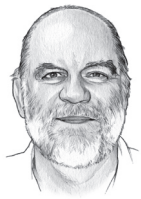



DOI: 10.1145/1364782.1364792

Stephen J. Andriole Eric Roberts

Point/Counterpoint
Technology Curriculum for the Early 21st Century

In case you missed IT, the world has changed.

Point: Stephen J. Andriole

THE FIELD OF information technology is changing and those responsible for educating the next generation of technology (information systems, computer science, and computer engineering) professionals have responded with curriculum that fails to address the depth, speed, or direction of these changes. If we want our students to enjoy productive and meaningful careers, we need to radically change the content of the curriculum of our technology majors. [a]

Change

The structure of the hardware/software/services technology industry is changing—morphing quickly from a set of fragmented hardware and software activities and vendors to a hardware/software/service provider model dominated by a shrinking number of vendors. IBM, Microsoft, HP, Dell, Intel, Oracle, Cisco, Accenture, EDS, CSC, and a few other companies are included in this $10B+ in revenue per year group.[b]

Is it improper to profile what these companies do and reverse engineer curricula? Most of these companies have robust R&D programs, manufacture hardware and software, and solve industry problems with technology. Their activities might well provide a useful—and obviously relevant—curriculum roadmap.

Another major trend is the standardization of software packages as the primary platform on which large enterprises compute and communicate. The software necessary to connect disparate software is no longer exclusively defined as proprietary middleware; instead, it's embedded in applications by the major vendors through interoperability standards based on Web Services and its extensions, service-oriented architecture (SOA), and event-driven architecture (EDA). Software is also installed less as more more companies rent applications from hosting vendors like Salesforce.com, Microsoft, and now even SAP. Many CIOs really want to get out of the enterprise software acquisition, deployment, or support business: the demand curve for software-as-a-service (SaaS) is steep. [c]

Relatively few vendors will produce most of the world's mainstream software in the coming years. The standardization of software will result in a concentration of software suppliers complying with a set of expanding integration and interoperability standards incarnated in evolving (service-oriented and event-driven) architectures. Just look at the mergers and acquisitions that have occurred in the software industry over the past few years. How many business intelligence (BI) vendors are independent? How many enterprise resource planning (ERP) vendors are left? Finally, greater amounts of software will exist only on servers accessed by increasingly thin clients. "Thin clients"—which have no local processing—will replace many "fat clients"—machines with lots of software, processing power, and storage.

When we layer outsourcing trends onto software trends, we see industry turning to offshore providers to satisfy their operational support requirements rather than U.S.-educated professionals who are not receiving enough of the knowledge or skills that industry values (or is willing to pay for, compared to offshore labor rates). Today those requirements are relatively low-level operational requirements but over time offshore providers will climb the food chain to more strategic technology capabilities. It's these latter areas that should

a. A more extensive discussion of these issues can be found in Stephen J. Andriole, "Business Technology Education in the Early 21st Century: The Ongoing Quest for Relevance," *Journal of Information Technology Education*, September 2006. I am referring here primarily to our undergraduate educational efforts.

b. Matthew Aslett, "CBR 50 Largest IT Vendors," *Computer Business Review*, July 19, 2006.

c. SaaS is growing in popularity as more companies appreciate the benefits of renting software. This avoids the in-house implementation phase and large enterprise software licensing fees. Industry analysts from the Gartner Group and Forrester Research, among others, report that by 2012 25% of all software will be rented. The decline of proprietary software will also be accelerated by the rising adoption of open source software.

catch the attention of U.S. educators preparing their students for technology careers since the sourcing battle for technology infrastructure and support is all but over—those who seek support for computing and communications infrastructures are driven more by labor rates than tradition, more by the advantages of commoditization than by customization. In fact, methodologies like ITIL (Information Technology Infrastructure Library) and COBIT (Control Objectives for Information and Related Technology) increasingly provide the means to cost-effectively manage and optimize infrastructures, making the infrastructure support business even less generous to technology professionals.

