Evolution of a Freshman Software Tools Class

Garth E. Thomas Jr., Michael V. Minnick, Dianchen Gang
Chemical and Civil Engineering Departments
Leonard C. Nelson College of Engineering
West Virginia University Institute of Technology
Montgomery, WV 25136

Abstract

Three years ago, the Leonard C. Nelson College of Engineering replaced a traditional programming course for engineers with an applied software tools course. This course was expected to better prepare the students for later courses as well as develop skills that would be useful in their professional careers. Students learn the basics of Excel®, Mathcad®, and Visual Basic for Applications® programming while using them for engineering applications. While the content of the course has not changed significantly since its inception, the delivery has. Much of this change in delivery was driven by student retention. Almost half of the students either dropped the course or earned less than satisfactory grades when the course was first offered. Subsequent modifications have greatly improved retention and student performance without compromising the quality of the course.

The paper will focus on the initial design of the course, the retention issues that developed, and the modifications to course delivery that were made to address these issues. Grading policy, structure of the course content, and active learning exercises were keys to improvement. We will show how changes in these facets of course management led to better course outcomes. The paper also discusses the effects of prior computer experience and mathematics preparation on the retention problem.

Purpose of the course

The software tools course was designed as a replacement for a traditional computer-programming course. Like many other engineering programs, instruction in a programming language had been required for all engineering majors at the West Virginia University Institute of Technology (WVU Tech), and was offered during the freshman year. This course was taught by the Computer Science faculty, and used C++ as the programming language. Principle topics of this course were language syntax, logic structures, and program development. At the end of the course, students were to have a rudimentary knowledge of programming concepts and the ability to write programs that may be needed in later classes. There was also a general belief among the faculty that the process of learning a programming language would develop logical thinking skills.

However, dissatisfaction with the programming course began to develop within several of the engineering programs. Faculty members observed that the programming course was not meeting the needs of the students in terms of providing instruction for computer software they would use later in the curriculum. The course instructors outside of the Computer Science and Computer Engineering/Electrical Engineering programs were not requiring the students to write programs. The faculty that were making use of computer-aided problem solution employed spreadsheets, simulation software, and packages such as Mathcad® or Matlab®. The situation was similar to what has been found in a national survey of Mechanical Engineering programs.[1] However, the students' lack of training with software tools meant that course time had to be devoted to providing such instruction. Unfortunately, other common outcomes were that software was not used as effectively as it could have been, or was not used at all. Jones[2] observed a similar pattern at ten other institutions. A number of faculty members were unwilling to sacrifice course time to provide instruction in the use of software.

Faculty also noted that the programming instruction was not having any noticeable effect on the development of logical thinking or problem solving. This is also not a novel observation and has been discussed at length by Urban-Lurain and Weinshank.[3] However, recognition of this fact removed a primary rationale for the programming course.

Advisory boards for Chemical, Civil and Mechanical Engineering were urging a focus on the use of software packages instead of programming. Their reasoning was that almost all engineers make use of computers and software in their work, but only a limited number actually write programs. This view was in agreement with recent surveys concerning skills needed by industrial practitioners.[4,5] All three boards recommended that the programming course be converted into a software applications course. The advisory board members also expressed the opinion that such a course would be most helpful in meeting ABET 2000 criterion (k) [6], which requires program graduates to demonstrate " an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice."

However, there was an additional reason for changing the freshman computer class. For many years the college of engineering has grappled with retention of students during the freshman year. One issue that the faculty felt was important was the lack of contact between engineering faculty and engineering students during the first year. Thus, students leave engineering programs without having experienced any facet of engineering. It was desired to teach a course that would give engineering faculty contact with engineering students early in their program of study, and introduce these students to engineering concepts.

Motivated by the recommendation from their advisory board, the Civil Engineering department wanted to drop the programming course from their curriculum and develop a new one for their majors. Since the Chemical and Mechanical Engineering departments also desired a similar change, the department chairs decided to develop a common course that would replace the programming course. The Electrical Engineering department decided to retain the programming course for their majors, but the Chemical, Civil, and Mechanical Engineering departments included the new course in their curricula. This replacement was made in the fall semester of 2002. The course was initially offered via several reserved sections of the old programming course, but a new designation as GENE 111 Software Tools for Engineers was provided in the spring 2003 semester.

