Introducing an 'Advanced Application Development' Course into the IS Curriculum -- Lessons from the Field

Ido Millet
Penn State Erie, The Behrend College

Follow this and additional works at: http://aisel.aisnet.org/amcis1995

Recommended Citation
http://aisel.aisnet.org/amcis1995/109
Introducing an 'Advanced Application Development' Course into the IS Curriculum -- Lessons from the Field

Ido Millet
Penn State Erie, The Behrend College

Introduction

IS education programs must strike a delicate balance between practice and theory. Typically, courses that require the use of a computer are considered as "practice" courses, others are considered "theory" courses. However, we can see a clear trend in IS education whereby more and more "theory" is taught in the context of "practice." Indeed, this trend is the correct reaction to the realities of the IS domain and the market place.

Another challenge is the need to keep up with the rapid evolution of tools and techniques. Here again, balance is the key word. Trying to stay right at the leading edge of information technology (IT) would place too much strain on faculty and on financial resources, while staying too far behind it would diminish the educational value of the program.

Adjustments to IS curriculum are sometimes like earthquakes; a gap between curriculum and IT practice develops until it exerts enough stress to cause a realignment. We are due for such a realignment and this paper focuses on how we can best achieve some of it.

Why Change?

Many IS education programs have not caught up yet with a significant change in the way information systems are being developed today. A recent survey shows that 55% of 445 CIOs already have a Client/Server computing strategy and an additional 20% plan to implement one in the next 18 months.

In order to prepare our students for these new realities, we must introduce them to theory, concepts, and skills in areas such as Client/Server applications, Object Oriented methodologies and techniques, Graphical User Interfaces (GUI), ClassHierarchies, EventDriven programming, 3tier computing, etc...

Some of these concepts can be introduced in courses such as 'Systems Analysis and Design' (which typically emphasize theory), and in courses such as 'C++' (which typically emphasize practice). Still, there is a clear need to exercise and integrate the theory and practice skills introduced in these courses.

This requires higherlevel courses, and indeed many IS programs offer courses such as "Information Systems Development," "Systems Design and Implementation," and "Information Systems Project." However, while such courses are very valuable for exercising and integrating the more stable components of the IS curriculum (project
management, analysis, design, and programming), they typically avoid using the more advanced tools and techniques mentioned above.

By adding an 'Advanced Application Development' course to the IS curriculum we can achieve several objectives with the least amount of resistance. Attempting to simply add advanced material into existing courses would typically sacrifice adequate coverage of existing material. It would also require frequent adjustments to core courses.

The main benefits of introducing an 'Advanced Application Development' course into the IS curriculum are the increase in the educational value and reputation of the program, and the improved placement statistics for the graduating students. As a side effect, such a course provides opportunities for updating faculty knowledge and skills. Over time, the tools and techniques introduced in such a course can also migrate to enrich lower-level courses.

The main challenges in introducing an 'Advanced Application Development' course into the IS curriculum are in getting permission and resources, preparing for the course, and designing it. The following sections discuss how we approached each of these challenges and what we learned in the process.

**Getting Permission**

Well before proposing any changes, I mentioned the general idea to other faculty and to the administration. I periodically circulated news clippings in relation to the rapid growth in Client/Server application development, and the success of PowerBuilder as the tool of choice in that market. This improved the chances of getting the actual proposal accepted.

In the formal proposal, I suggested introducing the course as an elective rather than as a required course. This approach worked very well since it reduced risk. The course could be easily discontinued after one semester if the response to it was negative.

In explaining the benefits of the proposed course I mentioned the following points: a) It will provide an opportunity for the students to integrate and apply what they learned in earlier courses, b) It will increase the marketability of our MIS graduates, c) It will enhance the reputation of the MIS program, and d) It may open opportunities for Continuing Education offerings.

**Getting Resources**

Given that at the time of the proposal the cost of Client/Server application development software was high, the question of budget for purchasing enough copies was an important one. PowerBuilder, my first choice, was selling at $3,595 per copy. The good news was that the vendor, Powersoft, wisely decided to offer the software to universities at a steep educational discount.
By joining Powersoft's "STAR" program we were able to purchase PowerBuilder for only $115 per copy. In the proposal to the administration I outlined the savings and requested 15 copies for installation at the computer lab. The total cost was $1,725 (instead of $53,925) and it was approved without difficulty.

The "STAR" program is still active and it also includes special discounts for training one faculty member. This brings us to the issue of faculty preparation.

Preparing to Teach the Course

In the Spring semester of 1995 we offered for the first time an 'Advanced Application Development' course. I started preparing to offer this course in 1993 after it became clear to me that application development was undergoing a revolution and that PowerBuilder would probably dominate as the tool of choice for Client/Server application development.

Getting prepared to teach such a course is perhaps the most challenging aspect of offering it. PowerBuilder, as well as any of its serious competitors (e.g., Borland's Delphi, and Gupta's SQL Windows) are very rich and complex tools. The difficulty of mastering these tools increases as you consider the need to deal with a separate database engine (PowerBuilder comes bundled with a singleuser version of WATCOM's DBMS), advanced SQL, the Windows environment (things that go GPF in the night), object orientation (e.g., eventdriven programming, inheritance, attributes, and methods), a full-strength programming environment, and GUI design issues. In addition, you can expect frequent and significant changes in the software due to the heated competition and the rapid evolution of the domain.

