Project-Based Courses

Alec J. Engebretson, Ph.D. and Mark M. Meysenburg, Ph.D.
Department of Information Science and Technology
Doane College
Crete, NE 68333
aengebretson@doane.edu mmeysenburg@doane.edu

Introduction

Instructors often assign projects. The process of completing a project provides many benefits to students However, projects are limited in size. A project is typically one component of a course, often performed towards the end of the course after the student has learned most of the material needed to complete the project. We term these *project-supplemented courses*.

Information technology professionals are constantly learning new concepts, tools, and techniques in order to complete assigned projects. This learning often occurs *not before* the project, but rather *during* the project. Therefore, in order to be successful professionals, computer science and information systems students must develop confidence and skill in learning new concepts, tools, and techniques and immediately applying them in solving problems associated with assigned projects.

In an effort to create such an environment in the classroom, one in which students develop confidence and skill in learning and applying new concepts, tools, and techniques *during* a project, the Doane College Department of Information Science and Technology (IST) has developed and offered four *project-based courses* over the last two years.

The purpose of this paper is to share our experiences with project-based courses. We hope that it contributes to a dialogue in which others share their experiences in this area.

Specific topics of the paper include:

- a description of project-based courses emphasizing their characteristics and benefits,
- an overview of our experiences with four project-based courses offered at our institution, and
- a discussion of what we have learned, both good and bad, about offering project-based courses.

Project-Based Courses

While there is not a definition of project-based courses, important characteristics include:

- The main objective of the course is to complete a relatively large project, working through all phases of a development life cycle. Note the emphasis is on the process not the product.

- All course activities support the completion of the project. Student learning occurs through the activities. These activities include, but are not limited to, workshops covering necessary concepts, tools, and techniques; numerous workdays; and testing, demonstration, and review meetings.

- Students are evaluated on their performance on elements related to the project.

The benefits of a project-based course are directly related to the choice of the project. With the guidance of our department's advisory council, which consists of a number of industry professionals, four knowledge and skill areas have been identified as being important foundations for a successful career in information technology. These areas have been placed in our mission statement and provide the criteria for identifying projects used in project-based courses. For a project to be considered, it must provide the opportunity for students to develop and apply the following:

- Computer knowledge - a demonstrated understanding of information science and technology concepts and processes, their relationships to each other, and their relationships to existing and emerging computing technologies,

- Hands-on skills - the confidence and skills to independently learn and apply existing and emerging computing technologies and processes,

- Independent learning - the confidence and skills to solve an unknown problem and to efficiently research, learn, and apply an unknown topic or skill to novel problem-solving situations, and

- Communication skills - the confidence and skills to effectively communicate (read, listen, write, and speak).

Experiences at Doane with Project-Based Courses

Four courses in the Information Science and Technology curriculum at Doane College have been modified to be project-based rather than project-supplemented. These courses include:

1. Software Engineering I - Offered in the fall of 2001, the project involved the debugging and modification of a large, open-source C program.

2. Database Application Design and Development - Offered in the spring of 2002, the project involved the analysis, design and development of a recruit management system for the women's basketball coach.

3. Web Application Design and Development - Offered in the fall of 2002, the project involved the analysis, design, and development of a web application to share and obtain variations of poker games for a user wanting such an application.

4. Networking II - Currently being offered (spring of 2003), the project involves the development and administration of a Linux network.

The project-based courses utilized a problem-solving framework employed in all IST courses with projects, whether project-based or project-supplemented. The framework uses a five-phase project methodology. The output of each phase is one or more deliverables that are used as input into the next phase. The phases (and typical deliverables) include: problem recognition (project description, project plan), analysis (project requirements and specifications), solution design (database design, data dictionary, website map, algorithms), implementation (the actual application or network), and support (installation plan, training, backup).

Project binders are used as a project management tool. The binders provide a central location for the deliverables of a project team. In addition, formal project reviews are held with the project team and the project leader (the course instructor).

Software Engineering I

Offered in the fall of 2001, the project involved the debugging and modification of a large, open-source game written in C. The game chosen for the class was the SlashEM derivative of the popular and long-lived Rouge. This course was the first of our project-based courses to be offered; in many ways, it was a prototype for the project-based courses discussed below. Details regarding this course were presented in greater depth at the MICS 2002 conference [Mey02].

