An Interview with

DONALD E. KNUTH

OH 332

Conducted by Philip L. Frana

On

8 November 2001

Stanford, California

Donald E. Knuth Interview

8 November 2001

Oral History 332

Abstract

Donald E. Knuth is Professor Emeritus of The Art of Computer Programming at Stanford University. In this oral history, Knuth ranges over a number of subjects in software development including software patenting, alphametics, structural programming, and collaboration. The oral history includes material on the writing of *The Art of Computer Programming* as well as his early education and Lutheran heritage.

This is an oral history with Donald E. Knuth, conducted on November 8, 2001, in Stanford, California. It is conducted by Philip Frana for the Software History Project under the auspices of a National Science Foundation grant.

Frana: Thank you so much Don for agreeing to do this. I must say that after reading *Things a Computer Scientist Rarely Talks About*, I was somewhat tempted to just ask you random questions from the 328 oral histories we have in our collection.

Knuth: Good idea.

Frana: But I did come up with a list of questions that I didn't have answers to. I met with Edsger Dijkstra over the summer, and you and he seem to be, I guess along with just a handful of other people, some of the earliest leaders in computer science. I notice, in particular, that you were both Burroughs consultants?

Knuth: Right.

Frana: Or Burroughs Fellows? Were you a Fellow?

Knuth: No, no. I was just a consultant to Burroughs. The story is that I went to college at Cal Tech - - I started grad school in 1960 - - and during that summer I wrote a compiler for Burroughs. We can talk more about that if you want. It was an Algol compiler that was for their drum computer, the 205. And when I came out here I found that they had a real good group of people in the Pasadena section of Burroughs. It was called the Electrodata Division. And I had fellowships from the National Science Foundation and from the Woodrow Wilson Foundation. But the fellowships had a restriction that you couldn't do anything except be a graduate student; you couldn't do any consulting or have any other income whatsoever. So I turned down the fellowships and worked for Burroughs as a consultant. That was my connection with computing while I was a math student at Stanford. And at Burroughs I had a wonderful relationship with the people there and I worked with both software and hardware groups. And I think I might have met Edsger for the first time when he came up to visit in the early 60s, about 62 or 63, maybe. And then I met him a few more times on trips to Europe. But he also had Burroughs, I guess, as his connection to the real world from his professorship in Eindhoven.

Frana: Did you write a series of white papers as Edsger did?

Knuth: No. He has this series *EWD 1* through, I don't know, *2500* or whatever it is.

Frana: Infinity…

Knuth: Yes. And I now have them on a CD, which is great.

Frana: Right. And they are now Web accessible too.

Knuth: Right. But he doesn't put indexes in his books. So when his *Selected Writings on Computing* came out, which reprinted a whole bunch of these *EWD*, I had to read the whole darn thing to find out what he said about me. I couldn't just look in the index and look up the pages. I wondered about his opinion, because he's quite a fervent critic of what he likes and doesn't like, and that's his big strength. I tend to be a bit more wishy-washy in my opinions. [laughter]

Now, I met Peter Naur in 1967. I remember that specifically because we had known each other only through publications and correspondence while he was editor of the *Algol Bulletin*. And he is another man of very strong likes and dislikes, and strong opinions. And it was interesting because I had been working on *The Art of Computer Programming*, and I sort of, at that time, viewed it as a fairly good outline for a curriculum in computer science - - if computer science were ever to become established as a discipline. I thought that my book went through the basic topics that should be included in computer science work. And Peter Naur and I had both been invited to Trondheim to speak. The University of Trondheim scheduled one day when they were trying to plan curriculum. This was in the spring of 1967, and they had invited both of us to go up there and just talk for an hour on what we thought would be a good outline for their students. And it turned out that Peter and I independently had the same outline. So we got to be friends.

Frana: Were you including a course on data structures at that point?

Knuth: Right, right. He published these things in a little booklet. It was only a sketch. He never developed things to the frontiers like I tried to do. But he did stucture the field in the same way and he had this nice word, which he introduced into the Danish language, "*datologi*," a concept that got called "computer science" by default in America.

Frana: Now do you object to that title "Computer Science," as strongly as Edsger does?

Knuth: I like it, actually. A rose is a rose. Words get their meaning by what they signify, not really by etymology. I mean, consider 'mathematics'. It was probably very badly chosen, but now everybody knows what mathematics is. So it doesn't really matter what you call it, as long as people start to understand what the term signifies. I object in some ways to 'informatics,' because I think it emphasizes the stuff that algorithms work on, rather than the processes. It seems that information is more of a static thing than a dynamic thing. But so what? I would be happy with that term too. And for the same reason, I didn't think "datalogy" was the optimum word. In Sweden they invented the word "*dator*" for a computer. And that I think was a brilliant choice. But most of the time words aren't optimum and we just - we stop thinking about it after a while. We don't even realize what the original meaning was.

Frana: Now the department down at UT Austin is called computer scienc<u>es</u>, plural.

Knuth: Oh, okay. Texas has to be bigger.

[laughter]

Knuth: I think it is plural in a few places. Wisconsin maybe … But we only have one here.

Frana: Do you find that odd?

Knuth: No. I just think it is a quirk of history.

Frana: Do you find people in America to be a-mathematical in general?

Knuth: Well, certainly it seems the way things are going. You take any particular subject that you are interested in and you try to see if somebody with an American high school education has learned it, and you will be appalled. You know, Jesse Jackson thinks that students know nothing about political science, and I am sure the chemists think that students don't know chemistry, and so on. But somehow they get it when they have to later. But I would say certainly the students now have been getting more of a superficial idea of mathematics than they used to. We have to do remedial stuff at Stanford that we didn't have to do thirty years ago.

Frana: Gio [Wiederhold] said much the same thing to me.

Knuth: The most scandalous thing was that Stanford's course in linear algebra could not get to eigenvalues because the students didn't know about complex numbers. Now every course at Stanford that takes linear algebra as a prerequisite does so because they want the students to know about eigenvalues. But here at Stanford, with one of the highest admission standards of any university, our students don't know complex numbers. So we have to teach them that when they get to college. Yes, this is definitely a breakdown.

Frana: Was your mathematics training in high school particularly good, or was it that you spent a lot of time actually doing problems?

Knuth: No, my mathematics training in high school was not good. My teachers could not answer my questions and so I decided I'd go into physics. I mean, I had played with mathematics in high school. I did a lot of work drawing graphs and plotting points and I used pi as the radix of a number system, and explored what the world would be like if you wanted to do logarithms and you had a number system based on pi. And I had played with stuff like that. But my teachers couldn't answer questions that I had. I had proved that 1 = -1; and they couldn't see anything wrong with it. And I thought, well, if 1 = -1, this is not so good. But I loved my physics teacher, and I decided to major in physics instead.