There are other trends changing the industry. R&D outsourcing is expanding. Data mining has become customer profiling, customization, and personalization. Supply chains are becoming transparent and have gone global. Real-time dynamic pricing (via intelligent rules engines) is spreading. Adding to this is the convergence of all things digital.

Where does technology curriculum address all of these trends? Where are the academic programs and certificates in SOA, EDA, hosting, SaaS, integration and interoperability, Web 2.0, Web 3.0, thin-client architecture, Web Services, open source software, sourcing and technology performance management? Where do students learn about interoperable architectures, roaming connectivity, real-time processing, rich converged media, user-generated content, global supply chain optimization, full-view business intelligence, predictive analytics, master data management, and crowdsourcing-based problem-solving?

Response

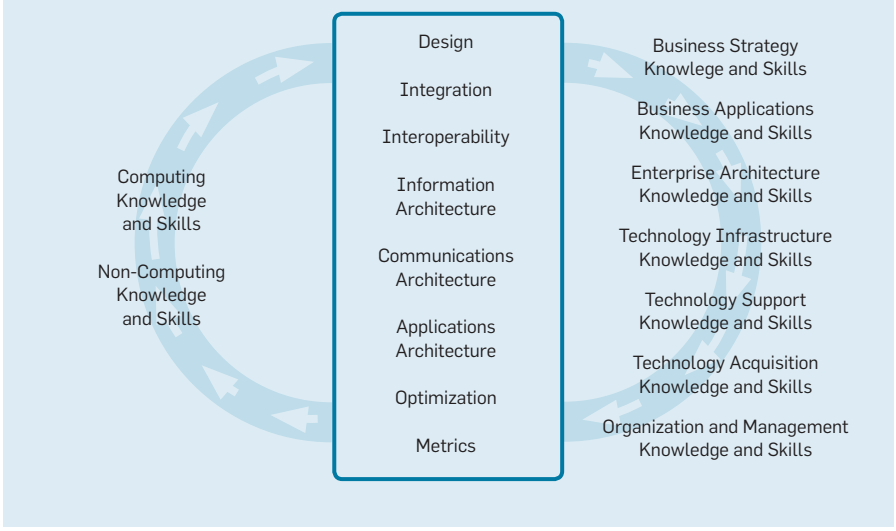
Several curriculum changes and guidelines have been proposed that attempt to address the changes in technology and design optimal pedagogical approaches in response to these changes. The Joint Task Force for Computing Curricula on Computing Curricula for the early 21st century identified five areas of computing degree concentrations: computer engineering, computer science, information systems, information technology, and software engineering.

These areas represent the academic programs that the Joint Task Force believes represent the state of the field and the educational outcomes our students should pursue. They've identified a suite of "computing" and "non-computing" areas that students in each of the five areas should understand. The list of knowledge and skills areas identified by the Joint Task Force that defines the components of the five areas was derived from academic programs and curricula that have evolved over a long period of time. I collected some data that also identified knowledge and skills areas—but from a practitioner's perspective.[d]

The table here presents the two sets of knowledge/skills areas side-by-side. The contrast is dramatic. The Joint Task Force's list barely correlates with the list developed from the practitioner surveys. Academic programs should acknowledge the widening gap between theory and practice, especially since it has enormous impact on their students' employment prospects. Regardless of what we call the academic majors and degrees, it's the content of each degree's curriculum that will determine our students' ability to find gainful employment.

One of the most important corporate knowledge areas today—in fact, the essence of business technology convergence—is enterprise architecture. Enterprise business-technology architecture is the linchpin among business strategy, strategic applications, technology infrastructure, and technology support. As business is enabled by technology and technology defines new business models and processes, the importance of enterprise business-technology architecture is increasing. This emerging core competency for the practice of the technology profession is unrepresented in the Joint Task Force's list of knowledge and skills areas—though it is a huge area in our practitioner survey. Similarly, business technology optimization is an opportunity area for educators. Increasing numbers of companies are struggling to optimize the performance of their software applications, networks, database management platforms, and infrastructure.