The objectives of the project were:

- Provide students with experience with software systems of "real world" size.

- Give students first-hand evidence of the benefits of the software engineering topics covered in class.

- Provide students with software maintenance experience.

Highlighted activities and deliverables in each phase of the project framework included the following:

- Problem recognition – Students first had to understand the SlashEM game, and the source code from which it could be built. This modeled the first steps maintenance programmers would need to take after being assigned to a software system. Each team had to parse the (often slightly inaccurate) documentation for SlashEM in order to successfully build the game for the first time; the first deliverable for this phase was a working executable. Once the system was up and running, each team had to produce a structure chart of the system, in order to begin to understand where the functionality of the game resided.

- Analysis – The teams working on this project had to add their own modifications and improvements to the SlashEM game. The teams were in direct competition with each other for this part of the project, so the "cooler" new features would be worth more. Students had to explore possible new features and determine their feasibility. Deliverables for this phase was a discussion of the analysis activities conducted, presented by each team during their final project presentation.

- Solution design – Once the teams had settled on their desired new features, they had to perform design tasks in order to determine how to implement them. Deliverables for this phase was a discussion of the design activities conducted, presented by each team during their final project presentation.

- Implementation – Once the new feature set was well understood and planned out, the teams had to actually implement the changes. Deliverables for this phase were the modified source code files of the system, properly annotated with comments to indicate where changes were made, and why they were made.

- Support – All of the activities and deliverables for this project are examples of how a "real world" software system might be supported and maintained.

In this class, the open-source software project was not the only component that students were evaluated on; there were also traditional exams and other assignments. The project, however, was worth fifty percent of each student's final grade. Each of the major project milestones of the open-source project, such as building SlashEM, producing documentation for the game, implementing added features, etc., made up a portion of the project grade for the course.

The course syllabus can be found at *http://ist.doane.edu/meysenburg/ist312.htm.*

Database Application Design and Development

Offered in the spring of 2002, the project involved the analysis, design and development of a recruit management system for the women's basketball coach. Ten teams of two students completed the project.

Through the completion of the project, the course had the following objectives:

- Provide the ability to effectively design a database and utilize a database management system to develop a database application.

- Gain experience in related topics that are of current interest including developing web-based applications using Active Server Pages (ASP) and VB Scripting.

- Develop the skills necessary to be an effective member of a development team including planning skills, problem-solving skills, and communication skills.

Highlighted activities and deliverables in each phase of the project framework included the following:

- Problem recognition – Each team created a document describing the project. In addition, each team created and updated throughout the project a plan that listed the tasks that needed to be completed in order to complete the project.

- Analysis - A class training session was held discussing analysis techniques. Interview questions were formed. The user was interviewed by different teams at different times. The interviews were videotaped and replayed to the entire class. Each team formed a document that outlined project requirements.

- Solution design – For this project, short training sessions were given in database design (normalization), database implementation (tables, relationships, forms, reports, and queries), VBA (Visual Basic for Applications), and ASP (Active Server Pages used for a simple web interface). Each team was required to document the design of their database including the normalized tables (and their fields and primary key), relationships, forms, and reports.

- Implementation – Fifteen days of class time were devoted to work days. This provided the teams the opportunity to ask questions of the project leader and share with other project teams. Internal documentation of VBA code and ASP pages was stressed.

- Support – An implementation and support plan was created by each team that included tasks for installing the application, training the user, and backing up the database.

Students were evaluated in three different manners.

1. As part of the training sessions, individual students were required to complete short exercises that demonstrated skill in database design and implementation, VBA, and ASP. Each student had to successfully complete all exercises in order to pass the course although the exercises were not used to actually calculate a course grade.

2. Sixty percent of the grade was based on the project deliverables. Teams turned project binders in three times during the semester. In addition, teams met individually with the project leader twice during the semester. Feedback was provided that allowed teams to develop effective deliverables placed in the project binders. The final application was demonstrated to the user.