It wasn't until my sophomore year in college that I switched over and realized that mathematics was really the thing for me. And this was partly out of selfishness. I liked the fact that with mathematics you could know that an answer was correct. With physics you never would know. You could go through your whole life and you would never know if anything you did was right. With mathematics you keep getting this reassurance. And so being the kind of person I am, that was nice. In my book on science and religion, I point out that I don't mind a few mysteries; still, I like to have some things certain. So that was what drew me into mathematics. I couldn't imagine how an astronomer could be happy just theorizing about what was happening on stars far, far away. You'd never be able to go there.

Frana: Do you have an answer? Are American students different today? In one of your interviews you discuss the problem of creativity versus gross absorption of knowledge.

Knuth: Well, that is part of it. Today we have mostly a sound byte culture, this lack of attention span and trying to learn how to pass exams.

Frana: Yes, it suddenly occurred to me that you've written many times that mathematics is reassuring. But in your Kyoto prize lecture in 1996, you said something like, 'Mathematics belongs to God.'

Knuth: Well, that was in a different context. It was in a context of patents or something like that.

Frana: Okay, okay.

Knuth: I said that you wouldn't claim ownership over the number ten. But I've actually had second thoughts about such things when the numbers get bigger.

Frana: What is your opinion about software patenting?

Knuth: I am against software patents on any idea that is so obvious that you would expect a typical student to invent it. I can maybe be persuaded that it is okay to have a patent on something really deep, like some interior point algorithm or something. But, certainly if I had to write the T$_E$X system in today's environment I would never start, because I would have to worry too much about getting permission to use hundreds of ideas. Of course, fortunately, we got most algorithms into software before people started taking this dumb view of patenting. But I think, anyway, I think software writers should be paid for services, for customization, for adapting programs, for the knowledge they have on how to maintain things. But not for the algorithms; not for the methods that are used. The method, that is something that is usually – consider the analogy I made to the patent office. I wrote an open letter to the patent commissioner. I said that algorithms are like words to a writer, or like lawyers have precedents. What would the law become if every time lawyers had a precedent they would charge a fee to other lawyers for citing their precedent. That would be, I think, very much analogous to what people want for software.

Frana: But with footnoting even, for an historian to pony up a quarter when they cite someone else's work . . .

Knuth: Or if you don't cite them it is worth a dime.

Frana: Have you seen the movement of software development overseas at all?

Knuth: No, I just came back from Munich and they said that the European Commission is analyzing the whole question of software patents. So one of the questions they asked me over there was what do I think about it? And I said, ‘Well, I hope you don’t go for it.’ I’ve been happy with what I’ve read from Britain. I think their attitude is quite healthy. But I don’t follow the issue closely. I am not a crusader for this like Richard Stallman. I found out that my talents lie in writing books about programming. I can talk to people at cocktail parties and say, ‘Oh, yes, I like this or don’t like this,’ but I don’t go out and give speeches about it and so on. Stallman encouraged me to write an open letter to the patent commission, once after a long phone conversation. I grudgingly gave in. He thought it would do some good. I doubt if it did, but anyway there it is.

Frana: Now beyond raw training, several computer scientists I’ve talked with have described mystical experiences. Is this an example of chance favoring the prepared mind? Is that what this is really about?

Knuth: Well, that would probably be the best explanation from a rational point of view. I mean Poincaré talks about these things, and many other people, Hadamard for instance. And you get this Eureka moment. Somehow, all of a sudden, you know that something is working. And usually it is.

Frana: The reason I ask is that we are sitting in this room; obviously on tape I can’t fully describe it, but there is shag carpet with interesting patterns on the floor, and you’ve got some pictures on the wall.

Knuth: It is kind of a psychedelic room.

Frana: Yes, it is multicolored. It has more than enough light bulbs, for instance, sixteen on the ceiling alone. Is that inspirational for you?

Knuth: No, no. We haven’t used this room very much actually, but we wanted to do something different. We like the idea of having different kinds of spaces in our house, not just beige rooms. And so when we saw these great carpets in Norway we fell in love with them, and we thought, ‘How can we use them at home?’ We also experimented with the copper panels here and things like that. But this room has never lived up to its expectations. These lights that are on, you can make them dimmer and brighter.

Frana: I thought maybe this was an idea room.

Knuth: Yes, well, it would be nice to have an idea room. But last night, I got an idea in another room - - while I was setting the dining room table. It was a dumb idea, but I might as well tell you. In the part of my book that I’m writing just now, one of the examples I’m using for algorithms is a puzzle called “alphametics” where you have words, then you add them together, and then you substitute digits for the letters, and it checks out. So as I am setting the table last night, it occurred to me that I should try ‘KNIFE + FORK + SPOON = SUPPER’. I realized in a flash that these words involve only ten letters, because, you know, the ‘K’ is used in both ‘KNIFE’ and ‘FORK’ and so on. So you count and there are only nine different letters in ‘KNIFE’ and ‘FORK’ and ‘SPOON’, leaving room for one additional letter. But anyway, as I am putting down the knife, fork, and spoon -- you know, I am thinking alphametics. And so I go up to my computer and try it. Unfortunately, however, ‘KNIFE + FORK + SPOON = SUPPER’ has no solution. So I added SOUP. And then it has exactly one solution; it is perfect! Another variation, ‘KNIFE + FORK + SPOON = DINNER’ had two solutions so it wasn’t so good.

I’ve been thinking about alphametics because they teach interesting lessons when you try different algorithms. You can do it the “brute” force way where you just blast away and go through all the possible permutations of letters versus numbers as fast as you can. There are three and a half million permutations, but if you only spend five machine instructions on each one, it is still much less than a second to go through and try them all. But then you can be more clever, and you can start working from right to left. You know, first the units digits have to check out, and then you figure out if there is a carry, and then if the tens digits work out. And by working from right to left, you don’t have to go through all 3.5 million permutations, you might only have to go through a few thousand. Yesterday I had played around with other ideas; during the day I had written a program for a left to right method. Some choices at the left are impossible - - you know, you can’t make this guy a nine because he’s too big. So that was in my head yesterday, alphametics. I invented a few really fun ones that are going to be in my next book, you can check it out.

Frana: Okay. This kind of puzzle-solving extends all the way back to your childhood. You stayed home a couple of weeks from school. There was a contest, right?