Course Development

Once it was agreed that a new course was needed, the next major decision was the content of the course. It was decided that Excel® would be used because spreadsheet use is so common and that particular software was available in all of the computer laboratories. It was also decided that Visual Basic for Applications (VBA®) programming would be taught since it extends the capabilities of Excel®, provides a platform to teach some generally useful programming concepts, and Co-op students and new graduates were reporting that they used VBA® in their work. Mathcad® was also included for capabilities such as advanced numerical functions and symbolic algebra. There was considerable discussion about whether Mathcad® or Matlab® should be used. Mathcad® was chosen because the civil engineers did not use Matlab® in their courses and Mathcad® was already available in several of the computer laboratories. The relative merits of the various software packages have been discussed elsewhere.[7-11] Our decisions were based on what our graduates were using in the workplace and what we were already using within the college of engineering.

It was also agreed that some basic numerical analysis be included in the class. In particular, basic operations of linear algebra, numerical solution of nonlinear equations, numerical approximation of functions, and interpolation were selected as topics of common interest. The numerical analysis work was to be applied to typical engineering problems to provide applications for the software tools. Students were provided with course notes on some of the numerical analysis topics, and the texts[12,13] used for the Excel® and Mathcad® portions of the course also contain material on numerical methods. Supplementary material was obtained from texts by Gottfried[14] and Pritchard[15].

The initial topic coverage of the course is shown in Table 1. Each session is a 50 minute period and meets three times per week. This was the format of the old programming course. It was retained in order to facilitate a simple replacement in the schedule. Approximately the first third of the course was devoted to instruction in the rudiments of the Excel® and Mathcad® software. During this portion of the course students used the software to solve simple, but practical problems. Work on these problems emphasized worksheet layout and development of graphs.

The midterm class sessions focused on numerical analysis. The students were introduced to the mathematical functions and solving tools available within the software. These tools were then applied to the numerical solution of various modest sized engineering problems. In the course of presenting the problems the instructors provided background as to the significance of the problem within the practice of engineering. This was done to achieve the goal of providing freshmen with some sense of what engineering encompasses.

The last third of the course was concerned with using VBA® to extend the capabilities of Excel®. This part of the course began with automation of Excel® tasks using macros. VBA® was introduced through modification of macros in the VBA® editor. The next level of development was to write functions to extend the numerical capabilities of Excel®. In order to enhance student interest, several of the larger projects involved the development of simple games, and provided a break from strictly technical projects. The textbook(16) used for this part of the course contains several such projects. At the conclusion of the course, the students were expected to be able to manipulate Excel® objects, use looping structures, and be able to perform input/output operations via several methods.

Another important aspect of the course design was to develop the learning outcomes for the course. The coursework was constructed so that the students would demonstrate the following abilities as they completed the problem and project assignments:

- the ability to create Excel® spreadsheets for problems of moderate difficulty.
- the ability to create Mathcad® worksheets for problems of moderate difficulty.
- the ability to use numerical functions provided within Excel® and Mathcad®.
- the ability to use Excel® and Mathcad® to solve interdependent algebraic equations.
- the ability to present numerical information graphically.
- the ability to design and write programs of moderate complexity in Visual Basic®.

The learning outcomes for this course were expected to at least partially satisfy the following ABET required outcomes[6]: (e) The ability to identify, formulate, and solve engineering problems, and (k) The ability to use the skills, techniques, and modern engineering tools necessary for engineering practice.

