Desktop Publishing Pioneer Meeting:
Day 1 Session 4 - Technology in the 1980s

Moderators by:
Burt Grad
David C. Brock

Editor:
Cheryl Baltes

Recorded May 22, 2017
Mountain View, CA

CHM Reference number: X8209.2017

Table of Contents

Desktop Publishing Meeting: Session 4: Technology in the 1980s

Conducted by Software Industry Special Interest Group

Abstract: Day 1, Session 4 of the Desktop Publishing Pioneer Meeting focuses on desktop publishing technology in the 1980s. The session begins with a discussion of the motivation behind Donald Knuth's development of TeX and the grassroots efforts behind its spread. Steve Kirsch then gives a brief history of FrameMaker and its earlier history and development period. An extended discussion focuses on PostScript, including a synopsis of its main technological innovations, relationship with key industry collaborators (such as Adobe, Apple, and ITC), and the differences between PostScript and competing approaches. The participants also touch on page description languages, various printer and font technologies, PageMaker, and Adobe Illustrator. The session concludes with a discussion of the market factors that led to the shift toward desktop publishing, including changes within the printing and typesetting industries and reductions in the costs of memory and hardware.

Participants:

Name	**Affiliation**
David Brock	Moderator, CHM Center for Software History)
Burton Grad	Moderator, SI SIG
Chuck Bigelow	Typography
Paul Brainerd	Aldus
Liz Crews (nee Bond)	Xerox PARC and Adobe
Chuck Geschke	Xerox PARC and Adobe
Steve Kirsch	FrameMaker
Donald Knuth	TeX
Butler Lampson	Xerox PARC
Lee Lorenzen	Ventura
Martin Ruckert	TeX

John Scull	Apple laser printer
Jonathan Seybold	Rocappi, Seybold Newsletter and Conferences
John Shoch	Xerox PARC
Charles Simonyi	Xerox PARC
Bob Sproull	Xerox PARC
Larry Tesler	Xerox PARC and Apple
John Warnock	Xerox PARC and Adobe
Richard Ying	Atex
Thomas Haigh	Historian, University of Wisconsin/ Milwaukee
Matthew Kirschenbaum	Historian, University of Maryland
Dave Walden	Historian
Ann Hardy	SI SIG
Mike Humphries	SI SIG
Doug Jerger	SI SIG
Ed LaHay	SI SIG
Hansen Hsu	CHM, Historian
John Markoff	CHM
Paul McJones	CHM, Software Preservation Group
Len Shustek	CHM, Chairman
Dag Spicer	CHM, historian
Marc Weber	CHM, Internet curator

Burt Grad: We're going to stay focused for the next two hours or so on the 1980s. If we've left out some technology things that we haven't covered well enough today, we have time during the business discussion sessions on Adobe and Aldus and the other companies, so we can pick up some of those pieces tomorrow. There are background things that I still think are fundamental that we haven't covered. I'd like to talk a bit more about the laser printers and how they evolved because that seems to be so fundamental. We want to talk a bit more about fonts and how they developed, and we want to cover PostScript and PageMaker, Because of a time conflict, I'm going to start completely out of sequence and talk with Steve Kirsch about FrameMaker before the others. I'm going to give Donald Knuth just a few minutes to talk about TeX. You've talked a little bit in your introduction about why you came up with it, but I'd like to talk a little bit more about the technologies, what you did, or what was done to make TeX so productive and so useful for so long.

TeX Technology

Donald Knuth: It had nothing to do with commercial plans. I was in a situation where I already had all my life's goals accomplished, so I didn't need to do something else. But I did want my books to come out looking good, so I researched as well as I could what that would take. I thought that was going to be a more or less private system at first. It seemed to me that the publishing world was being held back because everybody was doing proprietary stuff, and because I'd studied the history of compilers, I thought of TeX as kind of a document compiler. A very important model in my mind was Fortran. IBM had decided that it wasn't going to control Fortran and have it as an exclusive IBM thing; every manufacturer would be able to have their own Fortran compiler. So, I said the publishing industry is being held back because everybody who comes up with a good idea about how to make good books thinks that's going to make their fortune. I didn't need to make a fortune, so I said just figure out what it would be and hope to get cooperation instead of competition. That was very much in my thought. It turned out then that I got volunteers from all around the world to help make this happen.