Knuth: Yes, when I was in the eighth grade. You’ve certainly done a lot of homework about me. That’s impressive! I stayed home at the time because there was a contest sponsored by the Zeigler’s Giant Bar Company and they said, ‘How many words can you make out of the letters in the name Zeigler’s Giant Bar?’ I realized that if I took an unabridged dictionary there was a fairly good way to search through it, knowing that I was only going to use at most three A’s in a word, and things like that. But I wouldn’t ever have to look at most parts of the dictionary. For example, the words didn’t have a C, so I didn’t have to go through the C’s and so on. So it was a two-week project. I went through all the relevant parts of the dictionary and got a complete list of these things. I had more than twice as many words as the judges had on their master list. And I forgot to use the apostrophe.

Frana: With the apostrophe, you would have had even more.

Knuth: Yes, right, for words like “bar’s”.

Frana: I want to go back in time again and ask you about programming languages. You mentioned you worked on Algol compilers for Burroughs?

Knuth: Algol compilers, yes, right.

Frana: Why did Algol 60 never catch on?

Knuth: Well FORTRAN was very strong. People thought they had a big investment in that language and they didn't realize they were going to be changing their programs over and over all the time. But that was only one of the reasons. Probably the most important was that FORTRAN was more complete. Algol lacked a good way to get nice looking and readable output. That was never part of the standard. It was not considered real programming to deal with input/output issues. And FORTRAN had a way that worked. It wasn't elegant, but there it was. And since Algol didn't address this well, I think that was one of the main stumbling blocks in those days. So Algol never got enough momentum to carry it.

Frana: Did Algol 68 stress that?

Knuth: Algol 68 was hardly ever implemented in a good compiler. It was a huge language. It was too complicated to put much time into it.

Frana: Was there anything to the charge that 'Too many cooks spoil the broth'?

Knuth: They had a large committee; but it was really Adriaan van Wijngaarden who called the shots on Algol 68, so that wasn't the reason. It was too ambitious of a project, just like PL/I. Just to cope with a 300-page language description, I mean, you just can't do it. So, Niklaus Wirth came out with Pascal, which was greatly appealing then to the people who wanted a clean language and could understand it.

Frana: I hear many people complain about structured programming, and yet so many people think it is an important subject.

Knuth: Structured programming is important, of course. I wrote a long paper about structured programming. It was called, "Structured Programming with Go To Statements," and in that paper I had somehow fifty co-authors, because I incorporated a large number of comments from people all over the world to whom I had circulated the paper before publication. And I tried to make a case for what the strengths were of structured programming, and a case for what the strengths weren't.

The debate was a matter of form over substance. A lot of people thought structured programming just meant that you restricted your programming to not using certain features. And I hated that, just the way Hilbert hated intuitionist mathematics, which says that you can't use the law of exclusion when you are doing a mathematical proof. Well, I didn't want to be handcuffed with prohibitions of every feature that anybody had ever been able to abuse, because people were going around saying such and such is 'considered harmful'. You know Edsger said the Go To statement is considered harmful; and it <u>is</u> harmful, if you misuse it. But I made a catalogue of all the errors I found in T$_E$X, and 3 or 4 percent of the errors were due to misuse of Go To statements, and 3 or 4 percent of the errors were due to misuse of global variables, and 2-3-4 percent of the others were due to the misuse of if/then statements, and so on. So if you abolish everything that I didn't use properly at some point in the program, you'd have to take all features out of the language.

My point was that structured programming was still a wonderful revolution, because it taught us how to think of a program as having some structure and gave us some intellectually manageable ways to deal with the complexity. But people were attacking the form only.

Frana: So you were calling for elegance.

Knuth: I was calling for understanding and being able to have a way of comprehending the goal, which the abolishers of Go To tended to forget. But really a Go To statement is analogous to an assignment statement. An assignment statement gives a variable a new value and a Go To statement gives the program control a new value. And both of these concepts are nontrivial and need to be understood. I think it was Menabrea who asked Charles Babbage "What if you want to 'Go To' this part of your program?" and Babbage didn't have the foggiest idea what Menabrea was talking about. So it is a nontrivial concept to assign a new value to the control. It means logically that you have a certain abstraction, that corresponds to what it means to be at this point in the control. If you don't have a good abstract meaning for that, then your 'Go To' statements are very likely to get you in trouble because you might be going places where the abstractions collide with each other. That is why it was very dangerous to use them without discipline.

But I still find certain places where, when used properly, a Go To is better than to fake it. Just because somebody considers Go To a mortal sin does not convince me. It is like the zero population growth movement: The goal isn't really to have zero population growth; the goal is to improve people's quality of life. But they substitute a quantitative goal for the qualitative goal, and that's like viewing structured programming as only a non-'Go To' type of a program, which is foolish, and I am sure Edsger didn't mean it that way at all.

I have to tell you a little joke though. Actually, I gave a lecture in Eindhoven in the early 70s, and I was writing an algorithm on the board at the time; it was an algorithm for pattern matching in strings. When I came near the end I said, 'Oh, Edsger, is it allowed to say 'Go To' in this room?' He said, 'I saw it coming.'

[laughter]

Frana: I neglected to check this before I came out here, but were you at the Garmisch or Rome conferences on software engineering?

Knuth: No.

Frana: I didn't check the index again to see.

Knuth: No, I stopped going to conferences. It was too discouraging. Computer programming keeps getting harder because more stuff is discovered. I can cope with learning about one new technique per day, but I can't take ten in a day all at once. So conferences are depressing; it means I have so much more work to do. If I hide myself from the truth I am much happier.

Frana: I am very much in agreement with you. I'll often go to a conference - - I see one presentation and then am off working on my own.

Knuth: Well, for each day spent at a conference, I need five or six days to catch up afterwards. Of course it is nice to see old friends again, but I had been there and done that by the middle of the 70s.

Frana: Was "software engineering" an unfortunate term?

Knuth: A lot of people certainly like it and find it inspiring. I don't know. I was never enthused about the idea of software metrics, which was one of the pillars of software engineering. Because people thought that since it is an engineering discipline we need to measure things. You think, well, everything that works for one branch of engineering is going to work for the others and all we need to do is get our act together and be more disciplined about it and then the problems will be easier. But I think there is just going to be no Royal Road to programming. It is one of Edsger's greatest insights to compare the amount of intellectual effort that goes into computer programs, as compared to other kinds of human activity. In physics things fail, but they fail slowly; they sort of fail smoothly. But with logic there is no continuity; it just goes bad and everything is a mess. It is quite a different thing. There are many great achievements - you know, building a huge airplane is a tremendous achievement, making a movie like *Snow White and the Seven Dwarfs* is a tremendous achievement, all kinds of things - sending people to the moon - require massive cooperation of many people and so on. Yet such achievements are different from what happens when you have ten million lines of program. And so software engineering hopes to change that by making these lines of code much more like physics, so that they fail smoothly or something like that and there is all kinds of redundancy. In fact, it helps a little bit, but it doesn't reach the goal.

