

ICER Position Paper: Sarah Douglas, CIS Dept., University of Oregon

Andrew Grove, President of Intel until 1998, wrote a highly influential book on business management called *Only the Paranoid Survive* (Random House, 1999). In this book he observes that the business environment is chaotic and unforgiving, and full of what he calls “strategic inflection points”—transformational changes that, if grasped, can guarantee a firm’s growth, and, if not, can cause its death. For example, at one time the computer business was a vertical industry with competing computer manufacturers providing chips, hardware, OS, software, and sales and distribution. (Think of IBM, Univac, DEC in the 1960’s and 70’s.) Since then the industry has moved to a horizontal industry with individual, competing companies specializing in one of those product lines. Companies who succeed under one business model often can’t acknowledge its impending collapse. (Only IBM is left!)

Most of us would accept this general model of business, and can indeed find it a general model for predicting and explaining organizational change. The problem with using the model is knowing that there is a true “strategic inflection point”. The trick in business management is anticipating that such a change is occurring and then having the creativity and ability to transform the business to adapt.

Most of you know where I am going with this example. So my first question: Is computing education is undergoing a strategic inflection point as measured by its enrollment? The organizers have suggested that this is so and have offered the plunging enrollment in general and particularly among women and minorities as evidence. An inflection point, mathematically, is when the slope of a curve changes sign, for instance, from positive to negative: Is the enrollment decline just part of cyclic variability, or indicative of critical failure?

I would argue that we are, indeed, in the middle of a strategic inflection point, and if we don’t change our business model we may go the way of Univac, DEC and Burroughs. The reasons for declining enrollment have been given generally as the perceived lack of jobs due to the dot.com collapse and outsourcing to other countries. I think the problem goes much deeper. For example, I think a lot of young people today find the typical computer science curriculum uninteresting and somewhat irrelevant to the kinds of computer-related jobs that they know are available. The “hacker” or “gamer” image of the computer nerd turns off many women and minorities. (I note that these and the “entrepreneur” image have replaced “professional”.) Finally, computer science departments have tended to cultivate the reputation of offering challenging programs primarily for the very smartest and hardest working students. A computer science degree is not for the average student.

At the root of these difficulties is a more profound change in computing itself. During the past thirty years, computing has changed from a primarily scientific and engineering enterprise into a technology producing industry. Technology is the “employment of tools, machines, materials, and processes to do work, produce goods, perform services, or carry out other useful activities.” (*Academic Press Dictionary of Science and Technology*) Advances in developing usability of user interfaces and end-user programming have radically changed the skill levels required to use a computer to do sophisticated work. (Spreadsheets are extremely versatile and powerful programming systems.) Furthermore these application programs have the functionality to adapt to many different knowledge domains. The computer program itself has now become a consumer product and commodity.

What are the ramifications for computing education? A person educated as a scientist or even an engineer is simply not appropriate for this type of technology production. What is needed are people who are computing savvy and domain specific. Technology requires context-specific application that can only come from knowledge in that field. Perhaps this is what is meant by “integrative” computing education. Integrating computing technology into a context also means paying attention to human-computer interaction and social/ethical issues.

So how do we structure computing education to teach this new type of professional? I believe that we need to offer an information technology major with a combined computing core knowledge and domain core knowledge. And these need to be exciting. For example, here are some ideas in Table 1.

Table 1

Information Technology major	Domain disciplines
Bioinformatics, Medical informatics	Biology, Chemistry
Simulation Systems (Climate modeling)	Meteorology, Geophysics, Math
Game & Web design, Multimedia performance	Art, Graphics design, Dance, Theatre, Music
Robotics	Mechanical Engineering, Human biology
Technology and Democracy	Political science, sociology, philosophy
Digital Libraries, Open source software	Library science, Journalism
Natural language technology	Linguistics, psychology, foreign language
E-commerce	Business, economics, sociology

Students entering an IT major would also have a minor or second major in a domain discipline. The title could be “Computational *foo*” or “*Foo*-informatics”, as in “Neuro-informatics”. The focus of the curriculum in computing would be on areas such as human-computer interaction, programming, software engineering, social & professional issues, and information systems. These would be terminal BS/BA degrees. Students obtaining these degrees would find many employment opportunities. The degree would also be attractive to women and minorities who already major in large numbers in socially relevant disciplines such as psychology, sociology, and biology.

Lest you think I have abandoned traditional computer science, I believe there is still a role for traditional CS education as a science/engineering discipline. It should be targeted as a MS degree (5 year BS/MS combined?). The goal of this CS intensive education is for the reduced number of specialized students who will write the compilers, operating systems, sophisticated applications, etc. And there will always be a limited role for the development of high-risk, complex, unique systems that will require engineering rather than technology production. The enrollment in the CS major will be few.

What should the curriculum require for these two different types of majors? The following table is constructed from the ACM-IEEE Computing Curriculum 2001: Computer science body of knowledge core topics¹ (Final report, Figure 5-1, page 17).

Table 2:

<i>major</i>	DS	PF	AL	AR	OS	NC	PL	HC	GV	IS	IM	SP	SE	CN
IT		all	1-3			1-4	1-6	all	1-2	all	all	all	all	all
CS	all	all	all	all	all	all	all	1-2	1-2		1-5	1-8	1-8	

Finally, what should NSF do to test out these ideas? NSF could establish grant competition for “centers of excellence” in IT education such as the areas in Table 1. Proposals should select one or several areas for specialization and require joint participation of a domain discipline department. They should be designed as models for transfer to other institutions and include sufficient evaluation to determine their effectiveness. I would also encourage industry partnerships with a summer internship program.

None of what I have proposed in this paper is new. My particular contributions are (1) to offer arguments that we really do need to change computing education to accommodate computing technology education in addition to computer science/engineering; and (2) propose a specific curriculum to implement an information technology major.

¹ Discrete Structures (DS), Programming Fundamentals (PF), Algorithms and Complexity (AL), Architecture and Organization (AR), Operating Systems (OS), Net-Centric Computing (NC), Programming Languages (PL), Human-Computer Interaction (HC), Graphics and Visual Computing (GV), Intelligent Systems (IS), Information Management (IM), Social and Professional Issues (SP), Software Engineering (SE), Computational Science and Numerical Methods (CN)