl.
- Project Management including the use of software such as Microsoft Project.

3.5 Application Frameworks

An *application framework* is a reusable partial application that can be specialized to provide custom applications[10]. Application frameworks have been used in industry for almost 20 years, so it is important that computer science students employ this tool in software development. Our department has begun building an application framework for community service projects. The nonprofits for which these projects are built share certain common traits: They

- Provide services to diverse clients, free or at reduced cost.
- Employ both paid staff members and volunteers.

- Receive funds from a combination of government agencies, foundations and individual contributors and need to report on how the funds are used.
- Need to manage day-to-day operations such as scheduling classes and appointments
- Require different levels of access for different groups of users
- Prefer common web interfaces for use internally by the staff, and externally by contributors, clients and the public.

An application framework can provide partial functionality to meet these common needs. The framework can then be customized to meet more specific requirements. The application framework currently under development is marked by:

- A layered architecture with 5 layers
- Selection of a few proven languages for implementation

A Layered Architecture

A layered architecture (Figure 1) allows decomposition of the total project into subtasks and lessens the need for communication and coordination between the various groups working on different aspects of the problem. Each layer need only coordinate with the one or two layers adjacent to it.

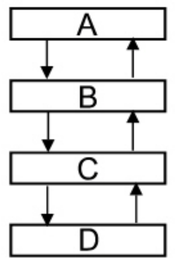

Figure 1 A closed layered architecture. Each layer receives services from lower layers and provides services to higher layers.

A typical project involves a back-end database component, front-end web access by different classes of users, and a back-end business logic that guides the responses made to requests by users. At the very least, this calls for 3 layers as shown in Figure 2.

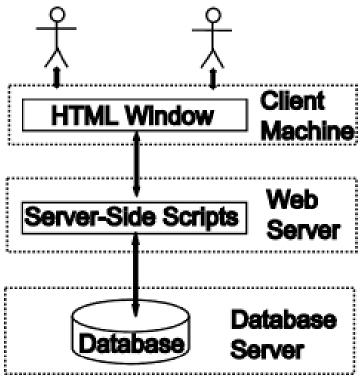

Figure 2. A 3-layer architecture for web-based applications

We found that 3 layers aren't enough. We often encountered these problems:

- Database design is difficult. Even a design that is functional in the sense that all queries can be satisfied may not be optimal. It may have redundancy and respond too slowly. Requiring the front-end team to tightly coordinate with the database designers can significantly delay a project.
- Designing professional-looking HTML interfaces is difficult. The web pages should conform to a common look and feel. Navigation should be intuitive for all classes of user. At the same time, the organization has business rules that indicate how to create responses to user requests. It is a mistake to mix up the processing of business rules with the task of formatting consistent pages.

The solution to these problems is to add two additional layers to the architecture as shown in Figure 3.

The Stored Procedures Layer

Stored procedures are provided by many database management systems to facilitate data extractions, insertions, updates and deletions. The user of a stored procedure need not know the actual structure of the underlying base tables in the database. A single insertion via a stored procedure can result in multiple operations on several base tables. Stored procedures can be designed directly from the requirements specification of the project. Once the specified functionality of each stored procedure is agreed upon, the back-end and front-end teams can go their separate ways, the back-end team designs the base tables and implements the stored procedures, and the front-end team calls the stored procedures in its programs and scripts.

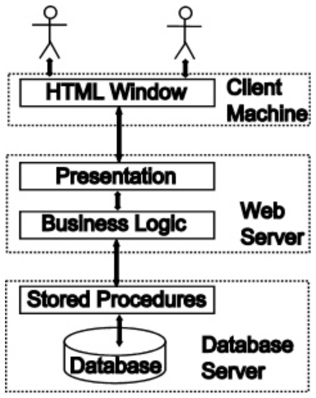

Figure 3. A layered architecture with separate layers for presentation and business logic.

Here are some added benefits of using stored procedures:

- Limiting database access to stored procedures allows tight security control.
- The database can be freely modified as long as the stored procedures are updated to continue to meet their specification.
- Users of stored procedures do not need to know how to do complex SQL queries.
- Data rules can be enforced to make sure related base tables are updated in consistent ways.
- Stored procedures are faster than embedded code or scripting, since they are stored in a compiled form.