Frana: You end up with a different set of problems?

Knuth: You push the problem, yes, you just shove the problem off onto someone else's lap. It doesn't get you the ... It is very tempting to think that if we just teach everybody how to use their statements correctly they will be able to come up with the programs that do subtle things. Well, no. It takes tough thinking and there are some lines -- there is one statement -- if I had to charge for the T$_E$X program, there is one statement that went through twelve revisions, and I had to think very hard about it each time. For that statement I would say, 'Okay, pay me for that one.' But I wouldn't know any technique of software engineering that would have got me there without just a certain amount of anguish. That is what is behind my comments. There are no magic bullets.

Frana: That sounds very similar to something Duane Whitlow said to me about SyncSort. There is this one line, he said, that really is the line that matters, and the one he had to work the hardest on.

Knuth: So you interviewed him then?

Frana: He actually was a participant at Xerox PARC last year when we had a history of the software products industry conference. He gave a paper.

Knuth: Oh, okay. So this was one of the patents? Yes, I finally found out…

Frana: Were you in the audience there?

Knuth: No, no, but I had to reach him. I mean I was trying to find out what the L stood for, Duane L. Whitlow, and I guess it is Leroy, anyway, and then I had to find out if you capitalize the R. I finally got in touch with him, but I had never met him, no. But I did like the idea of SyncSort, which I put in *Volume 3* a couple of years ago.

Frana: What do you think of "software physics"?

Knuth: I don't know what that is.

Frana: It is Ken Kolence's…

Knuth: I don't know anything about it. But I think he is looking for a magic bullet.

Frana: It just doesn't exist?

Knuth: No, it might exist, but it doesn't exist in my universe. I know what I am good at in my narrow domain. I am not saying everybody in the world should be like me. I only know that whatever it is that makes me like me, it correlates well with getting computers to do good things. But there are lots of people that are not like me and they don't like the analogies that I like. And if I try to explain something to them they will say, 'Oh, it is much easier if you understood and do this in terms of the law of thermodynamics' or something similar. Okay, but that goes in one ear and out the other for me. So when you say "software physics" it sounds to me like this guy from another world has his way of looking at it. His way might be better than mine, but it isn't mine, and I doubt if I'll be able to contribute one way or another to it.

Frana: To change directions a bit, and forgive me if I have this wrong, because I either read this or heard this, that at Stanford, during the Vietnam War, you joined the protesters. Is that right?

Knuth: Well, one day Bob Floyd and I sat in front of the computer science building, joining the students who were picketing. It was when Nixon had invaded Cambodia. We said, 'Well, this is kind of the last straw. We can't go on doing business as usual.' Although, actually we were talking to each other about sorting algorithms. And Bob and I were, in fact, inventing some sorting networks as we were sitting there. But we were outside the building because we wanted to be counted in the statistics of people saying so-and-so many people protested today. And the government's actions were just outrageous, it seemed to us.

In general, I rarely take a stand on anything, but I work it so that when I do do something, my exceptional action is meaningful. It is not that I just go with every breath of the wind.

Frana: You also spoke out against SDI in the Reagan years, is that right?

Knuth: I didn't. Only privately. I don't think I ever took a public stand about it.

Frana: It must have been a conversation I had with someone then.

Knuth: I think anybody who knew anything about the reliability programs was against SDI. I wasn't exceptional in this regard, it was just purely a boondoggle.

Frana: You know Gio [Wiederhold] at Stanford has done a lot of rocket research?

Knuth: I never knew that.

Frana: Oh, you didn't know that? That was his first job. He spent a lot of years on the problem.

Knuth: I only knew that he went into medicine.

Frana: Liquid fuel rockets, I guess.

Knuth: Oh, okay.

Frana: Is that a problem? Do you find that troubling in retrospect, the Cold War research that computer scientists did?

Knuth: Well, I have very limited experience with that. I spent a year working at IDA in cryptography and everybody I met there was committed to good work. I had no qualms about the part of government service that I saw in my colleagues there. I also knew that it wasn't natural for me to keep secrets. I was the kind of person who wants to be a professor and tell students everything, but working for them I was not allowed to tell my wife what I was working on. You know, I still haven't talked to her about it. I didn't like to be that way. So I figured that was a year of national service, and I would leave -- not because I didn't like it, it was just because I knew that my own contribution would be better in something where I wouldn't have to keep my mouth shut.

Frana: Well, let's move on to something more fun. You and John von Neumann, and Leo Szilard and several other people were all Lutheran High School attendees.

Knuth: Wait a minute. I can't believe this. John von Neumann? Are you …

Frana: John von Neumann went to a Lutheran High School.

Knuth: No. I thought he was – okay, he's not Jewish? You know, I met his brother Nicholas.

Frana: No, he's not Lutheran, but he did attend the Budapest Lutheran High School.

Knuth: Oh, that I never knew.

Frana: And I am wondering if there is something Lutheran about computer science.

Knuth: This is a very hard question to answer. This would be like saying, oh, a certain amount of people drink pasteurized milk when they are young -- I don't know. I doubt if there is. Luther did stand for the idea of independent intellectual activity. In religion he wanted both his head and his heart to be there and not – he didn't want to be detached from logic and then just become - just say he deserves to be saved because he can believe the most impossible things. He said, no, let's keep questioning stuff. And so sure, that would correlate. But I wouldn't say Lutherans are ahead of Presbyterians, or Muslims, or anything.

Frana: I was raised a Lutheran too, and that was my impression as well, that Luther was a scholar and that was a good thing.

Knuth: Yes, I come from a tradition where you can question things. But then, you also have this, 'Here I Stand', and your conscience, and so on. And that's fine. But I would say, the way it strikes mostly in my scientific work is the model that God knows what I am thinking. So I don't feel the need for privacy the way a lot of people do. In some sense I don't think I ever have privacy.

Unfortunately, Edsger had some very bad experiences with religion early on; he got to be very sensitive about it in the negative.

Frana: We never talked about that.

Knuth: Well, good. He insulted one of my students once because the guy was talking about programming and Edsger said, "Were you raised as a Catholic?"

Frana: Wow. Was he right?

Knuth: Yes, but of course he was, the guy was French. But Edsger somehow thought that there was something Catholic about his science, which didn't make any sense.

When I wrote the book *3:16*, I found out to my surprise that some people have really bitter feelings about the Church because of something that happened with their parents, or something else. And this was an aspect I just had not been aware of before.

