Focusing on Design

John E. Howland, Aaron Konstam and Gerald Pitts
Department of Computer Science
Trinity University
715 Stadium Drive
San Antonio, Texas 78212-7200
Voice: (210) 736-7480
Fax: (210) 736-7477
E-mail: jhowland@Trinity.Edu
E-mail: akonstam@Trinity.Edu
E-mail: gpitts@Trinity.Edu

November 26, 2002

Abstract

The role in which design should play in the undergraduate curriculum is addressed. An approach to introducing design into the curriculum through a sequence of courses where students solve design problems of increasing complexity is given. [1]

Subject Areas: Computer Science Education, Computer Science Curriculum.
Keywords: computer science, design.

1 Introduction

For many years computer science majors at Trinity University were required to complete a two semester senior level capstone course which involved analysis, design and prototype implementation of a problem of significant complexity in the fall semester. During the spring semester, computer science majors extend the prototype implementation of their solution to a complete solution.

[1]This paper has been published in the Journal of Computing in Small Colleges, Volume 14, Number 3, Pages 160 - 165, March 1999. This paper was presented at the CCSC South-Central Conference, April 17, 1999, St. Edwards University, Austin Texas.

One of the problems students encountered with this course involved insufficient skills and experience in analysis and design required for successful completion of complex problems. The typical kind of programming assignments and laboratory problems encountered in lower division computer science courses usually did not prepare students with adequate analysis and design skills to handle larger more complex problems.

To address this problem, the computer science department recently made significant changes to its curriculum for computer science majors with the introduction of three one-credit required design courses which are taken in the spring semester of the second, third and fourth years.

Students are introduced to formal analysis, design and specification methods [Ken 96, Pet 87, DeM 78, Bro 75], and are given an opportunity to apply these concepts twice (solving the second and third year design problems) before beginning their senior project problem. Third year students have already taken the second year design course, so there is some repetition of introductory analysis and design methodology which most students find helpful as their understanding of these tools and techniques matures. Also, third and fourth year design students are taught some object-oriented design [Col 94, Boo 94] techniques.

In addition to providing experience in analysis and design, the courses also have units which deal with professional conduct [ACM 98] and ethics.

2 The Design Course Sequence

The three design courses (second year, third year and fourth year) are scheduled to meet at the same time so that certain course activities may be scheduled to involve all three courses in a common meeting room. Each course has an instructor who oversees the individual activities of each class, but the instructors work together as a team to handle common course activities. When separate activities are scheduled, each course meets with its instructor in its regularly assigned classroom. Examples of common activities for all three courses include:

- guest lectures from industry professionals
- design presentations
- ethics presentations

Table 1 gives the course meeting schedule for the spring semester, 1998.

Scheduling all three courses to meet at the same time allows guest lecturers from industry the opportunity to speak to all of the computer science majors. Lecture topics have included professionalism, job opportunities and work environments in various sectors of the computing industry.

Another benefit of this course scheduling decision is that all of the majors have an opportunity to make their project presentations to an audience which

Class Period Emphasis: P - Professional, E - Ethical, D - Design

Class	Month	Day	Topic
D	January	14	1st 15 min, Common Introduction to Design I Introduction of 2nd and 3rd Year Design Problem (Sophomores, Juniors)
D		21	Common - Senior Project Presentations
DE		28	1st Hour Common Introduction to Design II Return to individual Classrooms Introduction of Design Problems
DE	February	4	Ethics Presentation
E		11	Common - Junior Group Presentation
DP		18	1st Hour Common Return to individual Classrooms
DE		25	
E	March	4	Common - Sophomore Group Presentation
		11	Spring Break - No Classes!
DP		18	1st Hour Common - Speaker Return to individual Classrooms
DE		25	
DP	April	1	
D		8	Common 1st Hour Junior Design Presentations 2nd Hour, Sophomore Design Presentations
DP		15	1st Hour Common - Speaker Return to individual Classrooms
DE		22	 2nd Hour, Student Course Evaluations

Table 1: Class Meeting Schedule

includes second year students, third year students and fourth year students. During the second class period, the senior students present their newly completed major project analyses, designs and prototypes. The sophomore and junior students benefit from this experience by seeing presentations from more experienced students and the second and third year students have an opportunity, later in the semester, to present their analysis, design and implementation of the second and third year design problems and have their presentations critiqued not only by the instructor but their upper class peers as well.

We have also found that grouping second, third and fourth year students together for certain common experiences helps create a sense of group (computer science majors) identity and tends to foster a friendly and beneficial competition among members of this group of students.

The sophomore and junior course sections are partitioned into project groups while the senior students may work in project teams or individually on their senior projects; the choice is up to the senior student.

Some class periods involve ethics discussions and presentations. Second and third year project teams plan presentations of ethical issues [Erm 97] in the computing field. Project teams have successfully used skits, role playing and even production of an ethics video. Presentation topics have ranged from software piracy issues, to trade secrets, honesty and validity of project schedules.