Table 1. Course Outline for the Software Tools Course

Topics	Sessions
Excel	6
Overview of worksheet and controls	
Numerical operations and formula editing	
Formatting and printing	
Built-in functions	
Graphing and charts	
Mathcad	8
Overview of worksheet and controls	
Editing and formatting worksheet entries	
Functions and toolbars	
Graphing	
Units	
File input and output	
Symbolic operations	
Numerical Analysis	12
Interpolation & approximation of functions	
Approximation of derivatives and integrals	
Linear algebra and matrix operations	
Solution of nonlinear equations	
Visual Basic for Applications	16
Macros and the VBA programming environment	
Data types	
Objects, properties, and methods	
Input/output operations	
Procedures	
Logical conditions and looping	
Arrays	
Forms and controls	
Error handling and debugging	
Multimedia and linking to other programs	

Student background

Almost all students come to the course with some computer experience. However, there are always a few, generally older, non-traditional students, who have very little computer experience whatsoever. These later students must come up to speed on their own since the course presentation assumes that the students have a working knowledge of the Windows® operating system, and are capable of working with files, using an Internet browser, and using an E-mail client. All students either have an E-mail address or quickly acquire one. Most begin to use E-mail on a regular basis once they realize that the course instructors use it as the primary means of communication between class meetings. The students are also encouraged to submit assignments electronically via E-mail attachments.

Although students are not required to purchase their own computers, most do or at least have routine access to one away from the computer lab. Almost all have the MS Office® software. A few also acquire a student version of the Mathcad® software. Computer access is generally not an issue except for the Mathcad® assignments.

Background surveys of students entering the software tools course show that most have some experience with Excel, though the majority has only limited experience. On the other hand, very few students have any experience with either Mathcad® or Excel® VBA®. Typical data, collected from 43 students in one instructor's section from Fall 2003 to Fall 2004, are shown in Figure 1. The data are consistent

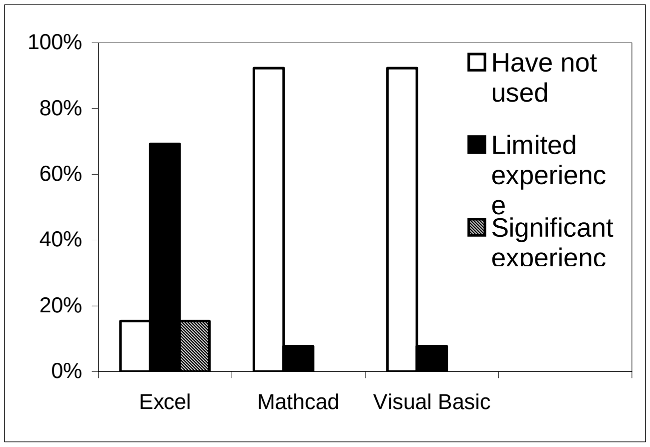

Figure 1. Initial Student Experience with Course Software.

with the results obtained in other sections of the course. These results predict that the students should handle the basic work in Excel® fairly easily, but have more difficulty with the Mathcad® and VBA® work. This has proven to be the case.

Calculus I is a co-requisite for the software tools class. Since many of our entering freshmen must enroll in algebra before taking calculus, very few students in the software tools course are entering freshmen. The vast majority of the students are in their second or third semesters at WVU Tech. While the course content does not require a background in calculus, students who are not ready for calculus are generally weak in analytical/logical thinking and have difficulty with the numerical analysis and programming concepts.

Course administration and pedagogy

It was decided that the course should be problem and project based. There were no examinations. The students were given approximately fifteen problem and five project assignments and were required to submit about eighty percent of them. The students could choose which assignments they submitted. The unusual feature was that no partial credit was given for assignments. Credit was only given if the work was substantially correct. The compensation was that the students could correct any work that was not accepted as long as it was submitted prior to a final cut-off date.

Initially, the course was taught much like the programming class it replaced. Students sat through a lecture or demonstration presented on a computer projection system. This was the first aspect of the class to be changed. The course instructors noticed a serious decline in class attendance, and several students commented that it was difficult to retain the demonstrated skills without actually practicing them. As a result, the instructors moved their classes to the engineering computer lab, and the course has been taught in the laboratory in all subsequent offerings. Lecture has also been reduced to five or ten minute concept presentations. For most of the class period, students either work through demonstrations with the instructor or work on practice problems.