Knowledge and skills areas and bridges.



d. During the period from 2002–2005, an online survey sponsored by the Cutter Consortium (a technology industry research organization; www.cutter.com) collected data from Chief Information Officers (CIOs), Chief Technology Officers (CTOs), technology managers, Chief Executive Officers (CEOs), Chief Financial Officers (CFOs), technology consultants and vendors about the content of the field, the skill sets necessary to succeed, and the technologies most likely to be applied, neglected, or decommissioned. Over 1,000 professionals responded to the survey. The survey data was subsequently presented to—and validated by—the Villanova University CIO Advisory Council, which consists of 25 CIOs from the Philadelphia, PA region.

Recommendations

While distinctions between computer engineering (CE) and the other disciplines are relatively easy to appreciate—especially because of the role that hardware plays in CE programs—the differences between information systems, information technology, software engineering, and computer science are much more difficult to understand and define, especially when we reference the changes occurring in the field. I believe there should be three flavors: computer engineering, computer science, and information systems.[e]

CS programs should focus less on alternative programming languages and more on architectures, integration, and interoperability; less on algorithms and discrete structures and more on software engineering best practices. SOA and EDA should be owned by CS curricula as should Web 2.0 and 3.0, SaaS, thin client architecture, digital security, open source software, interoperable architectures, roaming connectivity, near-real-time processing, and rich converged media, among other related areas. Programming? Who programs? And where does—and more importantly, will—programming occur? Programming will ultimately evolve to component assembly and components will be generated by relatively few professionals located in the U.S., Bangalore, Moscow, and Shanghai working for IBM, Oracle, SAP, Microsoft, Tata, Infosys, and Google. Put another way, is "programming" the core competency of computer science?

Of course there will be programming jobs for our students. But the number of those jobs will decline, become more specialized, and distributed across the globe. A simple metric: how many Fortune 1000 companies still hire programmers? In the 1980s and 1990s, companies like CIGNA—where I was CTO—had hundreds of programmers on staff. Today, Fortune 1000 companies have far fewer programmers than they did because of the rise of packaged applications and the labor-rate-driven sourcing options they

e. The focus here is on the relationship between computer science and information systems; CE will likely remain primarily hardware focused and in engineering colleges within the nation's universities.

ACM Joint Task Force knowledge and skills areas and practitioner areas.

ACM Task Force Areas
Computing Knowledge and Skills

- Programming Fundamentals
- Integrative Programming
- Algorithms and Complexity
- Computer Architecture and Organization
- Operating Systems Principles and Design
- Net Centric Principles and Design
- Platform Technologies
- Theory of Programming Languages
- Human-Computer Interactions
- Graphics and Visualization
- Intelligent Systems (AI)
- Information Management (Database) Theory
- Information Management (Database) Practice
- Scientific Computing (Numerical Methods)
- Legal/Professional/Ethics/Society
- Information Systems Development
- Analysis of Technical Requirements
- Engineering Foundations for Software
- Engineering Economics for Software
- Software Modeling and Analysis
- Software Design
- Software Verification and Validation
- Software Evolution (Maintenance)
- Software Process
- Software Quality
- Computer Systems Engineering
- Digital Logic
- Distributed Systems
- Security: Issues and Principles
- Security: Implementation and Management
- Systems Administration
- Systems Integration
- Digital Media Development
- Technical Support

Non-Computing Knowledge and Skills

- Organizational Theory
- Management of Information Systems Organization
- Decision Theory
- Organizational Behavior
- Organizational Change Management
- E-business
- General Systems Theory
- Risk Management (Project, Safety Risk)
- Project Management
- Analysis of Business Requirements
- Embedded Systems
- Circuits and Systems

Practitioner Areas
Business Strategy Knowledge and Skills

- Collaboration
- Customization and Personalization
- Supply Chain Management
- Business and Technology Convergence Strategy
- Competitor Intelligence
- Business Process Management