Several dozen people with different backgrounds volunteered to read the first draft of that book. So I had atheists, who would write ten page letters, make comments about the work, because they were interested in the ideas too, but they also told me about their history and I would learn from them. And Edsger definitely reacts against religion; against organized religion anyway.

But I am trying to explain to you in what ways did my religious education really interact with my scientific life. And I think it was not that I had the best science teachers, or things like that. It did give me this feeling of being part of a God-driven universe; it did affect the models that I have about my own life in ways that probably have some effect. I doubt if it is going to make me a better programmer, or a worse programmer. It just affects which kind; you know, I don't think I would be suitable to do top work in cryptography because of my not being that sneaky to understand the attacks that somebody might make, and also my not being so convinced that I need secrecy.

Frana: You accept the fundamental goodness of most human beings?

Knuth: Well, yes.

Frana: You have hope, or faith I guess.

Knuth: I can see the need for some low level of security, but I can't be convinced very easily that strong crypto is important. And maybe I could, maybe I couldn't, but anyway, because of my basic nature, I just don't think that I am suitable for being a top person in that kind of work.

Frana: Now the reason that I asked you about Luther and computing is that there is a long history of the printing trades in the Lutheran faith. It is how they communicated the Reformation to people.

Knuth: Yes, true.

Frana: Did that have any affect on your decision to do digital typography?

Knuth: It could have been implicit because my father did so much work with printing.

Frana: Oh, he did?

Knuth: He did a lot of work for all the churches around Milwaukee. My father had a mimeograph machine at home and he would make programs for special events and that kind of thing.

Frana: You mean leaflet kind of programs, not computer programs?

Knuth: Right. That's right. I've got to be careful. Right. And music, especially for choirs, and things like that; he would spend a lot of time by hand making low cost materials possible for choir. And it wasn't high tech printing at all. It was very much the kind you do for pennies to help out. It was a mission for him; it wasn't a way to make money. He did have a company, but he didn't make a profit on it. He did it as a service. In fact, he called it 'sERVice', Erv's Service. His name was Ervin.

Frana: What was your father like? I've never read any accounts of him other than that he was an organist.

Knuth: Oh, he was a good toastmaster, a kind of charismatic teacher, a choir director, and he had a good sense of humor. He took a lot of personal responsibility for everything that he did and worked hard on making many different things. He was treasurer of Lutheran High School, and he would pay the professors out of his own pocket if the budget money was not there. And he took responsibility very seriously.

He felt a call to local people, where I feel a call to global people. I don't do that much for my own friends the way he did. He did everything for his friends. But I've got people in Russia and Poland and China writing to me all the time saying thank you for what you did. And that is a completely different life from my dad, because he was the kind of person who never wanted to make a name for himself. He was very happy knowing that there were a hundred people, close friends - that he was improving their lives and so on. But me, I'm getting glory for stuff that doesn't take any more time, but just happens to be connected in a different way.

Frana: The Palo Alto area, though, is not like Milwaukee either.

Knuth: No. He was the kind of guy that makes America tick. He was really the lubrication that made things work. Whatever needed to be done, he would do it. And my mom is very similar.

Frana: She is still alive?

Knuth: She is 89 years old now.

Frana: Did your father live to a ripe old age then too?

Knuth: No. I think he was younger than I am now, 63 maybe, I think, 62. He died very suddenly. But my mom is still not retired. She works in real estate in downtown Milwaukee. And both of them were always doing volunteer work and various things. Not the kind of thing that ever gets acclaim. Lutheran education was really my dad's main goal in life. And he had been to college at River Forest in Chicago, and then became an elementary teacher in Milwaukee, and then went to the high school there. He saw education as a mission that could be accomplished in a loving atmosphere.

Frana: Yes. Now you chose to compartmentalize your Lutheran faith, at least at first. Was that by choice?

Knuth: I'm not sure what you mean.

Frana: You were 'Lutheran' on Sunday morning.

Knuth: Oh in that book I said …

Frana: I think that was the word you used, 'compartmentalized.'

Knuth: Well, I used the word when somebody asked me a question with respect to Bill Clinton. But I said that because it was just, you know, that was a buzzword.

I would say when I was young I was just pretty much mechanical in all respects. I was a test-taking machine. I learned the subjects so I could pass exams. But I was only into quantitative stuff; questions of elegance and beauty, and great literature, and such things, I didn't . . . I was - - what would you call it? I was developmentally challenged in those areas until I was maybe 30 years old. And then I started to read some great books, and to understand, and to really appreciate great music. Before that, I would know how to please my teacher and I could do the right phrasing, but I didn't have that soul. But then something happened to me as I got older.

Frana: What happened?

Knuth: I have no idea. Maybe I was just too busy or something. Anyway, in all aspects of my life I tried to obey the rules. And I think I always had an inferiority complex. In some ways I would say well, maybe I have to prove that I can do this. In the back of my mind I knew that I was a lot smarter than people thought I was, but still I kept trying to prove myself all the time, and I wasn't doing stuff because I wanted to do it, I was doing it because I was supposed to do it. Not that I hated to do the work; I was simply being dutiful. And so if it was time to study math, I would study math, or whatever. So I was kind of mechanical in that way. And it was later that I began to see more of the emotional part of life. Who knows what the reason why? Probably my wife gave me that.

Frana: Where did you meet her?

Knuth: We were both students. I was at Case and she was at Reserve, and I was dating her roommate and we would go out on double dates sometimes. And I found out that she was a pretty nice person. So I went and talked to her once about some problems I was having with her roommate. And she gave such nice answers I decided I liked her better.

Frana: The oldest story in the world. Date the roommate and then find the other one more attractive.

Knuth: Yes. She was a year behind me in school. But then we just spent a lot of time in libraries studying together because we found we had a lot that we liked to talk about.

Frana: I'd like to ask you a couple of other questions about digital typography before we get too far away from that. There is this tantalizing comment that you make in one of your books, that good typography helps you make good programs, or better programs. What did you mean by that?

Knuth: I am not sure if I said that about programs. …[pauses] But certainly, one of the whole ideas of structured programming and literate programming is that you have to be able to understand its complicated whole: 'What is the program.' So you need some aids to this understanding, and typography is one of those aids. With good typography you can perceive the structure, instead of imagining the text as just a chaotic string of characters. It's much better when those characters are arranged on a page in some reasonable way. So typography is partly the arrangement of things, you know, like indentation. Things that are bold or things that are, you know, in small type or italics. And in that way, without that typography, it will impact negatively on the way I can perceive what is going on.

