Designing a Prerequisite Course for a Computer Information Systems Program in a Computer Science Curriculum

Shaun-inn Wu[1]
Computer Science Department
California State University San Marcos
San Marcos, CA 92069

Abstract

We are designing a new Computer Information Systems program in our department. Being housed in the Computer Science Department, this program will emphasize more solid technical knowledge of software and hardware than traditional Information Systems programs housed in business schools. In order to strengthen the programming background of our students, a prerequisite course covering skills in word processing, spreadsheet, and database systems as well as programming techniques is very desirable. We are converting an existing general education course for non-Computer Science majors to such a prerequisite course for our new Computer Information Systems program.

Keywords: Computer Information Systems, CIS, prerequisite course, Java programming

1. INTRODUCTION

Our university is a relatively new institution in North San Diego County. Our Computer Science curriculum was designed according to the newest recommendations at the time: Computing Curricula 1991 (ACM/IEEE-CS, 1991). Currently, we are in an early stage of modifying our curriculum according to the new Computing Curricula 2001 (ACM/IEEE-CS, 2001). We are also in the process of proposing a new Computer Information Systems program based on the last two curriculum recommendations in Information Systems: IS'97 (ACM/AIS/AITP, 1997) and IS 2002 (ACM/AIS/AITP, 2002).

As we design a curriculum in Computer Information Systems to be proposed in the near future, we have consulted curriculum recommendations such as IS 2002 as well as many existing curricula in other institutions. There exists a great deal of variation as how educational goals are achieved through different curriculum designs. Because we have the advantage of not having to modify an old, existing curriculum in order to fit it into new curriculum recommendations, we can try to implement the new recommendations directly into recommended courses. However, since we are designing this new curriculum in the Department of Computer Science, we naturally would like to utilize as many Computer Science courses as possible. Hence, we will have to decide how to merge the recommendations into the Computer Science curriculum and propose a minimum set of new courses for Information Systems.

As stated in IS 2002 Model Curriculum recommendations, the prerequisite computer skills provide a personal capability for student use of information technology. Several applications are listed as useful to students and graduates, including word processing, Internet browsing, electronic mail, spreadsheet processing, database management, presentation graphics, and external data-

[1] shauninn@csusm.edu

base retrieval. As students are getting more and more computer literate, skills in word processing, Internet browsing and electronic mail are usually acquired prior to formal courses. Some institutions provide a required course covering the prerequisite IS skills level. Other institutions provide students laboratories with computer-based tutorial modules to acquire this competency. In both cases, competency tests may be used to ensure adequacy of prior knowledge.

At California State University San Marcos, we currently provide both a competency course and tutorial workshops to help all students to acquire certain skills in using computers. We also require students to pass a competency test to make sure students have adequate computer competency. In terms of the prerequisite computer skills, we not only want to cover the important applications but also start building programming skills. A current general education course can be modified to fulfill such needs.

2. DESIGN OF A CURRENT COURSE

Our current course was designed when we initially created our Computer Science curriculum. The goal of the course was to fulfill both the computer competency requirement of our institution and the upper-division general education requirement in science and mathematics. To meet the computer competency requirement, the class currently covers spreadsheet and database management systems quite extensively as well as a little word processing on PC. We also cover basic computer hardware and software, computer networking, history of computing, and computer related societal and ethics issues. To meet the requirement in upper-division general education, we cover problem solving on computers and programming. In this regard, we chose to use Pascal because the first of the two principal design aims is "to make available a language suitable to teach programming" (Jensen and Wirth, 1978; Jensen and Wirth, 1985). In addition to lectures, there are laboratory sessions for each application and to learn programming. In addition, there is a term project in which students are required to find a problem to solve, design a solution, and implement it.

The original design of the course followed the success of a similar course at University of California at Berkeley (Kiser, 1989; Patterson, et al., 1989). When we started to teach this course in a Macintosh laboratory in 1991, we used MacWrite, Excel, FileMaker Pro and Think Pascal. Thereafter, we taught a PC version of the course using WordPerfect, dBase II, Lotus 1-2-3 and Turbo Pascal. It became a very popular course on our campus. A few years later, we decided to only offer PC sections due to its huge popularity and the logistics problems of maintaining two different computing platforms for the same course. When the original software became obsolete, we began using Microsoft Word, Excel, and Access while maintaining Turbo Pascal for the algorithmic thinking and problem solving part of the course. Over the years, we adopted several generations of the lab manuals for the applications and programming labs. Currently, we are using Exploring Microsoft Office XP (Grauer and Barber, 2002) for word processing, spreadsheets, and database systems. There was also the necessity to create a new laboratory manual for Pascal programming as a result of moving from a DOS system environment to Windows and the availability of a newer version of Turbo Pascal.

