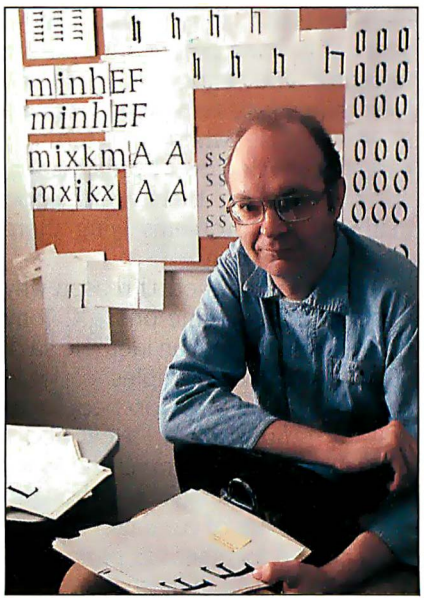
COMPUTER SCIENCE CONSIDERATIONS

CONDUCTED BY G. MICHAEL VOSE AND GREGG WILLIAMS

Donald Knuth speaks on his involvement with digital typography

Text processing as a computer science problem has consumed a major portion of the time and energy of Stanford professor Donald Knuth over the past eight years. Knuth authored and placed into the public domain a highly regarded typography system that he calls TEX *(pronounced "tech"), along with a font creation language called* METAFONT. *In conjunction with the completion of* TEX, *Knuth and Addison-Wesley are publishing a five-volume work entitled* Computers and Typesetting. *Volume 1 is* The TEXbook, *volume 2 is the source code for* TEX, *volume 3 is* The METAFONT Book, *volume 4 is the* METAFONT *source code, and volume 5 is* Computer Modern Typefaces.

To discover what so intrigued Knuth about this subject, BYTE *senior editors Gregg Williams and Mike Vose conducted the following interview with Professor Knuth at Addison-Wesley's offices in Reading, Massachusetts, on November 11, 1985.*



Donald Knuth

BYTE: *Dr. Knuth, how did you become involved with digital typography and the public-domain system known as* TEX?

Knuth: I got interested because I had written books and seen galley proofs, and suddenly computers were getting into the field of typesetting and the quality was going down.

Then I was working on a committee at Stanford planning an exam, and we got a hold of some drafts of Patrick Winston's book on artificial intelligence. We were looking at it to see if we should put it on the reading list for a comprehensive exam. It had just been brought in from Los Angeles where it had been done on a digital phototypesetter. This was the first time that I had ever seen digital type at high resolution. We had a cheap digital machine at Stanford that we thought of as a new toy. But never would I have associated it with printing a book that I'd be proud to own.

Then I saw this type, and it looked as good as any I had ever seen done with metal. I knew that it was done just with zeroes and ones. I knew that it was bits. I could never, in my mind, ever, conceive of doing anything with lenses or with lead, metallurgy, and things like that. But zeroes and ones was different. I felt that I understood zeroes and ones as well as anybody! All it involved was getting the right zeroes and ones in place and I would have a machine that would do the books and solve all the quality problems. And, also, I could do it once and for all. I still had a few more volumes to write *[of his seminal work,* The Art of Computer Programming, *a seven-volume series of which three volumes are finished]* and

(continued)

'I was excited that I started out trying to apply computer science to typography and wound up applying typography to computer science: in fact, right in the center of computer science.'

by the time I was ready with volume 7, the technology would change another three times and the quality would go down each time. So if I could only figure out a way to generate the right zeroes and ones, then I could have that in a computer program that I know how to write, and everything would be solved.

So within a week of seeing this example from Winston's book, I told my wife I had to start changing my present plans to work on typography. I was going to spend one year doing all this typography, and I was going to write a system that would be useful to do my books. At the end of the year I would go back to write those books the way I had been doing.

BYTE: *And what year was this?*
Knuth: That was 1977, '78. If I had estimated that it would take eight years, of course, I never would have started. I certainly didn't have any idea that this would be as difficult a problem as it turned out to be. It looked pretty easy to me at first.

BYTE: *So you embarked on this project mostly out of necessity—you needed a superior system for producing your books. Then, once you got into it, what captivated you about typography as a computer science problem that's held onto you for eight years?*
Knuth: I found that it was very rich. I found that there were a lot of things below the surface that were really interesting from both the theoretical and the practical points of view. For example, I needed to develop a lot of mathematics for rounding curves so that they looked right as raster images. At first, I didn't think that was going to be very hard. I didn't realize the importance of symmetry, how hard it is to make a left parenthesis look like a mirror image of a right parenthesis if you don't put the line exactly the same. If you have something that wants to be 2⅘ pixels wide and you put it down in one place it becomes 3 and in another place it becomes 2. All of the obvious approaches to visualization failed. I kept going on it because I felt that I was in the right place at the right time and was destined to do the job.

