High Tech/High Touch


John Naisbitt's Megatrends: Ten New Directions Transforming Our Lives was first published in 1982. It was a best seller and has won considerable acclaim. The second chapter of the book is titled "From Forced Technology to High Tech/ High Touch." In that chapter Naisbitt suggests "that whenever new technology is introduced into society, there must be a counterbalancing human response—that is, high touch—or the technology is rejected."

Naisbitt's high tech/high touch paradigm has interesting implications for computer education. Consider two scales, one labeled "tech" and the other labeled "touch," each running from low to high. The paradigm supports a conjecture that a person lies at some point on the "tech" scale and some point on the "touch" scale. Whatever a person's placements on these two scales, they represent a harmony or balance in their tech/touch.

The introduction of increased technology into a person's life produces an imbalance. For a person whose "tech" placement is high, additional technology represents only a modest percentage change and perhaps requires relatively little adjustment of "touch" to maintain a balance. But for a person placed low on the "tech" scale, even a modest amount of new technology may require a considerable adjustment to "touch."

High tech/high touch is a simple-minded paradigm, perhaps most useful for provoking discussion rather than providing a foundation to support educational change. But let's explore the paradigm a little more. We might guess that early adopters of computers were high-tech people. (At the same time they might be at any spot on the "touch" scale). Such high-tech people found it easy to adjust to computer technology and are now well established as computer leaders and teachers.

But as we attempt to introduce more and more people to computers, we soon move beyond the readily available supply of high-tech people who might be interested in computers. We begin to experience increased resistance as we attempt to introduce high-touch people to computer technology. Moreover, we have the added difficulty that the current computer leaders and teachers have a high-tech orientation, while the people they are attempting to teach have a high-touch orientation. These differences in orientation make effective communication difficult!

In recent years I have grown to understand some of the differences between high-tech and high-touch people. On a "touch" scale I have moved in the direction of higher touch. (I doubt if I have reached the midpoint yet, since I started so close to the low-touch end. But I am pleased with the progress I have made.)

Gradually, over the past eight years, I have experimented with increased use of high-touch ideas and activities in the computer education workshops I present. In recent years I have grown in ability to teach and make use of active listening, guided fantasy, small group discussion, large group interaction and other high-touch techniques. These ideas, and others, are included in my
Computer Education Leadership Development Workshop. The workshop even includes a substantial session on Stress and Burnout.

Another session in the workshop examines similarities and differences between mathematics education and computer science education. I view mathematics as a high-tech discipline—as the queen of the sciences—even though it differs from other science disciplines and their related technologies. We know that our mathematics education produces math anxiety and an "I can't do mathematics" syndrome among many people. Do we want the same results in computer science education?

Our mathematics education system is predicated upon two major assumptions. First, all people need to be able to do mathematics at a moderate level in order to survive in our society. Second, our society needs a number of professional mathematicians and other people who can function at a relatively high level in mathematics.

Thus, formal instruction in mathematics begins in the first grade or earlier, and a spiral curriculum approach is used in subsequent grades to ensure that almost all students develop a moderate level of mathematical knowledge. Beginning roughly at the junior high school level, our mathematics education system begins a process of separating off students who display good mathematical talent and learning ability. Others are discouraged by the system. They learn that they can't do mathematics as well as some of their colleagues and teachers; they feel insecure in their mathematical knowledge and perhaps get poor grades.

Early efforts to introduce computers into elementary and secondary school education tended to follow the mathematics education paradigm. That is not surprising, since much of this early teaching was done by math-oriented early adopters of computer technology. Moreover, there was considerable rationale to this approach, since computer programming and the underlying computer science seemed to be necessary in order for a person to use a computer.

But now we are moving beyond the early adoption stage. Many elementary schools, for example, are moving toward involving all of their teachers and all of their students in working with Logo. Some are developing a spiral curriculum scope and sequence that has many characteristics of a mathematics scope and sequence. It is my guess that this approach will soon produce junior high school computer-anxious students who assert, "I can't do computers."

The mathematics paradigm for an elementary school computer curriculum is not the only possible paradigm and it may not be the most appropriate one. Progress in computer software and hardware has made it possible for people to become effective users of computers without knowledge of the underlying computer programming and computer science. A "survival" level of computer-use skill is easily obtained without learning how to write programs. A spiral curriculum of computer science instruction need not begin in the first grade to develop high school graduates with a survival level of computer science knowledge. Nor do we have evidence that the supply of computer science graduate students will be diminished if computer programming is less emphasized at the precollege level.

This type of analysis suggests that we might look for other, more appropriate paradigms for computer education, especially at the lower grade levels. Perhaps art education provides a more appropriate paradigm? Art education tends to be quite high touch. Students explore the art media; they frequently set their own goals; they evaluate their own work and the work of others. Some elementary schools have taken the approach that Logo should be introduced using the art
education paradigm. Naisbitt's high tech/high touch ideas suggest that this approach will be more successful than approaching Logo using a mathematics education paradigm. I have talked with several elementary school teachers who have used this approach and feel that it is very successful.

The high tech/high touch paradigm can be used to examine other aspects of computer education. In my Computer Education Leadership Workshop I often ask participants to rank a set of qualifications essential to being a successful computer coordinator. I have now used this activity in a half dozen workshops. In every workshop the participants listed "Interpersonal and Communication Skills" as most important and "Technical Skills" as least important among the four general qualifications being rated. These workshop participants, many of whom are successful computer coordinators, are suggesting that high touch is more important than high tech.

My conclusion is that the high tech/high touch paradigm provides a useful approach to examining many aspects of computer education. I am sure you can think of your own examples and issues-such as whether extensive use of CAI will damage social development and skills. I'd like to hear from you about your examples.