3. Forty percent of the grade was based on an individual performance evaluation. An evaluation form was developed using as a guide actual evaluation forms given by a member of our advisory council and used by her company. Both midterm and final evaluations were performed. Evaluation criteria included such items as teamwork, communication, and participation.

The course syllabus can be found at *http://ist.doane.edu/courses/ist307s.htm.*

Web Application Design and Development

Offered in the fall of 2002, the project involved the analysis, design, and development of a web application to share and obtain variations of poker games for a user wanting such an application. This course had as a prerequisite the *Database Application Design and Development* course.

Through the completion of the project, the course had the following objectives:

- Provide the skills to create a web application using various techniques and tools.

- Develop the skills necessary to be an effective member of a development team including planning skills, problem-solving skills, and communication skills.

Highlighted activities and deliverables in each phase of the project framework included the following:

- Problem recognition – Each team created a document describing the project. In addition, each team created and updated throughout the project a plan that listed the tasks that needed to be completed in order to complete the project.

- Analysis - A class training session was held discussing analysis techniques. Interview questions were formed. The user came to class and was interviewed by

the class at different times throughout the semester. Each team formed a document that outlined project requirements.

- Solution design – For this project, students had to demonstrate basic skills in database design and development and web page development. Training sessions were not given in these areas. Students also had to demonstrate basic skills in creating ASP pages that would read data from a database, add a record to a database, update a record in a database, delete a record from a database, and provide access to a database to only authorized users. Training sessions were provided for these skills. Each team was required to document the design of their database, the layout of their website, and the logic of any active server pages.

- Implementation – Twenty days of class time were devoted to work days. This provided the teams the opportunity to ask questions of the project leader and share with other project teams. Internal documentation of ASP pages was stressed. In addition, teams created testing documents that were used by another team to test their application.

- Support – An implementation and support plan was created by each team that included tasks for installing the application, training the user, and backing up the database and website.

Students were evaluated in two different manners.

1. Individual students were required to complete short exercises that demonstrated skill in database design and implementation, web page development, and ASP pages to manipulate a database (create, read, update, and delete records along with securing pages). Each student had to successfully complete all exercises in order to receive forty points towards the final grade. These forty points were all or none. Students had to complete all exercises in order to receive the forty points or else they would receive zero points. The course grade was based on one hundred points. Failure to obtain the forty points meant failing the course.

2. The other sixty points towards the final grade was based on a Project Review Rubric. The rubric was designed with 10 criteria areas. If students met the expectations in each area, they would receive a B for the course. To receive an A, teams had to exceed expectations. The rubric was based on an evaluation form developed using as a guide actual evaluation forms given by a member of our advisory council and used by her company. Both midterm and final evaluations were performed. Evaluation criteria included such items as teamwork, communication, and participation.

The course syllabus can be found at *http://ist.doane.edu/courses/ist407s.htm.*

Computer Networking II

Currently being offered (spring of 2003), the project involves the development and administration of a Linux network. As there are many aspects of networking, ranging from the administration of a Linux LAN to using Linux servers to provide Internet services, this course is structured as a series of more-or-less self-contained "mini-projects."

The objectives for this course are to provide students, who have gained an understanding of networking fundamentals through the prerequisite *Computer Networking I* course, practical experience with the administration of a modern computer network. Topics such as basic connectivity, file sharing, centralized authentication, Internet services (mail, web, FTP, etc.), security, and user and group management.

Students are divided into five teams of three or four students each. Each team has their own miniature Linux network, with three computers and a hub or switch. For each of the "mini-projects," one student in each team is the designated project leader, responsible for the overall plan of implementation and the documentation of the project.

Each team maintains a project binder, with the following sections: a hardware catalog, listing the characteristics of the particular hardware components of the team's network; a Linux reference section, containing handwritten notes about Linux commands, files, and procedures that team members learn during the project; and a project section, which contains the documentation for each "mini-project" conducted by the students.

Highlighted activities and deliverables in each phase of the project framework included the following:

- Problem recognition – For each "mini-project," students are given only a basic introduction to the task at hand. For instance, for the Network Information Service (NIS) project, a brief background of NIS was provided, along with one of its principle uses: centralized, remote authentication for user login on Linux LANs. The teams were told that they needed to configure their network with a single NIS server to provide authentication services; the other machines in the network were to use the NIS server to authenticate user logins. After the introduction for each project, the students were required to further research the problem at hand using the course text, Linux man pages, and other resources of their choice.