Outcomes and retention issues

In spite of the modification of the course instruction, approximately half of the students either failed or earned a poor evaluation for the course during the first offering. However, many of these students had stopped attending class before the changes were made. The course instructors believed that another semester would be needed to determine if the instructional methodology would improve student performance.

One change that was made for the second semester was to set an early due date for the assignments, and then permit a fixed period for corrections after the initial evaluation. It was found during the first semester that students held off submitting anything until the cut-off date, even though early submission provided an opportunity for correction. A number of students did not have assignments accepted because the work contained significant errors or omissions and the last minute submission left no opportunity for correction. It was expected that this modification might also help with student retention.

In spite of the modifications, the overall student performance during the second semester was not improved. While the instructors were convinced that the modifications had improved the learning experience for the students who successfully completed the course, the high percentage of poor outcomes was a concern. However, to due instructor rotations, the course was offered for the third time in the fall of 2003 without further modification. The percentage of students with poor grades continued to be about fifty percent.

Anecdotal observation indicated that the classes passed through a critical period around midterm. At this point the course moves beyond basic Excel® and Mathcad® operations and begins to deal with more sophisticated applications. The students who will not complete the course generally quit submitting any work after midterm. This observation resulted in the hypothesis that the numerical analysis material and the programming were too difficult for a freshman level course. The department heads recommended that the numerical methods be dropped and that a slower pace be adopted for covering the programming topics.

However, in the meantime the course instructors had made a modification that did affect the class performance. It had also been observed that the students that ultimately performed poorly also had an irregular pattern of class attendance. Students became disengaged from the class without regular attendance, and did not gain the skills necessary for the problem and project work. As a result, they did not keep up with the assignments. Since many of the students come to the class with some Excel® experience, it is possible for them to handle the early assignments without attending class on a regular basis. This probably accounts for class performance deteriorating around midterm.

Both instructors implemented attendance policies in the spring of 2004. One of the instructors awarded twenty percent of the course credit for attendance. The other required attendance in order to pass the course, with five unexcused absences allowed. While the policies were different in operation, they had roughly the same effect. The percentage of students failing or earning a poor evaluation was cut almost in half. The course credit apparently induced more students to attend class. Students with poor attendance in the other class evidently realized fairly early that they could not pass the course and withdrew. No one failed because of excess absences. Figure 2. shows the correlation between the class attendance and course outcome for the section where attendance was awarded credit. From this figure it can be seen that class attendance played an important role in the students' grade.

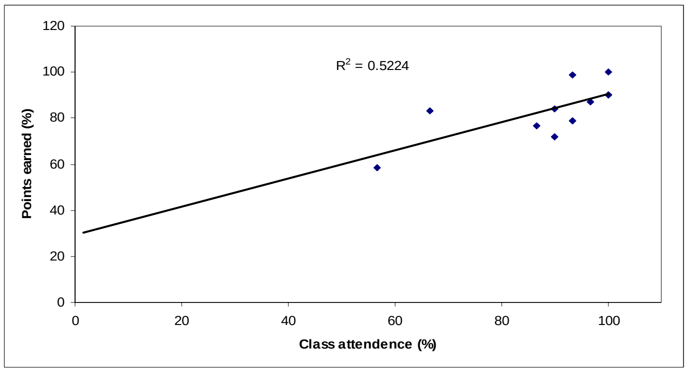

Figure 2. Relationship between class attendance and grade earned.

Since the implementation of attendance policies had a significant effect in improving student performance, the instructors for the fall 2004 sections continued this practice. However, rather than give credit for attendance, it was decided to award credit for quizzes and practice problems completed in class. Not only must students attend class; they must be actively engaged with the class activity. Instructors awarded thirty percent of the course grade for in class work. It was also decided to retain the numerical analysis topics, but to rearrange the course in order to cover VBA® programming immediately after covering Excel. This moved a difficult topic forward and provided an early indicator of how students would fare in the class. Table 2. provides an overview of the current arrangement of topics for the course.

