ICER White Paper

Eric Roberts January 1, 2006

Over the past couple of years, concerns about the state of computer science education have been very much on my mind. The challenges of declining enrollments and of teaching effective introductory courses with the complex tools of today have been central to my work with the ACM Education Board, the Computer Science Teachers Association, and the Java Task Force. The more I look at the situation, the more convinced I have become that the problems we face are complex and wide-ranging, in the sense that they involve factors far outside the domain of curriculum and pedagogy for which we are responsible as educators. While there are things that we can do, it would be a mistake to imagine that the problems stem from a single source or that they can be easily solved.

Even though current statistics are difficult to obtain because of delays in data gathering, I think there can be no question that the decline in enrollments that we understand from our individual experiences and interactions with colleagues elsewhere represents a real phenomenon. The estimates I have seen suggest that enrollments in computer science majors have fallen between 40 and 50 percent in the last five years, despite what appears to be continuing strong demand from industry for graduates with these skills. That decline represents a very real problem not only for academic departments that must cope with the effects of a "boom and bust" cycle of enrollments, but also for the overall health of the field and the competitiveness of the U.S. economy. I believe it is essential for our field to understand the reasons for this decline and to develop initiatives to reverse it. Some of those initiatives will center on academic institutions. Given the scope of the problems and the economic impact that it is likely to have, those initiatives will likely require intervention at the level of state and federal policy if they are to have any chance to succeed.

Unfortunately, I am concerned that much of the discussion to date tends to oversimplify the problem to such an extent that the proposed solutions must of necessity fall short. As I expect most of you have, I have been involved in many discussions about why this decline has occurred and what to do about it. In most of those discussions, people identify a list of factors that include the "nerd" image of the computer scientist, the perception of computer science as an antisocial environment in which one is tied to a computer screen, the sorry state of high-school computer science in most parts of the country, and the identification of computer science with programming. There is, of course, a great deal of truth behind each of these concerns. As an explanation for the downturn in interest among students, however, these specific arguments just don't wash for the simple reason that all of these concerns were just as applicable when computer science was so popular we couldn't keep students away. The primary reason for the recent downturn must therefore lie somewhere else. If we are seeking to identify the real causes, we must focus on factors that have changed since the peak of computer science's popularity at the turn of the millennium. Those changes could be in the nature and perception of computer science as a field, in the culture from which prospective students emerge, or in some combination of these two.

If you look at things from this perspective, the set of problems looks somewhat different. To me, the following factors offer the best explanation for the enrollment decline:

1. There is a widespread perception among prospective students that the opportunities once available in computing have disappeared, both because of the decline in the high-tech economy after the dot-com meltdown and because of the increasing offshoring of technical jobs. In some ways, this perception flies in the face of evidence, given that the economic data show continued high employment in the field. If, however, students were really attracted by the dream of instant riches, the opportunities are indeed fewer.
2. Continuing a longstanding trend that began well before the downturn, students at both the high-school and university level regard education in a more instrumental light, focusing less on its intrinsic value and more on the postgraduate options it affords. To

the extent that this change in perspective holds, we may have less flexibility to attract students by changing our curriculum, since they will be guided more strongly by the perception of downstream opportunities.

3. Computer science curricula have changed in the last decade to focus on languages (Java) and paradigms (object-oriented programming) that are difficult both to teach and to learn. This problem is particularly severe at the high-school level where many teachers are not adequately prepared.
4. Institutions responded to enrollment growth in the 1990s by making introductory CS courses into "filters." While that filtering had only a modest effect when the high-tech economy was booming, it has been much more "successful" in recent years.

In the face of these systemic factors, I am skeptical of most of the proposals I've heard that offer some radical curricular change as a miracle cure for the ills of the discipline. Many voices in the last few years have resurrected the idea of the breadth-first curriculum that was all the rage among educators in 1990. Students didn't like it then and are unlikely to be attracted to it now. Despite a strong endorsement from the *Curriculum 1991* report, those of us involved in the follow-on *Curriculum 2001* report could find no extant curricula that continued to use that approach. The idea, moreover, of *any* radical change in high-school curricula at this point in history strikes me as dangerous in the extreme. After the instability generated by the ill-advised upheavals in the Advanced Placement exam in recent years, any attempt to force major changes at the high-school level would not only fail to attract more students, but would in fact drive away the few teachers that remain.

All this is not to say that we shouldn't try to improve our curriculum. Of course we should. The effort that so many people have put into creating the CC200x reports are intended precisely along these lines. At the same time, we should have no illusions that (1) the decline in enrollments has anything to do with our curriculum or (2) that changes in the curriculum will bring back the hordes.

The question then is what can we do. I believe that the most important initiatives for the future of computer science education lie in the following areas:

- We need to identify new value propositions for computer science beyond the instant riches model. That image came to characterize far too much of our discipline, and we have not yet grown beyond that mythology. If we seek to restore the idea that computing has an elegance and fascination of its own, we will have smaller classes but perhaps more interesting ones.
- As a particular instance of the search for new ways of defining the value of computer science, we must begin to forge more interdisciplinary connections to emphasize the importance of computing to such diverse fields as biology, economics, or media studies. When our courses were oversubscribed, we couldn't afford to build these bridges; today, we can't afford not to.
- We absolutely have to fix the problems of complexity and instability that have made it nearly impossible to teach at the introductory level. While some have called for a return to simpler languages, I believe that the only feasible approach is to develop common strategies and tools that allow modern programming systems to be used effectively in the educational domain, as exemplified by the work of the Java Task Force.
- We need to eliminate those vestiges of the "filter course" model that came into existence during the boom years. In many institutions, it will be necessary to scale back the expectations we have of all students while continuing to encourage individual students who choose to go beyond those bounds.
- We must continue to take affirmative steps to broaden the audience for computing. Particularly given the negative mythologies that today permeate the culture, we have to find ways to attract and retain students to this vitally important field.