Frana: The wholeness of it.

Knuth: But I am not saying typography is going to be a magic bullet either. I am just saying that it helps a lot.

I worked, in fact, with the *Journal of the ACM* in the 60s, in part to help improve their standards of typesetting of computer programs because it was a new thing. It hadn't been faced before. And so I said, 'Let's think seriously about what is the right typography for programs, for the comprehension of the ideas, to get the point across to what the program means. In this new area we should look for matters of presentation.'

Frana: Here's a question that is much more specific: I know a lot more about Dijkstra's algorithm than I know about many other algorithms, because we had an extensive discussion about it. But is it very similar to your algorithm for finding the shortest path from the top to the bottom of a paragraph of text?

Knuth: It is the same. The way that I used the method is equivalent to Dijkstra's algorithm, except that I have to build the graph as I go. I convert the paragraph into a graph problem, but I discover some parts of the graph as I am going, to save time. It is his algorithm, but then I save time by not including things that would slow it down. For example, there is no point in exploring what would happen if you stopped the paragraph by breaking a line after the first word. Right. So I only look for the feasible breakpoints. And I learn what the feasible breakpoints are by applying Dijkstra's algorithm, and as I am applying it I know that some things are not feasible, so I don't even consider those. So it is an extension of Dijkstra's method.

But it is also, you can also view it as what they call dynamic programming, because discrete dynamic programming is essentially identical to finding the shortest path. So there are many points of view that boil down to the same underlying problem.

Frana: Okay. A few years ago in an interview you said that computer science, like most of other sciences, grows chiefly by lots of little steps rather than by giant ones.

Knuth: Oh, I did? Yes right.

[laughter]

Frana: I think you used the word 'chiefly,' as a qualifier. Have there been a few giant steps?

Knuth: Yes, the idea of structured programming is a giant step, but those are few and far between. And so just watch tonight during the question and answer session. Someone is going to say, ‘What do you think are the top five algorithms,’ you know, or something similar. I hate that kind of a question, because that is not really the way science grows. The important thing is not the top five; it is the bottom five. It is just that there are always five more. And the whole structure masses together by small steps.

Unfortunately, you can’t explain that to a Congressman who is going to fund the NSF. You have to tell the Congressman that we have this large project, and we are going to set these goals, and all this. But really, if you just take every scientist and say, ‘Do what you think is interesting,’ then you get the best science.

Frana: So, you know a little bit more than most scientists about the history of science, I would guess.

Knuth: Some parts.

Frana: Do see this as a cumulative process? That we are all standing on the shoulders of giants? Or are there Kuhnian paradigms?

Knuth: I think that we learn best by studying the past and seeing how other people came up with their ideas, and learning the process of learning by osmosis after seeing all these examples. And certainly the most striking thing to me is that human beings are able to go back to their history and see what previous humans have done and change because of that. Animals don’t - - or they do so very slowly. And so this is key for me, looking at source materials and seeing how ideas were when they were raw. I couldn’t have been anywhere near as effective if I hadn’t done an awful lot of reading of things from many previous centuries. If it is necessary I will get a friend to help me with the Chinese, or Japanese, or Hungarian, or whatever it is, but I am always looking at writings from previous times that are related to what I am pursuing now. Because to me, the way that humans evolved is central to the story. Maybe it is just because I am so egotistical that I want people to be reading my stuff in the future, so I read other people’s stuff in order to pay my dues.

Frana: Well, you know very well, though, that the past can be a strange place. In astronomy, for example, we have the geocentric universe.

Knuth: You have to put yourself in the minds of the ancient people. For two weeks in 1972, I was writing a paper about Babylonian algorithms. And I surrounded myself with Babylonian texts for two weeks. I immersed myself in reading as many tablets as I could so that I could say, ‘Oh, yes this is unusual,’ or ‘this is just same old, same old,’ when I was looking at another tablet. And I could start to try to get into the mind of those authors. They had a certain way of expressing things. They wouldn’t use algebra, but they had an equivalent. So I try to learn what their symbols mean and interpret their language. This is the thing that holds a lot of people back from reading sources. It is because they are unwilling to, or unable to, see or relearn the notations. That means, like with music, you have to either decide you are going to learn the other notation or else convert it into your own thing. With mathematics, I can usually do it without too much effort. But it does take a few hours of adjustment going in. With a little experience that becomes easier for you too. Music, I think, would be harder. Certainly some of the notations in music are so illogical that they only can be learned after years of training.

Frana: You have made the comment several times that maybe 1 in 50 people have the “computer scientist’s mind.”

Knuth: Yes.

Frana: I am wondering if a large number of those people are trained professional librarians? [laughter] There is some strangeness there. But can you pinpoint what it is about the mind of the computer scientist that is....

Knuth: That is different?

Frana: What are the characteristics?

Knuth: Two things: one is the ability to deal with non-uniform structure, where you have case one, case two, case three, case four. Or that you have a model of something where the first component is integer, the next component is a Boolean, and the next component is a real number, or something like that, you know, non-uniform structure. To deal fluently with those kinds of entities, which is not typical in other branches of mathematics, is critical. And the other characteristic ability is to shift levels quickly, from looking at something in the large to looking at something in the small, and many levels in between, jumping from one level of abstraction to another. You know that, when you are adding one to some number, that you are actually getting closer to some overarching goal. These skills, being able to deal with nonuniform objects and to see through things from the top level to the bottom level, these are very essential to computer programming, it seems to me. But maybe I am fooling myself because I am too close to it.

Frana: It is the hardest thing to really understand that which you are existing within.

Knuth: Yes.

Frana: I had a follow-up question there but I’ve now lost it.

Knuth: Is it about the 2 percent? I am thinking that we have to realize that there are people that we aren’t able to understand very well the other 98 percent, and they aren’t able to understand us very well. But we build bridges. We can make friends with somebody who is closer to us, and sort of get a network of people that bring things together. I am sure that I can’t write the right user-manual for my mother; but I could collaborate with somebody who could do it the right way.

Some people have this other notion that you can change yourself to think a different way. But that is only true to a small extent. Here’s an example that came up in a recent discussion in

Germany: What if the quantum computing takes over? What if suddenly people can manufacture these things, which require a completely different way of writing programs? Well, it might very well be true that I am absolutely no good at quantum computing and that I am not going to be able to change. (And it would be easy for me to write the *Art of Computer Programming* if nobody was doing programs the old way, so I wouldn't have to keep up with literature anymore.) But I have a certain way of grinding out stuff that I am good at, and it is not just that I studied it in college; it is that I have this quirky way of thinking.