Business Applications Knowledge and Skills

- Business Application Optimization
- Core Business Applications Management
- Business Analytics

Enterprise Architecture Knowledge and Skills

- Applications Architectures
- Data Architectures
- Security Architectures
- Business Scenario Development
- Enterprise Technology Architecture Modeling
- Enterprise Architecture

Technology Infrastructure Knowledge and Skills

- Messaging/Workflow/Calendaring
- Automation
- Database/Content/ Knowledge Management

Technology Support Knowledge and Skills

- Desktop/Laptop/PDA/Thin Client Support
- Data Center Operations
- Server Farm Design and Maintenance
- Network Design and Support
- Security and Privacy
- Procurement and Asset Management
- Asset Disposal

Technology Acquisition Knowledge and Skills

- Business Technology Acquisition Strategy
- RFP and SLA Development

Organization and Management Knowledge and Skills

- Reporting Relationships
- Centralization and Decentralization
- Governance

now have. This trend will accelerate resulting in fewer programming jobs for our students. Should we continue to produce more programmers?

In addition to the basics like data communications, database management, and enterprise applications, 21st-century IS programs should focus on business analytics, supply-chain optimization, technology performance management, business process modeling, full-view business intelligence, sourcing, and large amounts of technology management skills—in short, many of the items on the list of practitioner knowledge and skills.

CS programs can enable IS programs. The knowledge and skills areas proposed by the Joint Task Force should be extended to link to the knowledge and skills on the IS side. Clearly, the programs need to be coordinated—if we want to produce marketable human products.[f] The figure here suggests how this might work. The Joint Task Force knowledge and skills areas appear on the left and the practitioner knowledge and skills appear on the right side of the figure. In the middle are some "bridges" that might shrink the gap between the two areas. These bridges might become required for both CS and IS curricula and help CS programs become more relevant and IS programs more grounded in the enabling technology that supports business processes and transactions.

The essence of these suggestions is that CS and IS curriculum must dramatically change if we are to help our students compete. What was technologically significant 10 years ago is not nearly as significant today: hardly anyone needs to know how to program in multiple languages or craft complex, elegant algorithms that demonstrate alternative paths to the same computational objective. We know more about what software needs to do today than we did a decade ago—and you know what? There's less to do and support. This is the effect standards and commoditization have on an industry.

Our job as educators is to prepare students for the technology world-to-be, not the-one-that-was. A simple way to design new CS and IS curriculum is to observe what practitioners do today, project what they'll do tomorrow, and then identify the requisite enabling technologies (which will lead to new CS curriculum) and applied technologies and best practices (which will lead to new IS curriculum). I have attempted to energize this process by contrasting the Joint Task Force and practitioner knowledge and skills areas. I believe strongly in relevance-driven education and training, but also realize that not everyone believes education and training are closely related or that universities are responsible for preparing students for successful careers. Many believe the creation and communication of selected knowledge—regardless of its relevance to practice or professional careers—is the primary role of the modern university.

Differences of opinion are usually healthy, so let the debate begin. C

f. Most CS and IS programs exist on islands in most universities. They seldom coordinate curricula and generally have relatively little contact.

Stephen J. Andriole (stephen.andriole@villanova.edu) is the Thomas G. Labrecque Professor of Business at Villanova University where he conducts applied research in business-technology convergence.

Counterpoint: Eric Roberts

AS I READ Stephen Andriole's critique of computing education, I was reminded of the classic South Asian folk tale of the blind men and the elephant. You know the story: six blind men each try to describe an elephant after touching only a part of it. The trunk is like a snake, the tail is like a rope, the ear is like a fan, and so on. Each description contains a kernel of truth, but none comes close to capturing the reality of the elephant as a whole.