The improvements in student performance have been maintained. Figure 3. shows the trend in the percentage of students performing poorly over the past five semesters. There is a definite shift downward in this percentage. The average percentage was fifty percent for six course sections over the first three semesters the course was offered. The average was twenty-three percent for four sections over the last two semesters. The shift is statistically significant with a p-value of 0.004. This appears to indicate that the attendance policies have had a definite impact on student performance.

While we will continue to look for improvements for this course, further reduction in the percentage of poor performing students will become more difficult. Several other problems besides class attendance cause poor performance. If a student is performing poorly in other classes, attempts to salvage their grade in the other courses may take away time that is needed to perform well in this course. Some students do not devote sufficient time outside of class to complete the assignments. This could be due to an employment conflict – some students work nearly full time outside of class, limiting reading and practice time. Poor time management also plays a role. Many students attempt problems just before the due date and do not have time to deal with unexpected problems. Lack of maturity and experience leaves students ill-prepared to deal with the workload. Lack of interest could be a factor. Some students view the class as peripheral to their major; not realizing that they will need the skills in later courses.

Assessment of the performance of the students who complete the course successfully also raises a few problems. Many students do not read text or handout material, as indicated by poor quiz grades. They attempt to complete the problems and projects based on class experience alone, or search for relevant examples. This leads to poor initial work submissions and several rounds of feedback from the course instructors during the correction phase. It has also been observed that students often do not check their work to make sure that all requirements have been satisfied. There is also a fair amount of confusion about operations in programming language that don't fit with their previous experience. For example, many students struggle with the idea that A = B in a programming language is an assignment operation and that B = A is not an equivalent statement. These are problems that will need attention as the course is further refined.

Table 2. Modified Course Outline

Topics	Sessions
Excel	8
Overview of worksheet and controls	
Numerical operations and formula editing	
Formatting and printing	
Built-in functions	
Graphing and charts	
Visual Basic for Applications	16
The VBA programming environment	
Data types	
Input/output operations	
Procedures and logical conditions	
Loops and Arrays	
Basic Excel Objects	
Forms and control	
Error handling and debugging	
Mathcad	8
Overview of worksheet and controls	
Editing and formatting worksheet entries	
Units	
Functions and toolbars	
File input and output	
Graphing	
Symbolic operations	
Numerical Analysis	10
Linear algebra and matrix operations	
Solution of nonlinear equations	
Interpolation & approximation of functions	
Approximation of derivatives and integrals	

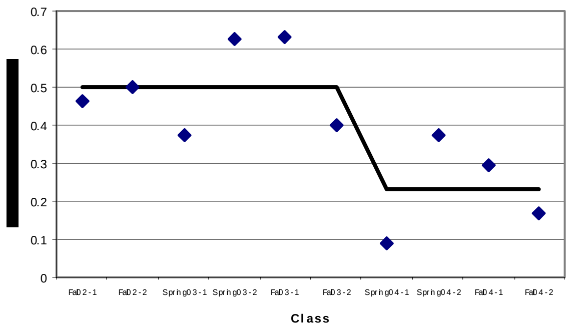

Figure 3. Trend in the fraction of students performing poorly.

The ultimate outcome is improvement of computer skills in later courses. The evidence is limited because the first cohort of students from this course is now in the junior year. However, two of the authors have had the opportunity to teach students in this group and have found that at the same point in the curriculum they were using software more often and more effectively than previous cohorts, and were able to handle more sophisticated assignments. We will conduct a more extensive survey covering all three of the affected departments, but the initial results indicate that this course is meeting its prime objective.

Future Improvements

One improvement that is expected to be in place before the next academic year is to convert from a fifty-minute format three times per week to a two-hour format two times per week, maintaining the same total course credit. The longer periods would permit the students who work at a slower pace to complete the in-class assignments in a timely fashion. Presently, every two weeks or so, a class period is used for catch-up work. This is not the most effective use of time and the proposed format is expected to work better given that the course content is now mostly activity based.