I knew that these were problems that took a pretty good mathematician to solve, and there weren't any other good mathematicians looking at it. So I felt that it was my duty, and it was also interesting. And partly because I felt that here I was with 40 years of training pertinent to this interesting and important problem. New things kept turning up because it was a case where the territory hadn't been gone over by mathematicians before, so there were good mathematical problems lying there just for the asking.

And there was another reason why I spent so much time on TEX. Tony Hoare came up with an idea. He said, "Don, we need examples of large computer programs for people to look at," and he said, "How about publishing your programs for TEX." That was mind-boggling. I thought, "I'm a professor of computer science and I hacked together this program in a big hurry trying to finish it in a year and now I'm supposed to publish it! Ouch—I have a reputation as a computer scientist. Nobody ever shows what you really do in computer programs, so this is out of the question. We tell students what they are supposed to do, but do we really have time to dot the i's and cross the t's when it comes down to it?" On the other hand, it seemed to me that this was kind of a ridiculous situation, for a professor of computer science to be ashamed of a program he had written. Could I really do something that would make a large program understandable? Could I write a program that was useful, accommodating the compromises of the real world, and still have something that I could say that I was proud of?

Then it occurred to me, I had one thing going for me that would make it easier: I had a typographic system, so I could use typography to help the documentation of my programs. So then I realized that there were lots of other ideas floating around that people had used that could all be brought together with typography in making a way of documenting programs so that a large program could be well understood.

This led to what is now called the WEB system, a new way to write programs. [*Editor's note*: *Knuth defines* WEB *as follows*: "WEB *is itself chiefly a combination of two other languages*: (1) *a document formatting language and* (2) *a programming language. My prototype* WEB *system uses* TEX *as the document formatting language and* Pascal *as the programming language, but the same principles would apply equally well if other languages were substituted.*" *Quoted from* "Literate Programming" *by* Donald E. Knuth, The Computer Journal, *vol.* 27, *no.* 2, 1984.]

BYTE: *So* WEB *as a programming paradigm grew out of a fusion of typography and structured programming?*
Knuth: It turned out that I got so excited about WEB that I wanted to go back and rewrite every program I had written since the 1950s. I felt that at last it was real programming. Of course, I'm too much of a fan of this [WEB] to be considered unbiased. I love the fact that once I got to be writing programs in this way it was a turn-on just because I felt that the program was being exposed the way a program should be, and I am an expositor at heart.

I was excited that I started out trying to apply computer science to typography and wound up applying

(continued)

'People just love to see something new that they can control and make words come out in a different way.'

typography to computer science: in fact, right in the center of computer science.

BYTE: *Right now one of the hottest topics in computers is desktop publishing, and there is a plethora of new programs out that do what* TEX *does, only not as well. Are programs like* TEX *going to fundamentally alter the way people work with words?*

Knuth: I think it will affect a lot of people. I'll just tell you what I know about this. Whenever something becomes a lot easier, when a person has the power to do something he couldn't do before, this affects his life. When something becomes 10 times cheaper than it was before, all of a sudden it becomes an option for somebody that they never would have thought of. In my case, when type became zeroes and ones instead of metal, it became an option to me.

I would say that about 60 percent of our students get infected with the idea that they can do beautiful typesetting. Therefore, they are writing better term papers. They are thinking more about the problem of communication, and, since they are in control of it and don't have to explain a notation to some intermediary, then they are coming up with better notations. They will now consider a part of their own job description to be communicating in type, which they never would have thought if they had only a typewriter. My own experience is mainly with computer science students, but other parts of the community are affected, too. You find a lot of chemists and a lot of physicists, and musicians to a great extent.

Even when we had only low-quality, low-resolution printers, the precursors to TEX excited people. Stanford, Carnegie-Mellon, MIT, and USC were given four obsolete XGP printers, which Xerox decided not to market, about 1972, '73. These printers had a resolution of 200 dots to the inch, but that resolution was actually 240 dpi in the middle of the page and 150 at the edges of the page. (Words looked different on different parts of the page because the machine was intended to scan in at the same distortion and scan out. It was not intended to be a computer generating the image; it was intended that the image was to be gotten by analog means and produced in analog.) The machine was of poor quality, but people had a lot of fun making fonts for it. After three years of this, so many people had come up to the Stanford AI Lab just to use that machine that the parking lot would be only half full on a day that the XGP was busted. An important part of their lifestyle was to be able to use this printer. And you see that there's this lurking tendency in a lot of people to experiment. When IBM puts out another type ball for the Selectric typewriter, Olde English or something, all of a sudden thousands of documents are created with Olde English in all caps. People just love to see something new that they can control and make words come out in a different way. This is lurking everywhere, and it is blossoming now because it's becoming available to people through less expensive machines all the time. So I know a revolution is coming. Some of the output people generate will be atrocious, but it will also have the good effect that people will take pride in their work; they will put some more time into it and do a good job.

BYTE: *Is* TEX *finished at this point?*

Knuth: Yes. Absolutely.

BYTE: *Are you going to move on?*

Knuth: I'm going to write volume 4. When I get back from sabbatical I'm going to spend three months gearing up to work on the book and start writing in January '87. ■