Andriole's characterization of computing in the early 21st century suffers from much the same failing in that it attempts to generalize observations derived from one part of the field to the entire discipline. He begins by observing, correctly, that the last few years have seen increasing "standardization of software packages as the primary platform on which large enterprises compute and communicate." But enterprise software is only part of the computing elephant. Computing is integral to many sectors of the modern economy: entertainment, education, science, engineering, medicine, economics, and many more. In most of those sectors, software is far from being a commodity product. Innovation in these areas continues to depend on developing new algorithms and writing the software necessary to make those algorithms real.

As an example, software development remains vital in the video game industry, which accounts for more than $10 billion a year in revenue. This sector is looking for people with an entirely different set of skills than those Andriole enumerates in his survey of "professionals" in the field—a category that he restricts largely to senior management concerned with enterprise-level information technology. That the hiring criteria of a CIO for a Fortune 500 company would differ from those of a video

The Blind Men and the Elephant

ILLUSTRATION BY CLARA E. ATWOOD FROM AUGUSTA STEVENSON'S CHILDREN'S CLASSICS IN DRAMATIC FORM: BOOK II. 1908.

game developer is hardly surprising. The two are looking at different parts of the elephant.

And what does the video game industry look for in its technology hires? As much as anything, video game companies are in the market for people with strong programming skills. At the 2007 conference on Innovation and Technology in Computer Science Education (ITiCSE) in Dundee, Scotland, keynote speaker Chris van der Kuyl, Scotland's leading entrepreneur in the video game industry, assured his audience that the greatest single factor limiting growth in his sector is a shortage of programming talent.

That any segment of the industry might be starved for programming talent will likely come as a surprise to someone who sees programming as a soon-to-be-obsolete skill. "Programming? Who programs?" Andriole asks, with rhetorical flourish. The answer, of course, is that millions of people around the world are productively engaged in precisely that activity.

Contrary to the impression Andriole creates in his column, there is no evidence that the demand for highly skilled software developers is declining. The agencies charged with predicting employment trends expect a substantial increase in employment for people with software development skills. The Bureau of Labor Statistics, in its December 2007 report *Employment Projections: 2006–16*, identifies "network systems and data communications analyst" as the single most rapidly growing occupational category over the next decade, with "computer software engineers, applications" in fourth place on that same survey. This data is hardly suggestive of a job category in decline.

Employment projections are by no means the only evidence of continued demand for people with software development skills. Business leaders from the top software companies routinely cite the shortage of technical expertise as the biggest stumbling block they face. Consider, for example, the following remarks by Microsoft chairman Bill Gates in a February 19, 2008 op-ed article for the *San Jose Mercury News*: "Today, there simply aren't enough people with the right skills to fill the growing demand for computer scientists and computer engineers. This is a critical problem because technology holds the key to progress, and to addressing many of the world's most pressing problems, including health care, education, global inequality and climate change." Other industry leaders—including Rick Rashid at Microsoft (see his column in this issue) and Google founders Larry Page and Sergey Brin—have raised similar concerns.

It is clear from such responses that not everyone in the computing industry shares Andriole's conviction that traditional software-development skills are no longer relevant. Even so, industry leaders across all sectors nonetheless have something in common: they cannot find enough people with the skills they seek. Faced with a shortfall in the hiring pipeline, it is perhaps natural to argue that educational institutions should stop wasting time on other aspects of the discipline and focus of the skills that are just right for one particular environment. That argument would have merit if there were an imbalance between supply and demand, with too many degree recipients trained for some occupations while other jobs went begging. That situation, however, does not exist in the computing industry today. There is a shortfall across the board, with not enough graduates to supply any of the major subdisciplines.

The most powerful illustration I have seen documenting the magnitude of this shortfall comes from a talk presented by John Sargent, Senior Policy Analyst for the Department of Commerce, at a February 2004 research conference sponsored by the Computing Research Association (CRA). The figure here combines the data from several of Sargent's slides into a single graphic that plots statistics on degree production against the anticipated annual demand for people with those degrees. As you can see from the leftmost set of bars, the projected annual number of job openings for engineers is approximately two-thirds the number of bachelor's degrees produced each year. The situation in the physical sciences is similar at a somewhat smaller scale. In biology, by contrast, the annual number of job openings is only about 10% of the number of bachelor's degrees. This situation suggests an oversupply that allows for increased selectivity on the part of employers, who are unlikely to hire biologists without advanced degrees.