- Analysis and solution design – for each "mini-project," the designated team leader was responsible for establishing an overall plan of action. The plan would include step-by-step actions to perform, and responsibilities for each team member. Also included would be the architecture of the network to be implemented, and the role of each machine in the network. This plan constitutes the analysis and design phase for each project.

- Implementation – Each team was responsible for the necessary configuration of their network for each project. Once the network was correctly configured, teams were required to demonstrate the basic functionality required by the project.

- Support – The teams kept documentation of everything they did in order to accomplish each project: what worked, in what order, and what didn't work. The documentation for each project was intended to be clear enough to allow other individuals to accomplish the same task, with a minimum of difficulty. In this way, the project documentation would simulate the kind of notes and procedures that real-world system administrators would keep in order to support the efficient operation of their networks.

All of the documentation for each project was placed in the team binders. The binders were collected and evaluated every two or three weeks.

Students were evaluated based on the following criteria:

1. At the end of each "mini-project," a short (15 minute) quiz was given over the project. The quiz covered the basics of what the students should have learned during the proceeding project. For example, during the TCP/IP project, the teams were required to configure their network to use DHCP for IP address allocation. The TCP/IP quiz asked students questions such as "What startup script should be used to start the DHCP client or server daemons when the computer is powered up?" These quizzes made up thirty-five percent of the student's course grade.

2. Each student was evaluated on the project documentation they produced when they were the project leader. The documentation was evaluated on a 0-2-4-6 point scale. A rubric showing examples and characteristics for each point level was included on the course syllabus. Individual documentation scores made up thirty percent of the students' course grade.

3. Each student was evaluated on the project documentation produced by everyone on their team, using the same evaluation criteria as above. Team documentation scores made up twenty percent of the students' course grade.

4. Attendance and a subjective evaluation of participation and effort made up the final fifteen percent of the students' course grade.

The syllabus for this course can be found at *http://ist.doane.edu/meysenburg/ist422.htm*.

What We Learned

Overall, the project-based courses met our expectations and provided the benefits we were anticipating. Students did have an experience with a project more in line with what might

be expected in the industry. While we have yet to gather formal assessment data, anecdotally we feel they did gain a better understanding of the project framework. In addition, they did develop confidence, skills, and knowledge in our four target areas: computer knowledge, hands-on skills, independent learning, and communication skills.

Based on our experiences, here are some suggestions if you are considering offering a project-based course:

- Have all students work on the same project. Focusing on one project makes the instructor a much more effective project leader. In addition, teams share what they learn and often engage in friendly competition leading to better learning experiences.

- Use homogeneous project teams. Placing people with similar skills on the same team may result in differences between teams, but in terms of individual learning will be better for all students.

- Provide plenty of work time in class. This time provides the perfect opportunity to share discoveries by other teams. It also provides time for the project leader to work with individual teams or to address all teams.

- Provide a structured project framework that includes a detailed task list with deadlines. Students need to know what is due and when. They also must think through the tasks they need to complete in order to meet the due date.

- Meet with teams individually to discuss the progress on the project and their deliverables. This is an opportunity to provide invaluable feedback. Meet at least twice during the project.

- Incorporate training sessions into the semester. This provides the opportunity for students to learn the basics of the tools they are to use to complete the project. However, don't teach them everything they will need to know. They need to develop the confidence and skill to learn what they need to know and then apply it.

- Related to the previous point, be sure to provide students with appropriate resources such as reference manuals and websites that they can use when they need to learn.

Summary

Not every course lends itself to be project-based. However, for those that do, using a project-based approach can provide benefits not found in courses without projects or that are project-supplemented. However, in order to realize these benefits, great care must go into the selection of the project and the logistics of its implementation.

References

[Mey02] Meysenburg, M. M. (2002) Using open source software for software engineering class projects. *Proceedings of the 35th Annual Midwest Instruction and Computing Symposium*, April 5-6, 2002, University of Northern Iowa, Cedar Falls, Iowa. http://www.micsymposium.org.