DHQ: Digital Humanities Quarterly
2013
Volume 7 Number 2

A review of Nathan Ensmenger, *The Computer Boys Take Over: Computers, Programmers, and the Politics of Technical Expertise* (Cambridge, MA, and London: MIT Press, 2010)

Trisha Campbell <TNC17_at_pitt_dot_edu>, University of Pittsburgh

Abstract

This is a review of Nathan Ensmenger's *The Computer Boys Take Over: Computers, Programmers, and the Politics of Technical Expertise* (Cambridge, MA, and London: MIT Press, 2010).

Last summer, I sat for a Rails Girls Workshop. The room was small, the chairs multi-colored assortments ramming against rows of Macbooks. We all got T-shirts with digital hearts on them. Looking around the room, I noted more colors of hair than colors of chairs, from purple to pink, to none at all. These women, I thought, are the future of the computer programmer. But this kind of programmer, the funky girl programmer, is a relatively new innovation, stemming from a fraught history of the "computer boy," as Nathan Ensmenger puts it in his part computational, part sociological study of the invention of the "computer boys," or today's "IT guys." Ensmenger's work explores how the social and technological developments of the twentieth century were made to happen through people, not impersonal processes, to socially re-construct the "computer revolution" as proffered via the computer boys. He offers a fresh history of the emergence of "computer specialists" as they began and sought to make an identity of their specialty through professionalization and organization into an agreed upon study. He does this by capturing the tension between what began as the craft of computer programming, then thought to be an art, and the growing "scientization" of programming in the 1970s. Tensions emerged between the craft-centered practices of vocational programmers (i.e. programming is poetry), the increasingly theoretical agenda of academic computer science, and the desire of corporate managers to control and routinize (i.e. what we know today as computer science) the process of software development. Eventually, amidst the tensions, a decision was made in the late 1960s to make the field more masculine.

Figure 1.

Ensmenger's work moves the focus off the technology and technological determinism of software studies, away from what has been falsely labeled the "computer revolution," which implies a deterministic phenomenon brought about by the computer. Instead, he puts the emphasis on the people — "the computer boys" — building the software and the rhetoric surrounding job demands, aptitude tests, and images of the programmer. His text enters into a line of historical work on technology attempting to re-make a history that includes people (those seen and unseen), politics, economics, social elements and rhetoric.

The Computer Girls

BY LOIS MANDEL

A trainee gets $8,000 a year
...a girl "senior systems analyst"
gets $20,000—and up!
Maybe it's time to investigate....

Ann Richardson, IBM systems engineer, designs a bridge via computer. Above (left) she checks her facts with fellow systems engineer, Marvin V. Fuchs. Right, she feeds facts into the computer. Below, Ann demonstrates on a viewing screen how her facts designed the bridge, and makes changes with a "light pen."

Twenty years ago, a girl could be a secretary, a school teacher . . . maybe a librarian, a social worker or a nurse. If she was really ambitious, she could go into the professions and compete with men . . . usually working harder and longer to earn less pay for the same job.

Now have come the big, dazzling computers—and a whole new kind of work for women: programming. Telling the miracle machines what to do and how to do it. Anything from predicting the weather to sending out billing notices from the local department store.

And if it doesn't sound like woman's work—well, it just is.

("I had this idea I'd be standing at a big machine and pressing buttons all day long," says a girl who programs for a Los Angeles bank. I couldn't have been further off the track. I figure out how the computer can solve a problem, and then instruct the machine to do it."

"It's just like planning a dinner," explains Dr. Grace Hopper, now a staff scientist in systems programming for Univac. (She helped develop the first electronic digital computer, the Eniac, in 1946.) "You have to plan ahead and schedule everything so it's ready when you need it. Programming requires patience and the ability to handle detail. Women are 'naturals' at computer programming."

What she's talking about is aptitude—the one most important quality a girl needs to become a programmer. She also needs a keen, logical mind. And if that throws out the old Billie Burke-Gracie Allen image of femininity, it's about time, because this is the age of the Computer Girls. There are twenty thousand of them in the United (cont. on page 54)

Figure 2.

As Ensmenger works through the “fault lines” in software development, he reveals how the “points of tension between groups or individuals, differing perceptions of reality or visions for the future, and subtle hierarchies and structures of power relationships” come out most notably in the disappearance of women from what is now the field of computer science/programming [Ensmenger 2010, 11]. This fault line in the history is a crucial underlying crack in the book and is implied, I think purposefully, by the moniker “the computer boys.” 3

Ultimately, Ensmenger wants his readers not only to understand that software is socially constructed but also to imagine that the software crises pervading the industry now, if seen through the lens of social construction, can help explain why “in an industry characterized by rapid change and innovation, the rhetoric of crisis has proven so remarkably persistent” [Ensmenger 2010, 243]. Along with this, he proposes three solutions to the software crisis brandished upon his history and relies on an argument for heterogeneity. This heterogeneity, though, is only complicated by what it means to invent again: “design in a heterogeneous environment is difficult… The hardest single part of building a software system is deciding precisely what to build” [Ensmenger 2010, 241]. 4