The bar graph for computer science at the right of the figure, however, reveals an entirely different situation. According to projections from the Bureau of Labor Statistics, the number of job openings for computer science exceeds the number of people receiving bachelor's degrees by almost a factor of four. Even if the industry were to hire every computer science graduate it would still have to look elsewhere for most of its new hires. That, indeed, is precisely what is happening. According to data presented by Caroline Wardle of the National Science Foundation at the CRA Snowbird conference in 2002, less than 40% of employees in computing-related jobs have computing degrees—a figure that stands in dramatic contrast to most other disciplines in which a degree in the field is part of the entry requirements. It is not that employers *prefer* candidates without formal training, but simply that there are nowhere near enough qualified graduates to satisfy the demand.

The problem that we face in computing education, therefore, is to increase the number of students. We cannot do that by arguing that only certain computing fields are worthy. The shortfall exists across the entire field. We need more students in each of the disciplines identified by the Joint ACM/IEEE-CS Task Force on Computing Curricula: computer science, computer engineering, software engineering, information systems, and information technology. Andriole would have us abandon software engineering, despite the fact that *Money* magazine recently put "software engineer" in first place in a list of the best jobs in the U.S. and despite the fact that the Bureau of Labor Statistics identifies "software engineer, applications" as one of the fastest-growing job categories.

Unfortunately, one of the biggest challenges that the ACM faces in its efforts to increase student interest in computing careers is precisely to counter the mythology about the dangers of offshoring that Andriole perpetuates in his column. His assertion that "programming will ultimately…be generated by relatively few professionals" largely located in places like Bangalore, Moscow, and Shanghai validates the fears so many high-school students express that computing careers will vanish as software development moves overseas.

The 2006 ACM report on *Globalization and Offshoring of Software*—a report to which Andriole contributed—finds no evidence to support this view. If anything, the opening of the offshore labor market in computing seems to have increased the number of computing jobs in the U.S., as illustrated by the following paragraph from the Executive Summary: "The economic theory of comparative advantage argues that if countries specialize in areas where they have a comparative advantage and they freely trade goods and services over the long run, all nations involved will gain greater wealth....This theory is supported to some extent by data from the U.S. Bureau of Labor Statistics (BLS). According to BLS reports, despite a significant increase in offshoring over the past five years, more IT jobs are available today in the U.S. than at the height of the dot-com boom. Moreover, IT jobs are predicted to be among the fastest-growing occupations over the next decade."

The reality is that the shortage of people with the expertise industry needs is so severe that companies will go anywhere in the world that can provide workers with the necessary skills. If those people exist in Bangalore, Moscow, or Shanghai, then companies will hire them there. And if those people exist in the U.S., those same companies will hire them here.