How did it feed into actual commercial things afterward? Well as you know, some of the people on my project turned out to be very useful in FrameMaker, etc. More than that, it found a real niche in the area of mathematical publishing, which nobody else cared about. It has come to a point where I think more than 95 percent of all mathematical documents have been done in TeX, maybe for the last 30 years.

The other main goal I had was that it would be frozen and not continually improving, so that somebody could write a document and assume that somebody else could get that exact same document 20 or 30 years from now, whatever computers were available then. Those were the kind of things I started out with, and I had wonderful cooperation from everyone. I invited to Stanford not only Chuck Bigelow, but many of the other leading font designers of the time for short visits. I didn't know that fonts were going to be so complicated at first, and as I said, I thought I'd finish the whole project in one year.

In 1985, I did think I had finished it, and we had a celebration at the Computer History Museum in Boston to celebrate the completion of the TeX project. Then, I found out that there were lots and lots of users from the rest of the world who couldn't afford to buy any of these typesetting systems that other people were buying. For example, in Eastern Europe, all kinds of people had started using TeX. People were using TeX in Iran, in Turkey, and eventually in China, everywhere, because it was public domain and they didn't have to pay for a thing. I had TeX users in Ethiopia that weren't able to make their books in Amharic because I had made the system without license. In order to take advantage of that, at the end of the 1980s, I spent another year or two making it so that it would work well with the languages of the world, that you could hyphenate different languages at the same time and things like this.

Grad: I have just a couple of questions. You said everybody uses TeX. Who produces the processors that allow them to do it, to produce the books?

Knuth: It was all user generated. I never went out and advertised it or anything like that; I didn't do anything to promote it. At first people got a thrill out of being able to help port it to other machines. After that, we got more users, and they started realizing they cared about quality typesetting too. I wasn't trying to do something for mass market in any sense. The whole idea of having something where you would distinguish between left quotes and right quotes, M dashes, and N dashes was way out there for most people. I wanted something for the person who wanted to go the extra mile for a book that was as good as anybody could make rather than something that was just good enough.

Grad: People are producing for each individual computer, each individual process. I mean somebody's writing programs to make this language work. It's like with Fortran; every computer manufacturer had a version.

Knuth: I wrote the program myself, and I wrote it in a portable way so that it could be easily converted. I was careful that I didn't use things that would run differently on different computers. For example, I don't use floating-point arithmetic because you couldn't trust an IBM floating point to equal a Honeywell floating point or whatever.

Grad: What language did you write your program in?

Knuth: The original language was Stanford Artificial Intelligence Language (SAIL), but then I converted it to Pascal because I wanted people to read my program. At one point, I think for a program of its size it had more people who understood it than any other program in the world. But then Pascal was an academic language, and it was quickly ported to other languages. The most important thing for me was that I also introduced something called literate programming, which is a way of writing programs that is intended to be read and understood by human beings.

Steve Kirsch: This is a practice that was only used in the dark ages; it's no longer practiced today. <laughter>

Yes, it only has a few hundred thousand users now and a few Academy Awards to show for it.

Dave Walden: Answering Burt's question, today the Pascal web environment goes to C and then goes everywhere else.

Knuth: Right.

Grad: So, C is the current language in which it's been implemented.

Knuth: It's in C Web.

Grad: It's a wonderful story, and for a lot of the people who do this kind of work, it's a small subset; it doesn't have the commercial opportunity or money, but it's so valuable to the whole universe.

Chuck Bigelow: Jonathan Seybold organized a seminar or a meeting at Stanford in 1980, which is where I think I first meet Don Knuth. At that there were several commercial manufacturers of composition software that drove Linotron 202s and VIPs and other things. After the seminar, Don explained what it was and we saw his students do things. Somebody was asking the commercial developers what they thought of it, and they all said, "Well, it isn't fast enough, the coding isn't clear enough, and we do it much faster and we it much better." None of them are now in business.

Grad: Don is not in business, but his stuff succeeded.

Bigelow: His stuff has succeeded way beyond and in far greater numbers of distribution and usage than any of the commercial programs that could handle math at the time.

Jonathan Seybold: I will say that there's another fundamental difference: this was written for the authors to use, where the active writing is done on the terminal. I know a number of you use TeX, and it's a standard format for papers and books. You can deliver it to the publisher in TeX format, and you can write in TeX, LaTeX, or some variation there. This rapidly became not just a publishing system but a sort of authoring-to-publishing ecosystem if you will. LaTeX and this stuff were built around TeX.