Since this course is designed primarily for students of other majors, our Computer Science majors are not allowed to take this course to fulfill their academic requirement.

3. PROPOSED NEW COURSE

As stated earlier, our students are getting more and more computer literate, and skills in word processing, Internet browsing and electronic mail are usually acquired prior to formal courses. Hence, we probably will only provide some tutorial materials to a few students who need to strengthen these skills without having to devote class time to them. Instead, we will only cover applications in spreadsheets, presentation graphics and database management systems. Because our Computer Information Systems program will be offered in Department of Computer Science, we would like to emphasize our program in software fluency. Hence, it helps to expose our students to problem solving skills and software solution implementation and some software engineering.

In order to integrate it with e-commerce in new Internet environment, we decide to experiment with a Java programming environment in this new course. A new set of laboratory sessions was designed in last year and we just finished test-teaching these new materials in a few summer courses. Students seemed to become far more interested in programming after this change because Java has state-of-art tools for working with Internet. The interest is also due to the way these laboratory sessions were designed.

In the design of our programming lab sessions, we took the approach of applying case studies that have been used successfully in many other disciplines. Instead of teaching bits and pieces of a programming language in the way Computer Science courses traditionally do, we present each lab as a case study (Wu, 1997). We design a solution for each case and then show students how to implement the solution. Although the solution is given to the students, they still learn the important concepts of software design and implementation because the solution is presented step by step. They also pick up some useful tools in the program modules that could be used in their own problem solving. We also try to enable students' capabilities to program audio and video early. Students get to see the results of their programs right away in the early stage of learning problem solving on computers. Hence, students get more involved right from the beginning of their programming experience.

The laboratory sessions in problem solving and Pascal programming are currently being updated into a multi-media book on CD for the new Microsoft Windows environment (Sebrechts, 2003). In addition, we have also redesigned the laboratory sessions for the new course using Java programming (Wu, et al., 2003). Java was chosen because it is ideal for programming with the Internet, which is increasingly important in business world. After appropriate lectures covering the basic structure of Java programs and usages of basic graphics library, our very first lab session was to design a graphics object and display it on the screen. The goal of the current lab materials is to display a (hot-air) balloon with white clouds in the blue sky above the ground in green color (presumably covered by grass). At the beginning of the lab materials, we explain how to design a solution that simply displays a balloon and show them how such a program can be constructed. Then, they are asked to add in a new method to draw the clouds and make a call to this method. Subsequently, they are asked to add another method to draw the ground. The programming tools they learned through this program will eventually help them to draw other graphics objects in their term project. The development of this program is continued in another lab session. Students learn to move the balloon with basic animation techniques after they learn more program structures such as loops.

Other lab sessions are designed to help students learn how to make sounds on computers (emulate different musical instruments) and eventually play some music with the computer. Some other labs teach students how to develop solutions to construct conversations between computers and users; students create a program that emulates a computer psychiatrist responding to users (patients) much in the same way as the Elisa program written by Joseph Weizenbaum at MIT many years ago does. We also developed lab sessions that help students create a simulation of a small database of simple records, an address book.

The course is delivered in two 75-minute sessions each week. The session is either a lecture or a lab. The following table shows the appropriate timetable for different sessions in a semester:

Class	Subject
lecture	Introduction
video lecture	Giant Brain
lab	Microsoft PowerPoint Session 1
lecture	Computer Hardware
lab	Microsoft Excel Session 1
lecture	Computer Software
lab	Microsoft Excel Session 2
lecture	Algorithmic Thinking, Problem Solving
lab	Microsoft Excel Session 3
lecture	Transition Into Java
lab	Your First Java Program
lecture	Conditions And Choices in Java
lab	Playing A Note
lecture	Repeating Similar Actions in Java

lab	Keyboard Music
exam	Midterm Exam
video lecture	Inventing The Future
lecture	Tracing And Debugging Your Programs in Java
lab	Flying Balloon
lecture	Handling Data Of The Same Type in Java
lab	Alisa
video lecture	The Paperback Computers
lab	Microsoft Excel Session 4
video lecture	The Thinking Machine
lab	Microsoft Access Session 1
video lecture	The World At Your Fingertips
lab	Microsoft Access Session 2
lecture	Computer Networks
lab	Build your own web page and use email
lecture	Computer Ethics And Societal Issues
lab	Microsoft Access Session 3
exam	Final Exam

4. CONCLUSIONS

In designing the new Computer Information Systems program in our Computer Science Department, we would like to reuse as many existing Computer Science and Business courses as possible. While there are some differences between the Computer Science and Computer Information Systems programs, there are many common courses such as basic programming courses, database management systems, and Management Information Systems (IMPAC, 2002). However, since we are focusing the design of the prerequisite course to the Computer Information Systems program in this paper, we will discuss the specifics of the overlapping between the two programs and how they share certain courses elsewhere.

One of the major changes from IS'97 to IS 2002 is the merge of two courses, IS'97.P0 - Knowledge Work Software Tool and IS'97.P2 - Personal Productivity with IS Technology, into one course, IS 2002.P0 - Personal Productivity with IS Technology. More specifically, the learning units from 1 to 4 and from 13.1 to 13.16 and 15.17 originally covered in two courses now are covered in the same course. It greatly reduces the room to cover functions of certain applications in the P0 course if necessary. Since the prerequisite of the IS 2002.P0 course is elementary knowledge of word processing, spreadsheets, e-mail, and Web browsing, it is our belief that we need a prerequisite course to this course to ensure the required skills in using computers. Because the life cycle of development of information systems is covered in the new IS 2002.P0 course, it is essential that students be exposed to some software implementation experience. It is our opinion that such a prerequisite course prior to the IS 2002.P0 course should cover both the more sophisticated applications as well as provide some problem solving and programming experience. The design of our new course can accomplish such goals with a few modifications from our general education course such as adding coverage for the presentation software and updating the problem solving with software development and implementation using Java.

5. References

ACM/AIS/AITP, (1997), IS'97: Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems, Association for Computing Machinery, Association for Information Systems, and Association for Information Technology Professionals.

ACM/AIS/AITP, (2002), IS 2002 - Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems, Association for Computing Machinery, Association for Information Systems, and Association for Information Technology Professionals.

ACM/IEEE-CS, Computing Curricula 1991, (1990), Association for Computing Machinery and Institute of Electrical and Electronics Engineers – Computer Science Society, December 17, 1990.

ACM/IEEE-CS, Computing Curricula 2001, (2001), Association for Computing Machinery and Institute of Electrical and

Electronics Engineers – Computer Science Society, December 15, 2001.

Grauer, Robert T. and Maryann Barber, (2002), Exploring Microsoft Office XP, Prentice Hall, Upper Saddle River, New Jersey.

IMPAC, (2002), IMPAC Annual Report 2001-2002, Intersegmental Major Preparation Articulated Curriculum Project, State of California.

Jensen, Kathleen and Niklaus Wirth, (1978), Pascal User Manual and Report, 2nd ed., Springer-Verlag, New York.

Jensen, Kathleen and Niklaus Wirth, (1985), Pascal User Manual and Report – ISO Pascal Standard, 3rd ed., Springer-Verlag, New York.

Kiser, Denise S., (1989), Computing Unbound Hands-on Exercises for the IBM PC, W. W. Norton & Company, New York.

Patterson, David, Denise S. Kiser, and D. Neel Smith, (1989), Computing Unbound Hands-on Exercises for the Macintosh, W. W. Norton & Company, New York.

Sebrechts, Patrick, (2003), Programming with Turbo Pascal, Pearson Custom Publishing.

Wu, Shaun-inn, (1997), Learning Pascal Programming by Case Studies, Forbes Inc.

Wu, Shaun-inn, Serif Avcibasioglu, and John T. Wu, (2003), Learning Java Programming by Case Studies, Laboratory Manuals, California State University San Marcos.