Frana: Like the way some people have difficulty with client/server architecture. The old mainframe guys that just, they tell me, they just don't get it.

Knuth: Yes. I can understand. It is like your software physics guy in a way too. But you know, they need to 'get it' in some other way, so let them be happy and get it in their way. And maybe the new way is going to be better, but I'll never be able to do it because I am different. I think that in computer science there is no one narrow focus. It is pretty much different from many other fields, because we are selected by our skills, by our profile of abilities, not by our mission to compute. People go into medicine with many different kinds of skills and things; their career is based on the goal of saving lives, making lives healthier, whatever. But my career is different. It is something that I happen to be good at and other people are happy that I am good at it, so I'll do it. But I am not advancing a cause, like medicine.

Frana: Now do you feel that there is more -- I hate this word -- but more "normal" science going on right now in computer science? I mean, quantum computing, who knows if it will work, it is still in the future.

Knuth: Well, there are all kinds of collaborations going on. And the thing is, as fields get more specialized I think the tendency is going to be more that people define themselves by having two sub-sub specialties. That is, I think I said this before, where you are going to be a person who happens to be good at one thing: that might be related to computer science, and something else: that might be related to chemistry.

Frana: So it will be defined by two sub-sub specialties?

Knuth: Yes, right, and this will make a kind of a web of disciplines. And it is going to be hard for quality control because there won't be that many people with the same two sub-sub specialties that will be able to review each other. It will be harder to be on tenure committees in those days. But it seems inevitable that the world is going to have to go to that kind of a model where people have two foci, a combination of abilities that makes them unique; they will realize that that is why they were born, because they can help bridge this gap. And there are so many different things to bridge in science.

I find that there are many parts of science that I can be interested in from an outsider point of view, but I can never feel that I could do myself. And there are other parts, where I feel it is my mission to do it, and I am part of it. We keep coming back to Dijkstra because he is one of the most universal people that I know. Almost any subject that we would talk about he would know an amazing amount about it.

Frana: But if I may, I would have to say that you exist in the other world, among the other 98 percent of us, far better than he does.

Knuth: Well, it is interesting that you say that. Dijkstra's writings may be more specialized, but not his informal conversations.

Frana: You seem to be able to make that jump.

Knuth: Well, I am not sure. As a student I found that everything I was doing in those days, it seemed to me, was a mixture of mathematics and writing. That once I had those skills, then I was just applying them in different proportions.

Frana: You said in your Turing lecture that, "Science is what we understand well enough to explain to the computer, art is everything else."

Knuth: Yes, yes.

Frana: But then why call the series, *"The Art of Computer Programming"?*

Knuth: Because we don't understand it, we haven't made it automatic. We haven't got an algorithm to write the programs yet. We are trying to convert programming into a science, but as we do then the art goes ahead. As science advances then art stays maybe a couple of jumps ahead, so far. And then I also believe that I'm describing the art of computer programming because of the elegance of it, the beauty, the aesthetics of it.

Frana: So you think of it really as both a science and an art?

Knuth: There is art, in the sense of fine art, and then there is art in the sense of artificial, not in nature. In my Turing lecture I tried to meditate about why I call it *The Art of Computer Programming,* and what does it mean. I went to the library and found fifty books that had both the word art and science in their titles. And I looked at the word in history as to how it was used. And I came to the conclusion that the right way to understand it is that art becomes science when we reach a level that we don't need to think about it anymore, we can program it. And that is the greatest mystery, at least in science now, is what it means to know something, to have cognition. It is a mystery what consciousness is, but if we knew what consciousness was then my definition would not work. I am sort of saying that once we cross the gap from the point of needing a brain to understand it, once the computer would understand it, then it becomes science.

Frana: It wouldn't be much fun if we really understood consciousness would it?

Knuth: Well, who knows? [laughter]

Frana: I think maybe we'd move on to a different level of consciousness.

Knuth: Yes, right, maybe we'd jump ahead again.

Frana: I don't know if you've read Bolter's *Turing's Man* or any of these critiques of the cybernetic vision of society. Have we become our machines?

Knuth: Those kinds of things, the most I've ever thought about them is all in my book, in the MIT lectures. The insight that I got from studying the game of Life was as close as I came to understanding some of these issues. And other people are way ahead of me on those things. But this all this is worth exploring. Again, it is like astronomy, you can never go 'out there.' So I am happier having most of my questions in terms where I know I've got the answers.

Frana: But you don't see that society has become digital?

Knuth: Certainly the world has changed a lot faster in some respects than I ever thought it would, and computers have become much more relevant to the world than I would have believed possible. It's still an oxymoron to speak about a 'famous computer scientist,' yet I think we are getting too much attention; too much credit for stuff compared to other sciences. But that will go away.

Frana: This is a terribly unfair question, but I was in the Valley a year ago at this time, and then again this year. And judging by the number of empty storefronts, things have really changed around here from this time last year. What happened?

Knuth: Well, certainly last year was the top of the boom. But if you look at Iowa, or anywhere, you will find that there is lots of turnover everywhere you go and the economy keeps reinventing itself all the time. There are cycles. So there has been, for a long time, and now the whole world economy is going through an adjustment. Future historians might think last year was the time when the boom was at its peak. In other words, there was such a shortage of skilled labor, where anything you wanted you had more ideas to do than people to do them. So this sort of builds on itself.

Frana: But then we ran out of ideas?

Knuth: No. No. You've got to get to the point where you have to get to the quality people, instead of the ones who are just along for the ride. But I am a writer, and I don't know much about money.

Frana: I know this is a terribly unfair question. They say there have been eight boom bust cycles since World War II in this industry alone, and this is just another one. But then there are people who say it really is different, and we have built up something over the 90s that was kind of a pyramid scheme.

Knuth: No. I don't think that. The world is different in a way, that is, you don't go back to pre-Internet time. The judgment has to be made as to how much and in what ways people are going to pay for this new world where some things are possible that were never possible before. So now you have to think of what is a fair way to compensate for this, and what we can handle, and how to do it. Nobody has a good way to decide that except by trial and error. It is sort of like inside of our body there are all kinds of cells attacking other cells all the time and fighting it out, and that seems to be the best way to cope with complex changes. It is chaotic, these corpuscles are going after other stuff and it is a war in there, and it is not just somebody applying Dijkstra's algorithm to find the shortest path. You compete.

Frana: We are restoring equilibrium and regaining homeostasis?

