Computer-Based Modeling to Enhance Problem-Solving

C. Dianne Raubenheimer
Director of Assessment
College of Engineering
North Carolina State University
dianne_raubenheimer@ncsu.edu

Amy Craig
Coordinator of Student-Owned Computing Program
College of Engineering
North Carolina State University
aecraig@ncsu.edu

Rebecca Brent
Faculty Development Consultant
rbrent@mindspring.com

Key Words: computers, curriculum, engineering, problem-solving

Abstract

The primary objectives of this project are to examine (1) how to develop students' problem-solving and computational skills early in their program of study and (2) how to further enhance those skills by reinforcing critical computing concepts semester after semester. The project is a component of North Carolina State University's (NCSU's) quality enhancement plan, which relates to the use of technology in enhancing student learning. The project began by focusing on new introductory computer-based modeling courses that were created in two engineering departments, and it has since expanded to include other departments.

Outline/ Presentation Notes

Many engineering curricula around the country are re-evaluating their introductory computer programming requirement. At NCSU, several departments have changed from the traditional Java or C++ course to something more applicable to their discipline. Realizing that the standard introductory programming course no longer appropriately complements the education of systems engineers, three departments (Textile Engineering [TE], Industrial and Systems Engineering [ISE], and Chemical and Bimolecular Engineering [CBE]) looked at similar approaches to developing or revising existing courses to help students with algorithmic thinking and problem- solving using computing.

These courses aim to educate students to model problems relevant to their specific engineering disciplines, solve these problems using modeling tools (including a range of software platforms, such as Excel with VBA), and analyze the solutions through decision support (i.e., to become "power users" not programmers).

Other departments in the College of Engineering have expressed interest in reviewing their introductory computer programming course requirement and implementing a course similar to those already developed in TE, ISE, and CBE. This is the "scale-out" portion of our project, as we seek to expand the work and develop similar introductory courses in other engineering disciplines. The second part of the project is the "scale-up" portion, which entails linking computational processes and skills across courses in the curriculum; that is, developing a computational thread at successive levels in program curricula. We acknowledge that not every course lends itself to the use of computational tools, but there are courses at successive curriculum levels where it is appropriate and beneficial to student learning for computational tools to be used and problem-solving skills to be reinforced. **Figure 1** schematically represents these two parts of our project, the "scale out" and "scale up" components.

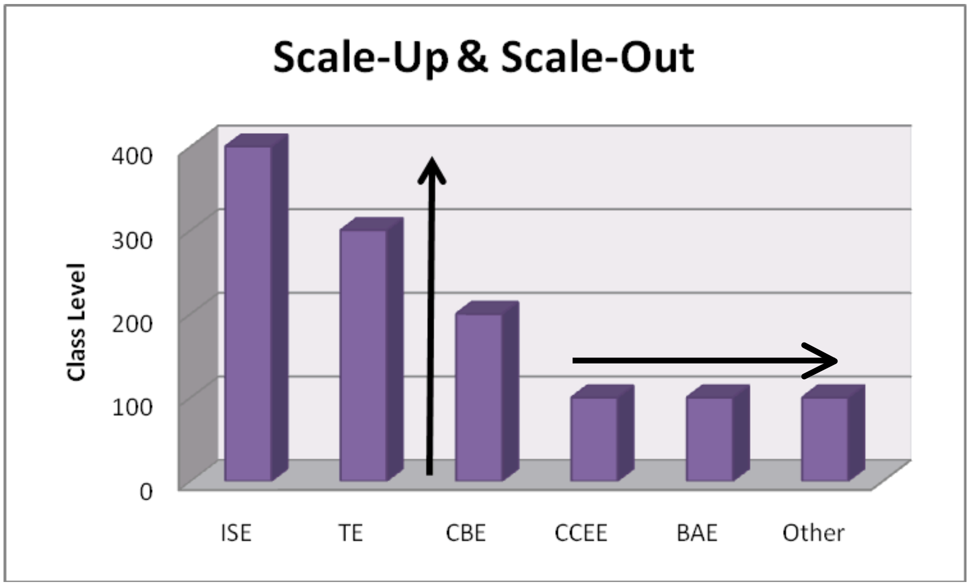

Figure 1. Schematic representation of the implementation process.

This project is funded out of the Provost's office and is part of the university Quality Enhancement Plan (QEP) called Learning in a Technology Rich Environment (LITRE), which focuses on the role of technology in enhancing student learning. It is one of three large technology projects, with the potential for wide-reaching impact across the campus, selected for support in the Phase II part of the QEP (July 2007–July 2009).

Assessment Activities

The project has four main research questions, each being assessed using different instruments.