The Business Logic Layer

Server-side scripting is separated into two layers:

- Presentation layer: Responsible for formatting HTML pages to convey the response to the user.
- Business logic layer[11]: Responsible for determining the content of responses according to the policies and procedures of the organization.

For example: In the ULEM project, there were rules for determining whether a person was eligible to be considered for enrollment in a program. The logic of this determination was somewhat complex and depended both on information submitted by the user requesting enrollment, and the user’s profile and history stored in the database.

3.6 Implementation Languages that Work

Although numerous programming languages are available with which to build applications, it is impossible to teach all of them. Here is a short list of languages, which in our experience are adequate for building solutions for our clients:

- SQL – Needed for the database back-end and to build the stored procedures.
- PHP – One of several scripting languages available for server-side programming. PHP has strong database interfaces. Some other possible scripting languages are Perl and ASP.
- HTML – Needed for client and server side web programming.
- Javascript – Useful for client-side checking of requests prior to submission.

4. CONCLUSION - WHAT HAS BEEN LEARNED

The use of real projects has a place in academia. Faculty hone their skills. Students get a better experience, one that has led to coops and permanent placement with participating agencies. They have a real portfolio to show potential employers.

Colleges exist in communities, and absorb resources within them. Examples are increased traffic, parking issues, housing issues, and noise. Colleges in lower socioeconomic areas can cause resentment within the community. Giving back is not only good policy, but meets a crucial need and exposes students to the positive effects of serving others. Some students have changed career paths after working with these kinds of organizations.

However, once service-learning is undertaken, it must be tightly managed and defined, which consumes time from faculty involved. The costs should be taken into account up front. Students can grow discouraged when projects fail; so can the users in the nonprofit organization. Being aware of conflicts, communication issues, and feature creep is vital. Well-defined roles and deadlines are imperative, as well as keeping lines of communication open between faculty, students, and nonprofit technical and other staff. Using a modular design for implementation can avoid repetition of code and easier expansion of the systems.

4.1 Future Directions

We will continue using service learning projects. Ideas being discussed include having the preliminary systems analysis done in the sophomore level Systems Analysis and Design course so that application and database developers can start in immediately.

There is now an office of Service Learning and Civic Engagement on campus that can assist the faculty in procuring projects and provide administrative assistance to faculty to ease their workload. We hope to expand the service program to agencies further from campus, perhaps even overseas, to further broaden students' experiences

REFERENCES

[1] Lisa M. MacLean & Michael Werner, *Student Satisfaction Using Real Projects For Object-Oriented and Database Design*, International Conference on Informatics Education & Research in Seattle, Washington, December 2003.

[2] Accreditation Board for Engineering and Technology, Item IV-17 in *the 2000-2001 Criteria for Accrediting Computer Programs*. http://www.abet.org/images/Criteria/cac_criteria_a.pdf.

[3]P Sanderson and K Vollmar, *A Primer for Applying Service Learning to Computer Science*, SIGCSE Bulletin, Volume 32, Number 1, March 2000.

[4]http://www.charityamerica.com/charities/infocauseway/CharityAbout.cfm?CharityID=192

[5] R. Peter Heine, *Community Services Needs Assessment: An Innovative Approach*, Developments in Business Simulation & Experimental Learning, Vol. 24, 1997. http://sbaweb.wayne.edu/~absel/bkl/vol24/24an.pdf.

[6] Lisa M. MacLean & Thomas Pencek, *Benefits And Difficulties In Use Of Real Projects For Advanced Database Applications*, 17th Annual Conference of the International Academy of Information Systems in Barcelona, Spain, September 2002.

[7] http://uk.builder.com/architecture/db/0,39026552,39256058,00.htm

[8] http://www.php.net

[9] http://www.wit.edu/prospective/catalog/11-Sec_%20C.pdf, p. 22.

[10] R. Johnson & B. Foot, *Designing Reusable Classes*, Journal of Object-Oriented Programming, Vol. 1, No. 5, pp. 22-35, 1988.

[11] http://www.woodger.ca/archweb.htm