Are Too Many Programmers Too Narrowly Trained?

David Clark

Increasing commercialization, rapidly changing technologies, shorter deadlines, the Internet, and other factors have radically changed the software industry in the last few years.

This has caused a number of concerns in the software community. One of the most important concerns is whether this fast-moving, quickly changing world has led to such a demand for programmers, that employers are hiring many programmers who are too narrowly trained. The issue is whether these programmers know the specific languages and techniques in demand at the moment but don't know solid engineering fundamentals.

Many high-profile computer-related problems, such as Y2K and system crashes, can be traced to narrowly trained programmers, according to Norman Matloff, professor of computer science at the University of California, Davis.

The use of programmers with narrow training can lead to buggy or broken applications, as well as expensive delays of product releases, Matloff said. Moreover, he said, narrowly trained programmers may lose their jobs when their skill sets are no longer in demand and then have trouble finding new positions because they lack the education in software-engineering fundamentals to learn new skills quickly.

Some industry observers express concern that schools are training programmers too narrowly.

However, Eric Roberts, professor of computer science at Stanford University, said that many schools are educating programmers well but that companies don't always hire well-trained programmers. Companies are having so much trouble finding enough qualified programmers to fill job openings, he said, they don't always hire broadly skilled people. "I worry that this is a cause of [skills-focused] problems," he said.

The issue of narrowly trained programmers will become increasingly important as employers continue to struggle trying to meet the demand for programmers. Figure 1 shows US Bureau of Labor Statistics projections that between 1996 and 2006, domestic employers alone will have to find 306,000 new programmers.

THE IDEAL

Many computer science educators agree that software engineers should learn engineering principles. This is different from learning specific operational activities, such as how to program in Java or C++, noted software consultant Roger S. Pressman of R. S. Pressman & Associates (http://www.rspa.com), a software consultancy.

A broad, engineering-based education is important because programming, as well as other elements of computer technology, has become very complex, explained Ellen Ullman, an independent software developer and author. Programmers now must know much more than just a few languages.

Meanwhile, software engineers must also keep up with the latest technology and techniques on their own, Ullman said. The willingness and ability to do that are two things a student gets from a broad education, she said.

Ed Lazowska, chair of the University of Washington's Computer Science and Engineering Department, agrees that an emphasis on fundamentals is required. "From my point of view, this is a field, more than any other, that requires a broad base of education and the capability and will for lifelong learning because today's skills are going to be completely useless three years down the road."

TREND TOWARD NARROW TRAINING

Many observers say a number of factors in both the software industry and academia have helped drive the trend toward programmers with narrow training.

Driving factors: industry

Some critics say that software development companies, in their rush to profitability, are interested only in hiring people with the specific programming skills, such as Java or Active Server Page scripting, that they need at the time.

Fast-paced startups frequently don't have time to develop their employees' skills and focus only on the skills, not the broad training, that programmers already have, said Portal Software's chief technology officer, John Boring.

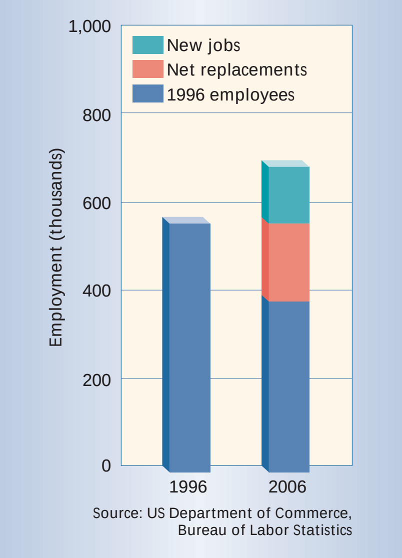
UC Davis' Matloff said many companies focus on hiring programmers with specific skills in two ways. First, he said, some employers use skills-based job descriptions to attract programmers with relatively little training and experience, who will work for lower salaries. Second, he added, many companies sincerely believe specific skills are so important to their operations, they need to focus only on hiring programmers with those skills.

The way companies handle the blizzard of job applications they receive also contributes to skills-based hiring. "It's all about matching buzzwords on resumes to the job requirement," said Boring, who has recruited for Apple Computer, Netscape Communications, and other companies. Human resources staffs don't have time to carefully research potential employees, he said, and the use by many companies of resume-scanning software makes the problem worse.

However, narrow training is actually adequate for some specialized jobs, said Ullman. As the number and type of computer-related jobs have grown over the years, a number of software jobs, including some related to Web site development and maintenance, have become very narrow in scope, she explained.

Many high-profile computer-related problems can be traced to narrowly trained programmers.

Norman Matloff, Professor of computer science, University of California, Davis



Source: US Department of Commerce, Bureau of Labor Statistics

Figure 1. The US Department of Commerce's Bureau of Labor Statistics projects that domestic employers will need 697,000 programmers by 2006, 129,000 more than in 1996. In addition, the bureau predicts that of the 568,000 programmers working in the US in 1996, 177,000 will leave the profession within the next decade and have to be replaced. This means US employers must find 306,000 new programmers by 2006.

Driving factors: education

When employers overemphasize skills, some students get the impression that skills are all that count, Matloff said. So while schools may offer a variety of courses, some students may choose to focus on learning specific programming skills.

