Computer Science and Engineering Curricula

V. Vemuri

Au: Figures will be put in numerical order and page makeup rules will be followed before your article is printed. Ed.

ED: please provide manuscript received dates, etc.

Abstract— **The challenges of maintaining relevancy and currency in computer engineering curricula are examined. The emerging trend to emphasize design experience early on, the need to stress written and oral communication skills, and the need for industry-university collaboration in funding instructional laboratories are identified as the three main ingredients for success in an undergraduate computer engineering curriculum.**

I. Introduction

ONE of the perennial challenges faced by educators is the need to maintain currency and relevancy to their curricula in the face of changing times. The new Information Revolution, spurred by the development of low-cost and high-speed computers, is touching every facet of our lifestyle. With PC-sized supercomputers, cellular radios, and fax machines becoming as ubiquitous as radios and televisions; with microprocessors controlling such household goods as washing machines, dishwashers and ovens; and fuzzy logic and neural nets becoming common household words, the modern computer and communications technology is playing a pivotal role in the economic well-being of many nations. Has the academic community recognized this new reality? Are the curricula keeping up with the current changes and challenges?

During the past twenty years, this emerging discipline has assumed various names. For convenience, we refer to this here as computer science and engineering (CSE). In addition to the rapid pace of developments in this area, other important forces are at work prompting a need for frequent curricular revisions. The needs of employers and the characteristics of a computer engineering career are continually changing. The preparation and motivation of students are also experiencing a transformation. These phenomena are not entirely new to our times, nor are they confined to CSE. Engineering educators faced several such challenges in the past and they routinely responded with recommendations for model curricula for four-year undergraduate programs [1]–[5]. Indeed, there has never been a shortage of studies on model curricula. (A sample model curriculum is shown in the appendix.) What seems to impede progress is that curricular recommendations have the tendency to exhibit tremendous "implementation inertias." They resist all but the most incremental changes.

II. The Symptoms and Solutions

The "Phoenix Committee" (at Worcester Polytechnic Institute in Massachusetts), named thus to reflect its mission of completely overhauling the undergraduate engineering curriculum, conducted an extensive survey of their EE alumni and identified some important categories of comments received from 300 of the respondents [6]. I conducted my own, albeit informal and anecdotal, survey of some undergraduate and graduate students. The results from these surveys, summarized below, identify the following needs:

1) More practical and hands-on experience, such as a research or design project, to accompany traditional lectures.
2) Courses that emphasize the computer as an engineering design tool, in general, and improved computer literacy, in particular.
3) Diversity of courses beyond the (computer engineering) major. The scope of this diversity included not only courses from other engineering disciplines, such as thermodynamics, fluid mechanics, materials, and project management, but also topics in general education, especially writing and rhetoric.
4) Cultivation of the art of thinking and communicating clearly. Many "computer-generation" students learn to communicate only with their terminal or workstation. They are ill prepared in written and oral communication skills as well as in interpersonal skills.
5) A philosophical approach that makes the student realize that the whole is more than the sum of the parts. The body of knowledge a student should learn is often parcelled into semester- or quarter-long courses.

Many students have difficulty realizing the interrelationships among these subjects or how one uses the knowledge thus acquired to solve a practical problem.

The following remedial steps were often suggested by the respondents of the above survey. First, requiring the students to attend at least two sets of undergraduate seminars in two different terms. Unlike graduate seminars, these are tutorial and expository in nature with the topics drawn from contemporary research issues, but presented in an exciting and thought provoking manner. The speakers, drawn from both industry and academia, should be carefully selected for their ability to inspire and excite, with the speaker's own contribution, to the topic under discussion, being a secondary consideration. The idea is to offer one of these seminars in the freshman year, the formative period in a student's undergraduate life, in order to assist the student to make an informed decision in selecting a major. Students are required only to listen and make a brief report at the end of the term. The goal here is to sustain and nourish any budding interest in the sciences (say, CSE), thus forestalling the possibility of students switching majors or even dropping out from engineering.

Manuscript received
The author
IEEE Log Number 9206224.

The second remedial step is to require all freshman engineering students (native as well as foreign) to take a one term course in writing or rhetoric [7]. The third step is to require students to write a term paper, not in lieu of any written laboratory reports in a design engineering course. This can conveniently be done in conjunction with the second seminar course suggested earlier; the term paper topic could be one inspired by the topics of the seminar. This is typically done in the junior and senior years. Selecting a topic, doing a literature search, scoping the problem, doing the problem without knowing the "correct answer," documenting the results concisely, and making a brief presentation of the results in front of a class are skills that are as important as writing a correct program and proving it correct.