Knuth: Yes, something like that. But the fact is, there is great potential, that will take years of work, that is unrealized by the machines that we have now. And it is difficult to do those jobs and it takes a lot of work. But I am never going to say it is a stupid idea to do this work and we should all stop working on it because no one is going to buy it. That would be the most foolish thing possible. All of these skills that people have here are vital, and in fact, we are going to need millions more such people, and my 2 percent rule predicts that there won't be enough. There is always going to be a shortage of computer-science-oriented skills, and we're far from exploiting these computers as they are today to what they could be doing for us.

Frana: Now I have to check here, but did you supervise twenty-eight Ph.D.'s?

Knuth: Yes.

Frana: I understand how Stanford provides seed money to students who have ideas. But did any of your students get that kind of help to get started and are most of them university people now today, or are some of them in private industry?

Knuth: I think they are about half and half between industry and academics. My Brazilian student, I think, did some kind of company building. I've lost touch with one or two of them, but I think that in industry, they have mostly been working in research labs -- not founding anything themselves, but being part of a team that somebody else is managing. In the university they've been chairing some departments and things like that, but that is not the same thing as starting Yahoo!.

But I would say that those twenty-eight students were twenty-eight completely different kids. It is like proving my point that computer scientists know how to deal with different cases.

Frana: That is a nice real-world example. I understand people are unpredictable that way. Have some of them become your closest confidants?

Knuth: We are close in different ways. So I'd say some of them I have lost track of, but with others we help each other when our computer crashes, or we work together on editorial projects, or whatever it is.

Frana: Just to repeat the question, with computer science as a whole, or academia as a whole, can you name people that are your closest confidants. People that you feel you can call and bounce an idea off of?

Knuth: My wife. Also, as I am writing drafts, whatever people are the experts in the section I am working on, I am always sending them materials to check on. For example, last week I bounced an idea off Professor Carla Savage at North Carolina. I met her only once, and I wouldn't call her my closest confidant; but in a way, on the subject that I am working on now, it is something that she has many publications. And so I said, 'Carla what do you think of this problem? I just worked on it for four hours. I can't get it, but I don't want to just put it in the wastebasket, so I might even have time to put it in my book.' Then she said, 'Hey Don, it is a nice problem and I think I can do it.' The next day she sends me an answer and I looked at the answer and I said, 'Yes, just patch it in this one way and now I think it is right, and I'll put it in my book. It is elegant and beautiful.' That is just one example of the way I'm working now.

So you see, I work in batches. Every six weeks I am into a different subculture of computer science as I write a different part of the materials for *The Art of Computer Programming*. And during those six weeks I have a different set of closest confidants.

Frana: I get you. So you really do have a very broad set of collaborators?

Knuth: Yes. It is just fantastic how many people have helped me with this thing. And then I put new stuff on the Internet, and in a week I get one hundred letters. A fourteen year old boy in Germany recently pointed out that I misspelled Nuremberg. (I said Nuremburg.) I am getting tremendous help.

Frana: And this is one of the reasons why you don't use email so much anymore?

Knuth: Oh, I would never get anything done.

Frana: This way you can still fire off as many messages as you like.

Knuth: I can use the Web, where people who are interested can find this stuff. I don't send it to them, but they find it if they are looking for it. But with Carla, I sent her the message because I knew that she was an expert on this subject and that she wouldn't mind being interrupted.

Frana: This is the last question that I have, but there may be other things that you'd like to contribute to this interview. And maybe I'll find out the answer tonight at Xerox PARC: What is it like to be the chief spokesperson for computer science?

Knuth: I don't know if I am.

Frana: Others have said that.

Knuth: I can be a writer, who tries to organize other people's ideas into some kind of a more coherent structure so that it is easier to put things together. I can see that I could be viewed as a scholar that does his best to check out sources of material, so that people get credit where it is due. And to check facts over, not just to look at the abstract of something, but to see what the methods were that did it and to fill in holes if necessary. I look at my role as being able to understand the motivations and terminology of one group of specialists and boil it down to a certain extent so that people in other parts of the field can use it. I try to listen to the theoreticians and select what they have done that is important to the programmer on the street; to remove technical jargon when possible.

But I have never been good at any kind of a role that would be making policy, or advising people on strategies, or what to do. I have always been best at refining things that are there and bringing order out of chaos. I sometimes raise new ideas that might stimulate people, but not really in a way that would be in any way controlling the flow. The only time I have ever advocated something strongly was with literate programming; but I do this always with the caveat that it works for me, not knowing if it would work for anybody else. When I work with a system that I have created myself, I can always change it if I don't like it. But everybody who works with my system has to work with what I give them. So I am not able to judge my own stuff impartially.

So anyway, I have always felt bad about if anyone says, 'Don, please forecast the future,' or 'Don, please vote for or against something.' But I have always felt comfortable with, 'Don, can you write an exposition on this, or can you check these facts out?' That's what I think I do reasonably well.

Now, of course, there are people who think that my books are completely impossible to understand. I can sympathize with that. When I was a student I would listen to other students and they would say, 'Oh, here is a book by William Feller on probability, it is really hard to understand'. And so I never believed that I would be able to understand a word that Feller said and I didn't read his books until about ten years later. Finally when I had the courage to crack them open, I found a marvelous exposition.

But still, I can't say that my books are easy to understand. All I can say is that they are much easier to understand than what I've worked with. I've gotten closer to a simple presentation, but I haven't maybe got there, to the end. I try to keep out the jargon except where necessary. So I have never used the word "bijection" yet in *The Art*. I could, but why? Someday if I need it, I will, but I don't use scary terminology. I know some people who think words like that are hyper-mathematical. So I boil stuff down and I use a level of mathematics that I have to, where necessary, but I try to translate from the different subcultures that my sources are into another language.

I got a letter yesterday from a guy, he said he doesn't know what I meant when I said 'parity'. And so I realized, you know, I thought everybody knew parity meant the distinction between even or odd. But no, this guy didn't know that and he was motivated really, so I got help from him. But what I am trying to say is, technical writing is what I do well. But as a spokesman for computer science? I am more of a spokesman for computer scient<u>ists</u>.

Frana: Than for computer science.

Knuth: Yes, trying to regurgitate things that my colleagues have done in a way that makes them more memorable or easier to fit together. So this means that I must read stuff from many sources -- like all those folders I showed you upstairs -- and some of it comes from physicists, and some comes from electrical engineers, some comes from the work in AI, some comes from complexity theory, and so on. And I learn their buzzwords, but I don't use them myself unless I have to. That is what I do best.

Frana: Let's hope we are at least ten thousand more days away from this, but have you thought about a fitting epitaph?

Knuth: Epitaph. No. No. No. I am still alive. [laughter] That is interesting, people who wrote their own epitaph. No, I'll let somebody else do that. But maybe it should be an alphametic.

END OF INTERVIEW.