Pressman said students want to learn such skills as Java, COM, and CORBA because they're hot. "There are very few students that will request [a class in] partial differential equations," he said.

Many training consultants and industry observers say there are also problems with many schools' current approaches to programmer education.

"There are serious problems with the way software developers are being trained and educated today, but I wouldn't say it's too narrow or too broad," said Larry Constantine, professor at the School of Computing Sciences, University of Technology, Sydney, Australia. It's just focused on the wrong things, such as the characteristics of specific languages or working environments or tools, rather than the fundamentals of what is ultimately an engineering discipline, he said.

Alan Davis, CEO and founder of OmniVista, a software consultancy, added that many programming students also are not extensively trained in any important aspects of the software business other than coding.

Les Hatton, founder of Oakwood Computing (http://www.oakcomp.co.uk), a software consultancy, and a professor of software reliability at the University of Kent, UK, agreed with Davis. "We don't generally teach testing, risk management, reliability theory, control process, feedback concepts, and conservative engineering," he said.

Many universities focus on training students primarily so that they can get jobs in the software industry, which generally doesn't have a clue about activities like testing and risk management, he said.

Meanwhile, Davis said, many software developers are being trained by teachers who never developed software themselves. He said most university professors cannot train programmers in modern software practices.

This is not the case at Stanford and probably is not true for most schools with a high-level software engineering faculty, said Roberts, himself a former software developer. This might be the case at many other schools, though, he noted. Good software teachers are in short supply to universities, just like good programmers are in short supply to companies, he explained.

Roberts said the situation is improving a little in university computer science departments. Nonetheless, many people hired for software engineering and systems-building jobs come from academic disciplines other than computer science.

In fact, Matloff said, as in the past, only one of every four new programmers

Learning software engineering principles is different from learning specific programming languages.

Roger S. Pressman,
Software consultant,
R.S. Pressman & Associates

receives a computer science degree. Therefore, he said, even if computer science departments radically improve their curricula, it would affect only one-fourth of new programmers.

PROBLEMS OF NARROW TRAINING

At a minimum, Pressman said, the software produced by companies that primarily hire narrowly trained programmers frequently is produced inefficiently. At worst, he said, the software doesn't work well.

Meanwhile, many narrowly trained programmers will not have the background in fundamental software-engineering principles to learn the new skills that the marketplace will demand when employers no longer want the skills they currently have. And it is unlikely another Y2K problem will come along to create jobs for programmers with skills in Cobol and other older languages.

Meanwhile, narrow training could keep the industry from finding new ways to think about computers, in order to deal most effectively with such developments and opportunities as the Internet, Ullman said.

"It's time for people to start thinking in other ways about how we code. Our languages are pretty old," she said.

The industry cannot come up with new ways to look at computers without a broad point of view that provides an abstract knowledge of computer science, supported by other disciplines, she said.

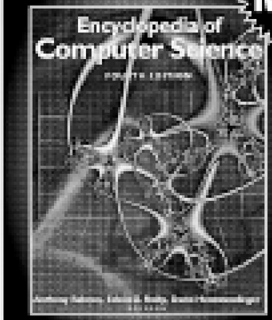
PEERING INTO THE FUTURE

To improve programmer quality, educators and consultants often suggest that the software industry adopt a model of mandatory licensing and certification, as is used in the medical and legal professions. This could establish base skill-set and competency levels, and help schools standardize their curricula.

Proponents acknowledge that it could be difficult and time-consuming for software companies and others to agree on the standards that would be used in a licensing and certification process.

The University of Technology's Constantine, who advocates licensing and certification, said, "If things had collapsed significantly after Y2K I think there would have been a hue and cry for standards to define the profession." Now, without a crisis, he expresses little hope this will occur in the near future.

Omni-Vista's Davis advocated a new, multidimensional training approach for programmers. For example, he said, software schools should be more like medical schools, with the best practicing software developers as teachers. In addition, he said, the curricula should offer large doses



of fundamentals in the first two years and then a year or two of internship.

This would expose students to the workplace, train them in programming basics, and show how their work should relate to an organization's business goals.

According to the University of Washington's Lazowska, schools might make some of these changes if the software industry clearly said it wants programmers with a strong, broad undergraduate education.

Some software companies are no longer looking for programmers based solely on specific skill sets. This is the case with Microsoft, said Megan Morreale, a recruiting director with the company. "We're having to focus more on generic problem-solving skills these days because we're not finding the specific skills we're looking for." Part of the problem, she said, is that students and veteran programmers cannot keep current with the range and number of emerging new software technologies.

In the near future, though, many other companies may react to the rapid changes in software technology by continuing to hire programmers with a specific set of high-demand skills as quickly as possible.

Some suggest the software industry adopt licensing and certification for programmers.

In fact, said Davis, "I think it's going to get worse because the rate at which software paradigms are changing is accelerating to the point that methods now change in 18-month time frames," said Davis.

According to Constantine, perhaps the most likely motivation for vendors to stop hiring narrowly trained programmers is that this would improve product quality.

He said, "The best we can hope for is that leading companies will become aware of the fact that users are becoming less and less tolerant of user interfaces that are difficult or impossible to use and of software that falls apart many times a day." ✷

David Clark is a freelance technology writer based in Torrance, California. Contact him at dwclark@earthlink.net.

Editor: Lee Garber, *Computer*, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1314; l.garber@computer.org