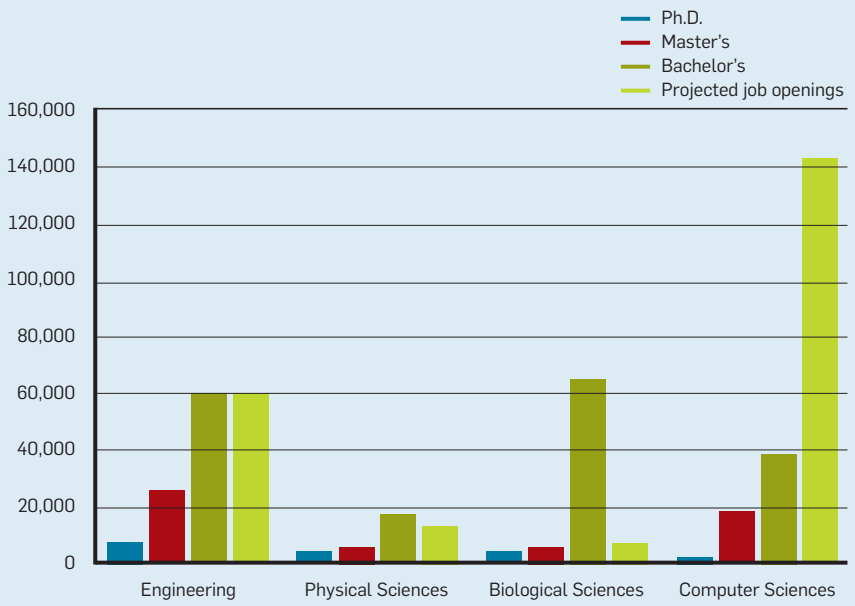
Unfortunately, all too many people seem to believe that companies always seek to minimize labor costs, typically by employing workers at the lower salaries that prevail in developing countries. That view, however, represents a fundamental misunderstanding of labor economics. Companies are not primarily concerned with minimizing costs; after all, they could accomplish that goal by shutting down. Companies are in the business of maximizing return.

A simple thought experiment will make this difference clear. Suppose you are Microsoft and are looking to hire people with stellar software development skills. One of the candidates you are considering is a recent graduate from a top-notch Silicon Valley university; given current salaries in the U.S., the cost of hiring this candidate might be, considering benefits and structural costs, approximately $200,000 a year. You have another candidate in Bangalore who will cost you only $75,000. Both candidates seem extraordinarily well qualified and show every sign of being extremely productive software engineers, capable of generating perhaps $1,000,000 in annual revenue. What do you do?

The answer, of course, is that Microsoft hires them both. Although the software engineer in Bangalore might be more cost-effective, what possible reason could there be for throwing away $800,000 a year? As long as qualified candidates are scarce and capital is plentiful, companies will hire anyone for whom the marginal value exceeds the marginal cost. The value that a company can recognize from the services of talented software developers vastly exceeds their costs, irrespective of in which country they reside or in what currency they are paid.

The only way software development jobs will move entirely overseas is if the U.S. abandons the playing field by failing to produce students with the necessary skills. As the *New York Times* editorial page observed on March 1, 2006, shortly after the publication of the ACM globalization report: "Perhaps that explains what the report says is declining interest in computer science among American college students. Students may think, Why bother if all the jobs are in India? But the computer sector is booming, while the number of students interested in going into the field is falling. The industry isn't gone, but it will be if we don't start generating the necessary dynamic work force." Andriole's failure to understand that the computing industry extends far beyond enterprise software and his perpetuation of the myths that drive students away can only make it more difficult to generate the dynamic work force the U.S. needs to remain competitive in the global marketplace. C

U.S. degree production and annual employment projections.



Source: Adapted from a presentation by John Sargent, Senior Policy Analyst, Department of Commerce, at the CRA Computing Research Summit, February 23, 2004. Original sources listed as National Science Foundation/Division of Science Resources Statistics; degree data from Department of Education/National Center for Education Statistics: Integrated Postsecondary Education Data System Completions Survey; and NSF/SRS; Survey of Earned Doctorates; and Projected Annual Average Job Openings derived from Department of Commerce (Office of Technology Policy) analysis of Bureau of Labor Statistics 2002–2012 projections.
See www.cra.org/govaffairs/content.php?cid=22.

Even though the statistics in the figure are derived from surveys taken several years ago, there is no reason to believe the situation has changed in any qualitative way. Comparing the 2002 and 2006 reports from the Bureau of Labor Statistics suggests that employment demand may have shifted by as much as 10% percent in certain categories. The fundamental message of the figure would not change even if the numbers were off by a factor of two.

Eric Roberts (eroberts@cs.stanford.edu) is a professor of computer science at Stanford University, co-chair and principal author of the computer science volume produced by the joint ACM/IEEE-CS Task Force on Computing Curricula 2001, and past chair of the ACM Education Board.