Overall Research Questions

1. What are student approaches to modeling and problem-solving and how do they change over time as students move into upper-division courses and use programming and computational tools to model and solve discipline-specific problems?

   Instrumentation: (a) Student surveys assessing attitudes and confidence in specific course outcomes, (b) survey questions about modeling and problem-solving, (c) specially developed, common problem-solving tasks, (d) student reflections about how they go about solving tasks, and (e) course-related samples of student work. The survey data is being compared to students completing these same or similar surveys in subsequent semesters, to establish trends in self-confidence as students are exposed to the new course sequences. The problem-solving tasks and reflections are being compared to students in upper-division courses who have not been through the new course sequences and to longitudinally track the problem-solving abilities of students who do experience the new course sequences.

2. What characteristics (e.g., gender, GPA) do the learners bring to problem-solving processes that assist or hinder their success as modelers and problem-solvers?

   Instrumentation: (a) Student profile data (e.g., GPA, gender), and (b) student surveys, (e.g., the Revised Study Process Questionnaire), and self-efficacy and beliefs about the course survey. These data will be related to data on student problem-solving, student performance in courses, and other student surveys.

3. Does student performance in the discipline improve with the new approaches to teaching modeling and problem-solving?

   Instrumentation: (a) Course-specific, end-of-semester GPA comparisons to the same courses in previous years.

4. How do the various faculty involved in the project use technology inside and outside of the classroom to enhance student learning?

   Instrumentation: (a) Baseline faculty survey, (b) field notes of research participants, and (c) faculty interviews.

Lessons Learned

While we are still early in the project, we have learned a number of lessons about increasing computer use in engineering departments.

- *Change is hard, and takes more time than expected.* We should all know this by now, but it still comes as a shock that it takes so long to make changes happen. While people who are first-adopters will readily jump in and try new things, the rank and file faculty take much longer and are much harder to convince that change is even necessary.
- *The cultures of different departments are very diverse and must be taken into account.* The three departments we are working with vary in the leadership styles of their department heads, the readiness of the faculty to embrace the changes, and the computational needs of the students. It is important to the success of the project to be sensitive to the differences and flexible in designing the course changes and assessments.
- *Talking about teaching is not a common activity in most departments.* In our initial meetings, it was often obvious that faculty had almost no idea about what their colleagues were doing in their classes. One unexpected value of a project like this one is that it gets people talking to each other and sharing ideas.
- *Assessment activities are critical for helping faculty realize there is a problem, become galvanized to take action, and continue to implement changes.* When faculty are confronted with deficiencies in student confidence and skill levels, they are more receptive to making changes to address them. By the same token, when there is ongoing assessment of activities, faculty are able to see results and are encouraged to continue making the effort to change.
- *Departmental leadership is needed to get faculty to the table.* Department heads have the leverage to encourage faculty to make good-faith efforts to modify their

courses. It can help the project considerably if the department head strongly supports the change and participates in some of the discussions.

- *Champions within departments are critical to success.* Given how challenging it is to make curricular changes, having respected senior faculty who believe in, and participate in, the project makes everything go more smoothly. Such faculty can mentor their colleagues as they try new things and can be positive voices in faculty discussions. Champions are often the first to try out new tools and techniques and have the enthusiasm to keep trying, even when problems arise.
- *Support for computer problems and activity development is an important element in getting faculty to incorporate computational tools.* The one thing all faculty have in common is not enough time to do everything they need to do. Locating and developing problems and activities takes focused time, time most faculty are not willing to take. We have found that providing problem development and pedagogical support to faculty facilitates their making changes they would be unlikely to attempt without such support.

URLs

North Carolina State University, LITRE—Learning in a Technology Rich Environment: A Quality Enhancement Plan for North Carolina State University. http://litre.ncsu.edu/dfiles/COE.html (accessed March 10, 2008).

Wolcott, Susan K. 2006. Steps for Better Thinking: Developmental Problem Solving Process. http://www.wolcottlynch.com/EducatorResources.html (accessed March 10, 2008).

Bibliography/ References

Felder, R., and R. Brent. 2000. Is technology a friend or foe of learning? *Chemical Engineering Education 3* (4):326–327. Also available at http://www.ncsu.edu/felder-public/Columns/Technology.html (accessed March 10, 2008)

Joines, J., S. Roberts and C. D. Raubenheimer. 2007. Computer-Based Modeling for Engineers using Excel and VBA. *Proceedings of the 2007 American Society for Engineering Education International Conference,* Honolulu, HI. Also available at http://www.asee.org/acPapers/code/getPaper.cfm?paperID=13338&pdf=AC2007Full3009.pdf (accessed March 10, 2008).