One feature of course management that needs some work is to place more emphasis on quizzes or assignments that will force the students to do the background reading. Better pre-class preparation is expected to lead to more effective learning from the classroom activities.

Some basic training in a methodology like Quality Function Deployment (QFD) might be useful in helping the students to understand and define specifications. The number of problems and projects returned for correction would be reduced if student work met all of the requirements outlined in the assignment. This would also provide the students with some experience in using a tool is widely used within the engineering profession.

Conclusion

The software tools class was developed to provide students with a set of skills that will be needed in their professional practice. Preliminary results indicate that this objective is being met. The course delivery and management has evolved to provide a better learning environment and to deal with a serious retention problem that developed. It is hoped that the solution developed for this course might be used in other courses where student retention is a serious problem.

Bibliography

1. Hodge, B. K. and W. G. Steele, "A Survey of Computational Paradigms in Undergraduate Mechanical Engineering Education," *Journal of Engineering Education*, 91, 415 (2002)

2. Jones, J. B., "The Non-Use of Computers in Undergraduate Engineering Science Courses," *Journal of Engineering Education*, 87, 11 (1998)

3. Urban-Lurain, Mark and Donald J. Weinshank, "Do Non-Computer Science Students Need To Program," *Journal of Engineering Education*, 90, 535 (2001)

4. Kantor, Jeffrey C. and Thomas F. Edgar, "Computing Skills in the Chemical Engineering Curriculum," Computers in Chemical Engineering Education, Brice Carnahan, Editor, CACHE Corp., Austin, TX, 1996.

5. Lang, James D., Susan Cruse, Francis D. McVey, and John McMasters, "Industry Expectations of New Engineers: A Survey to Assist Curriculum Designers," *Journal of Engineering Education*, 88, 43 (1999)

6. Engineering Accreditation Commission, Criteria for Accrediting Engineering Programs: 2005-2006 cycle, www.abet.org, December 2004.

7. Shacham M. and M. B. Cutlip, "A comparison of Six Numerical Software Packages for Educational Use in the Chemical Engineering Curriculum," Computers in Education Journal, IX, 9, July-Sept 1999.

8. Harb, J. N., A. Jones, R. L. Rowley, and W. V. Wilding, "Use of Computational Tools in Engineering Education: A Case Study on the

Use of Mathcad®," *Chem. Eng. Ed*., 31, 180 (1997)

9. Dorgan, John R. and J. Thomas McKinnon, "Mathematica in the ChE Curriculum," *Chem. Eng. Ed*., 30, 136 (1996)

10. Sandler, S. I., "Spreadsheets for Thermodynamics Instruction: Another Point of View," *Chem. Eng. Ed*., 31, 18 (1997)

11. Mackenzie, Judith G and Maurice Allen, "Mathematical Power Tools: Maple, Mathematica, MATLAB, and Excel," *Chem. Eng. Ed*., 32, 156 (1998)

12. Liengme, Bernard V., A Guide to Microsoft Excel 2002 for Scientists and Engineers, 3rd ed., Butterworth-Heinemann, 2002.

13. Larsen, Ronald W., Introduction to Mathcad 2000, Prentice Hall, 2001

14. Gottfried, Byron S., Spreadsheet Tools for Engineers, McGraw-Hill, 2000

15. Pritchard, Philip J., Mathcad: A Tool for Engineering Problem Solving, McGraw-Hill, 1998

16. Birnbaum, Duane, Microsoft Excel VBA Programming for the Absolute Beginner, Premier Press, 2002

Biographical Information

Garth E. Thomas Jr. is the Chair of the Chemical Engineering department at the West Virginia University Institute of Technology. His teaching duties focus on process design, modeling, numerical analysis, and application of statistical tools.

Michael V. Minnick is a Professor in the Chemical Engineering department at the West Virginia University Institute of Technology. His areas of interest are process control and process modeling.

Dianchen Gang is an Assistant Professor of Civil Engineering at the West Virginia University Institute of Technology. His expertise is in the area of environmental engineering.