2.1 Second Year Design Problem

The second year design problem used in the spring semester, 1998, involved the design and implementation of the Unix `sed` command. This problem was chosen because it was of moderate complexity but suitable for applying the standard analysis and design techniques. The advantage we saw in picking this as the design target was that `sed` was a program whose functionality and interface were well documented and understood. It was easy for students to compare the behavior of their implementation of `sed` with the behavior of the Unix version. It should be emphasized that the students were directed not just to implement their `sed` *look-alike* but to approach the implementation task as a full scale design project. They were not allowed to code any module on which they had not first completed a formal design.

2.2 Third Year Design Problem

The third year design problem used in the spring semester, 1998, involved the design and implementation of an emulator for the MIPS R3000 processor. The emulator was to execute the same bit patterns in memory that a real R3000 processor would execute. The emulator was to be an instruction set emulator only. This simplified the problem in that students did not need to design an emulation of system buses and external processor interfaces. None of these students had much (some had none) experience with assembly language or machine language programming, so this project is designed, in part, to provide an opportunity for students to learn more about the organization and operation of real computer systems.

2.3 Fourth Year Design Problem

The fourth year design problem is selected by seniors in the fall semester when they begin their capstone course. When they enter the fourth year design course, the seniors have just completed the analysis, design and prototype implementation of their senior project in the fall semester and are expected to provide leadership by example to the second and third year students in the common portions of these three courses. Their formal presentations during the second week of the course introduce the analysis and design phase in a polished professional manner and provide examples for the younger, less mature students to follow.

The nature of the design problems picked by the seniors has varied from virtual environment interfaces to Internet database mining query systems. Each fourth year design project must be approved by the computer science department for feasibility and level of effort before it is initiated. Many software systems have been developed for industry with little complaint and much praise, while some, of course, remain purely academic. In fact some systems are still in use after more than 10 years. The software implementation, testing and final review is required to be completed before a senior can graduate and is, therefore, the primary focus of students in the fourth year design course. It has been our experience that the fourth year students regularly attend the course and provide examples of leadership and insight to the younger students in the second and third year design courses.

3 Student Course Evaluations

The first offering of these courses occurred in the spring semester 1998. The instructors were interested in measuring student opinions about the courses in an effort to improve the courses in subsequent offerings. A special assessment form was developed and administered to the students, in addition to the standard university course evaluation form. Sixty students were enrolled in the three courses and 95% of those participating in the course evaluations felt that overall, the course was very beneficial. Some of the positive aspects of the course included:

- guest speakers
- sense of community (computer science)
- information on jobs
- group work skills
- learning more about systems design
- presentations
- learning about planning
- ethics

Some of the negative aspects of the course included:

- too much work for a 1 credit course
- courses seemed unorganized at times
- more interesting projects for sophomores and juniors

We intend to address these issues in an effort to improve the courses.

4 Conclusions

A problem arose in the design of the sophomore projects that surprised us and will have to be addressed in the future. Although, the students were given specific documentation on the operation of `sed`, the material turned to be neither specific or directed enough to prevent the students from universally making the same mistake in their design. Every group produced programs that closely matched the functionality of `sed`. But nearly every group did not see the necessity of matching the interface of the Unix `sed`. We should have caught this but the problem was that the design process we had them go through did not adequately include a design of the interface. The solution is, obviously, to put more emphasis on the interface in the requirements of the student's design efforts. We will certainly do that in the future.

The third year design class was told to not consider the problem of designing a user interface for the simulator. A simple mechanism of reading startup values for registers and memory from standard input before a simulation run and writing register contents as well as memory to standard output at the end of a simulation run was suggested. This approach allows suites of test files to be developed and their result files to be compared with known correct results. However, each project team elected to ignore this suggestion and each spent excessive effort (at the expense of simulator completeness) to design (sometimes elaborate) interactive simulator user interfaces. In future offerings, the instructors intend to enforce project specifications to avoid this kind of problem.

During the initial offering of these three courses, the instructors are enthusiastic about the benefits of sharing common activities between the courses and plan to enhance this aspect of the courses. Much of the success of the second and third year courses depend on the choice of an appropriate design problem. We plan to develop a suite of different design problems so that the problems are not repeated frequently. A variety of different Unix utility programs could be used just as effectively as `sed` in the second year course. Other machine architectures could be simulated in the third year design course.

References

[ACM 98] ACM, Code of Professional Conduct, from Bylaw 19, Bylaws of the ACM, 1998.

[Boo 94] Booch, Grady, *Object-Oriented Analysis and Design with Applications*, The Benjamin/Cummings Publishing Company, Inc., 1994.

[Bro 75] Brooks, Frederick P., *The Mythical Man-Month*, Addison-Wesley Publishing Company, Inc., 1975.

[Col 94] Coleman, Derek, Arnold, Patrick, Bodoff, Stephanie, Dollin, Chris, Gilchrist, Helena, Hayes, Fiona and Jeremanaes, Paul, *Object-Oriented Development, The Fusion Method*, Prentice Hall, 1994.

[DeM 78] De Marco, Tom, *Structured Analysis and System Specification*, Yourdan, 1978.

[Erm 97] Ermann, Willams and Shauf, *Computers, Ethics and Society*, 2nd Edition, Oxford University Press, 1997.

[Ken 96] Kendall, Penny A., *Introduction to Systems Analysis & Design: A Structured Approach*, Irwin, 1996.

[Pet 87] Peters, Lawrence, *Advanced Structured Analysis and Design*, Prentice Hall, 1987.