Regardless of these difficulties, most IS programs would have at least one faculty member who is motivated enough (or foolish enough) to get involved in teaching such a course.

There are several methods that I found very useful in preparing to teach this course. I started with the obvious ones of buying the software, going through the manuals, doing the tutorials, studying the sample application that comes with the software, and reading general books and articles on Client/Server application development as well as specific books and journals on PowerBuilder. I selected one of these books *Developing PowerBuilder 3 Applications, Best Seller Edition* by Bill Hatfield as the textbook for the course.

For universities that participate in the "STAR" program, Powersoft offers the four day introductory class in PowerBuilder to one full-time faculty member at a cost of $375 (a 75% discount). I took this class and found it very useful. After returning from this class, it is important to immediately follow through by reviewing the material and building on it with more practice. I neglected to do this and probably paid for it in longer reorientation later.
The PowerBuilder discussion list on the internet (comp.softsys.powerbuilder) is a wonderful source of information. By reading most of the messages in this active discussion list, you can gain a lot of practical knowledge that typically comes only from long periods of actual work with a software tool. This knowledge helped me develop a few practice applications. Later, it also helped me troubleshoot many of the problems my students encountered during work on their group projects.

Designing the Course

The main objectives I set for the course were 1) to develop competencies required for using modern application development tools and techniques, and 2) to build on and integrate knowledge gained in previous courses. The first objective was served mainly through lab sessions and assignments. The second objective was achieved through regular class sessions and a multiphase team project.

Almost half of the twice weekly class meetings were conducted at a computer lab. Since the class was enrolled to its maximum of 25 students, and since we had only 15 copies installed at the lab (one of which was used by me on a PC equipped with an overhead projection panel), students had to double up on PCs. I believe this worked very well. Having two students work together improved their ability to keep up with the rapid pace of the lab sessions.

In order to minimize orientation cost at the beginning of each session, I used the same sample application as background for most of the lab sessions. Several students mentioned that they liked this approach because it provided continuity and allowed them to concentrate on the new material being introduced.

Most of the lab sessions were motivated by problems in the way the sample application was running. Since the students developed the sample application as a tutorial assignment at the beginning of the semester, they were very interested to discover various "hidden" problems and to find ways for solving them. This meant that the introduction of new techniques was done in the natural context of putting them to good use. I believe this contributed significantly to the speed and quality of the learning process.

Another technique I used periodically was to ask some of the more advanced students to lead the class through a certain development procedure after it was explained once. Besides providing diversion to the audience, it allowed me to gage how well the material was absorbed and where students were experiencing problems.

After a few meetings, it became clear that some students were not able to keep up with the pace of the lab sessions due to their attempt to take detailed notes. I decided to distribute detailed descriptions of each lab session at the following class meeting (I couldn't prepare the notes in advance since frequently the sessions took unexpected turns). This solved the problem very well. It also addressed the problem of students missing a lab session.
According to the course syllabus, each student was expected to read the PowerBuilder discussion list at least once a week, select an interesting message, and forward it to me. This worked extremely well. It provided the students with examples of typical problems and solutions, and a sense of how PowerBuilder was impacting individuals and organizations. Besides improving students' knowledge it also improved their motivation. Students told me that being able to understand most of the messages toward the end of the semester (while initially being baffled by most of them), was a nice way to gage the progress they have made and to gain confidence. I was gratified to see a message on the discussion list from one of my students answering correctly a nontrivial question.

The most important component in the success of this course was the design of the team project. To make things manageable, I assigned the same application to all the teams. However, I picked a complex and novel domain instead of the common (e.g., bookstore) examples. Student comments confirmed that this made the project more educational.

The project had five phases. At the end of each of the first three phases, after grading what was submitted by the teams, I provided a comparison of the various solutions and discussed the typical mistakes. This provided the students with a sense of variety and creativity that can be expected in application development. It also demonstrated that without proper analysis, data modeling, and data design, the application development effort will probably fail. At the end of each of the first three phases I also provided my solution to that phase. This allowed the teams to start from a solid base, facilitated in-class discussions, and simplified grading. It also provided an opportunity to discuss and demonstrate the richness and depth of proper analysis.

The first and second phases of the project required the students to prepare a data model (ERD) and a data design for the application. The third phase required an object plan, a Function Point Analysis, and the creation of the database (including all extended attributes). The fourth and fifth phases were the alpha and beta versions of the system. Each team received detailed comments on their alpha version and had to specify how they addressed them in their beta version. One mistake I made in the design of these last two phases was in not requesting that each team submit a list of problem areas in their application.

Two midterm examinations were administered, each with a written part dealing with the conceptual material and an inlab portion dealing with PowerBuilder skills.

Results

On the last week of the semester I administered anonymous questionnaires in order to assess students' reactions to the course. The results were very positive. 100% of the students indicated that the course should be offered again and that they would recommend this course to other students.

The positive response cannot be attributed to just having fun in this course; the students estimated they spent an average of 10 hours per week outside class on this course. It may
be wise to assign this course four rather than three credits. The added class time could be very useful.

The students appreciated the effect of the course on their market value. Three students indicated the course already helped them land a job. Other students indicated the course significantly helped them handle job interviews. A month later, I am aware of several students with good jobs in Client/Server application development. One student received a starting salary which is well above anything we experienced before.

Based on the reactions to the course we plan to keep offering this course once a year. In the presentation I will provide more details aimed at encouraging and helping other faculty members to introduce a similar course into their IS curriculum.