And therein lies my only problem with the book, and it is my same problem with many academic books; the thing is socially constructed and the pages of the text helped us get there, but *how* do we build from this heterogeneity. Is there a method? Perhaps this will be Ensmenger’s second book. Either way, I greatly enjoyed Ensmenger’s charming telling of *Cosmopolitan*s aptitude surveys for women or the silly job ads from RCA. He does an interesting history of software, of the computer boys, of the missing computer girls (as *Cosmo* dubbed them), and he prepares the way for those of us in the fit of this software crisis to re-think what we build, at the very level of invention. 5

The book opens with Chapter 1, “Introduction: Computer Revolutionaries,” in which Ensmenger presents a term he does 6

a lot with, both in the book and on his website (http://thecomputerboys.com/): *the computer people*. Computer people, for Ensmenger, are those people neglected by history and discourse in the chatter about the computer. This is in part, he says, the "conventions of the genre," but it has to do with the traditional bias of the computer and the history of the *computer,* rather than the history of the computer people. Ensmenger uses the figure of "computer people" to trace a history of software through the computer boys as they sought to make a role for themselves and to get at the invention of the computer user: "Historians have long suggested that technological innovators, including the designers of electronic computers, also invent the kind of people they expect to use their innovations" [Ensmenger 2010, 13].

7 In Chapter 2, "The Black Art of Programming," Ensmenger continues his peopled computer history with the story of how programming became magical black art. He maps the origin of computer programming from the 1950s through the 1960s, particularly 1968, when it was decided at a NATO Conference on Software Engineering that programming had come to be perceived as "too artistic." This was an attempt to steer programming into something more scientific and masculine. This chapter is full of interesting old ads from RCA and stories of agents flowing in and out of how we perceive of programming today.

8 Chapter 3, "Chess Players, Music Lovers, and Mathematicians" and Chapter 4, "Tower of Babel," discuss the aptitude testing rampant in the beginning of the computer revolution. The search for qualified people and "clever fellows" began with certain quirky criteria set out for what might make a good programmer. Programming at this time was thought to be an innate quality, where details, task-orientation, and anti-social tendencies seemed to equal a good programmer. Gender again shows up, because even while *Cosmopolitan* put out aptitude tests for their female readers, the questions were slanted toward an antisocial, mathematically inclined male.

9 From this masculinization of a profession and group of people came the disciplinization of modern computing, where computer programming became a computer *science.* Ensmenger explores the rise and "scientization" of the field in Chapter 5, "The Rise of Computer Science," through the messy compromises involving what computing should or ought to look like.

10 In Chapter 6, "The Cosa Nostra of the Data Processing Industry" we meet Herbert Grosch, who called programmers the "Cosa Nostra" of the computer industry: "[programmers] are at once the most unmanageable and the most poorly managed specialisms in our society. Actors and artists pale by comparison. Only our mathematicians are as cantankerous, and it's a calamity that so many of them get recruited by simplistic personnel men" [Ensmenger 2010, 137]. This chapter marks the break between managers and programmers and gives way to the invisible computer technician.

11 Bit by bit and history by history, Ensmenger has re-composed the computer technician back into a body and personality. A body and personality, which he suggests, can be otherwise. In his final chapter, "Conclusion: Visible Technician," he offers what to do with all the people — boys, to be exact — that created our current digital world and software situation. He asks where all the "computer girls" went, never quite settling on an answer to this question. He blames professionalization and the construction of a gender identity tied to the construction of the programmer identity. Yet the question of women still looms and seems to serve as a backdrop for many of his questions; it is the invention of the "computer user" as made by the invention of the computer technician, that Ensmeger gives us, which has become a one-to-one loop that leaves the invention of the woman out of the process. This is a wonderfully complex position to begin thinking about where all the women have gone. That said, he does give three crucial lessons to be learned in his history as it relates to software, which I will only adumbrate to wet your palate: design is heterogeneous and software is always about more than code, more than technology, and more than the professional and managerial dictions; it does at its very most and its very least affect human beings amidst a network of other agents.

12 Nathan Ensmenger's book is an impressive and engrossing historical work. He brings the crises and the peopling of computer programming alive. His historical artifacts become characters and a full picture of what this heterogeneous history looks like emerges. He adeptly weaves together competing voices in history with competing personalities to leave us with this haunting and antagonizing last line, echoed from the rhetoric of programming as created by the history: "almost thirty years after the NATO Conference on Software Engineering many programmers are still concluding

that ‘excellent developers, like excellent musicians and artists, are born, not made’ ” [Ensmenger 2010, 243]. Let us finally see that they have been made and can be re-made, like the hair of the Rails Girls, in various shades.

Works Cited

Ensmenger 2010 Ensmenger, Nathan. *The Computer Boys Take Over: Computers, Programmers, and the Politics of Technical Expertise*. Cambridge: MIT Press, 2010.

This work is licensed under a Creative Commons Attribution-NoDerivatives 4.0 International License.