III. Implementation

How do we compress all this course work into a four-year curriculum? Even the best of students find it necessary to spend 12 hours per course per week. Further demands on a student's time will only be counterproductive as there is less time left for reflection and contemplation. This brings us to one of the frequent challenges faced by curriculum designers, namely striking a compromise between what is basic and what is current while balancing the department's ever shrinking budgets. Preserving the basics shall always be the prime objective of any curriculum. Pursuing this objective in its purest form has its pitfalls; rote memorization exercises and drills will surely turn off the student seeking a challenge.

Currency and relevancy in computing could be influenced by cooperative efforts between industry and universities. The common complaint that academia really does not understand the problems that are important to industry is, I believe, motivated by short-sighted considerations. The prevailing "client-server model" that portrays the student as the product of the "university factory" for consumption by the industrial customer is wrong. Corporations, with an enlightened self interest, should participate with universities in the development of the necessary instructional infrastructure. These efforts should go beyond the traditional cooperative education, it should also include the development of instructional and research laboratories, an area where severe budgetary disincentives exist.

Appendix

The Computer Society of the IEEE published a model curriculum for CSE in 1977. By 1981, the society recognized the changing trends of the times and commissioned a committee (on which this author served as a member) with the charge to make a major revision. This committee's report, *The 1983 IEEE Computer Society Model Program in Computer Science and Engineering* appeared in the April 1984 issue of the COMPUTER magazine, along with several other articles on the role of computers in education. The committee eschewed identifying a single complete curriculum spanning four years. "They preferred instead to identify a body of knowledge that should be a part of any curriculum. The criteria for selecting the fundamental concepts—the core of the curriculum—were to provide a student with broad background in engineering principles coupled with in-depth knowledge of hardware, software, application tradeoffs and the basic modeling techniques used to represent the computing process. Another criterion for the core was that it stay within a 33-semester credit hour limit." With these objectives, the following 13 "subject areas" were identified as the Core of the Model Program. The Lecture/Recitation components of the Core Model Program are:

1) Fundamentals of Computing,
2) Data Structures,
3) System Software and Software Engineering,
4) Computing Languages,
5) Operating Systems
6) Logic Design,
7) Digital System Design
8) Computer Architecture, and
9) Interfacing and Communications.

The Laboratory components of the Core Model Program are:

1) Introduction to Computing Laboratory,
2) Software Engineering Laboratory,
3) Digital Systems Design Laboratory, and
4) Project Laboratory.

Then the committee went on and provided a detailed description of each of these core areas.

The committee also listed 15 advanced subject areas, for in-depth study. The committee felt that an individual institution should string together these subject area modules in ways that are suitable to their individual needs. That no other Model Curriculum came out of the Computer Society during the past decade attests to the apparent satisfaction felt by the Society with its 1983 Model Curriculum.

Acknowledgment

The author would like to thank Sunil Vemuri, a graduate student at Stanford University; Mytilee Vemuri, a freshman at the University of California at Davis and Jim Shackelford, Associate Dean of Undergraduate Studies, University of California, Davis for helpful discussions.

References

[1] Nat. Acad. Eng.: Cosine Commit., Commission on Eng. Educ., *Computer Science in Electrical Engineering*, Washington, DC, 1967.

[2] IEEE Computer Society: Educ. Commit., *Model curricula in Computer Science and Engineering*, Rep. No. EH0119–8, IEEE CS Press, Los Alamitos, CA Jan. 1977.

[3] ACM Education Board: Commit. Comput. Curricula, *ACM Recommended Curricula for Computer Science and Information Processing Programs in Colleges and Universities 1968–1981*. New York: ACM Press, 1981.

[4] J. T. Cain, G. G. Langdon, Jr., and M. R. Varanasi, "The IEEE computer society model program in computer science and engineering," *Computer*, IEEE Computer Society Press, vol. 17, no. 4, pp. 8–17, Apr. 1984.

[5] "The ACM Response: The scope and directions of computer science," *CACM*, vol. 34, no. 10, pp. 121–131, Oct. 1991.

[6] D. Christiansen, "Do students really get it?", *IEEE Spectrum*, vol. 20, no. 6, p. 17, June 1992.

[7] V. Booth, *Communicating in Science: Writing and Speaking*. Cambridge, MA: Cambridge University Press, 1985.