FrameMaker Technology

Grad: Although it's a little out of sequence, let's start with FrameMaker and Frame Technologies.

Kirsch: Before I do that, I want to say I wrote my Master's thesis at the Massachusetts Institute of Technology (MIT) in TeX. And I'm sure a lot of people are still using it today to write their theses.

Grad: We'll talk business tomorrow, but today talk technology. Where did you come up with the ideas? What technologies did you develop?

Kirsch: The genesis is that I had a relationship with Sun Microsystems because I started a company called Mouse Systems, and Sun was one of our first customers. I met a guy named John Gage there, who was their chief scientist. And long story short, one day he called me up and said he had run into a guy, Charles Corfield, who was this genius out of New York. Charles was just a really, really smart guy, and he talked his way into a loaner Sun workstation because Sun was always giving out workstations for promising developers. He started developing this prototype desktop publishing system to ostensibly compete with Interleaf, which was much more well-funded and well-staffed at the time. He was just kind of a very smart guy with an idea of building a good publishing system. I started talking with him and was pretty intrigued. I showed it around to people, and they were intrigued. I hooked him up with a guy named David Murray who had developed FileMaker on the Mac and was kind of the user interface (UI) specialist.

Grad: What period are we talking about?

Kirsch: July 1986. Also, there was a guy named Ken Keller who was one of the founders and was very technical, and David Fuchs used to work on TeX. I was more on the business side, so I stayed away from the code. David Murray lived in Morgan Hill, so Charles rented a house in Morgan Hill and David and Charles would essentially code. Charles was a maniac. He could code for 36 hours at a time. Then he would take maybe eight hours of sleep and go another 36 hours—just amazing, physical stamina as well as brilliance. He did the whole thing with David's collaboration. We got the whole thing kind of up and running and it was great. We had this notion of these frames where you could have a block of text; you would put text in it, and you would link this block with that block. We called them frames. You could flow things very easily, so it was very, very flexible in that respect. There was just one slight difficulty in that Charles had never really thought about printing. He had this whole thing designed, and printing was an afterthought. It was like, "Okay, it looks great on the screen, but can you actually get it to print out?" Then Charles got the PostScript book and started reading PostScript. I think it was like two weeks later that they had it printing out. So, it was developed on Sun workstations.

Grad: Who was your market?

Kirsch: It was technical writers.

Bob Sproull: Sun was a big customer.

Kirsch: Yes, Sun was a big customer. Interleaf was quite popular, but Interleaf was a bit harder to use and a lot less flexible in terms of the design layouts that you could have. It was very structured, whereas FrameMaker was the very unstructured version of it. Interleaf was very structured and sort of page oriented. We were in this niche of being able to do very long documents but also have the flexibility of layouts. Tech pubs were our focus. We distributed on these little mag tape things that you would stick into a Sun workstation. They were the 3M cartridges. It was called Prototype FrameMaker 0.6, and we sold the prototype software for $2,500 a pop. That helped to bootstrap the company. We sold software licenses to Toshiba and Digital Equipment later to help fund us, and then we got venture capital somewhat later.

Grad: We'll talk about the royalty arrangements and so forth tomorrow.

Kirsch: Hambrecht & Quist was also an investor.

Grad: Where did the original ideas come from, and who had the idea of doing it this way?

Kirsch: Charles had the idea—just one guy in an apartment in New York had the idea to do desktop publishing.

Grad: Was this a kind of an Interleaf clone?

Kirsch: No, this was a different way of looking at it. It was just, "Hey, here's how I would solve the problem to create a flexible paradigm that could be used to create any sort of documents, whether it's a one-page document or it's a 200-page manual."

Grad: Was the idea of the frame that each thing was moveable or controllable in some way?

Kirsch: Interleaf and other systems at the time for doing longer documents were very structured. They would say, “You're going to have the two-column layout and it's fixed. You can flow the text in two columns, and you can't do anything to break that.” We treated a page as a canvas where you could write on it in very much the same way as Aldus PageMaker, but we combined it with sort of structured and long document thinking. One of the innovations that David Murray had was this thing called Maker Interchange Format (MIF), where we could dump out the document, so it was a portable like an XML version of the document. Instead of putting it out like the content version of PostScript, which is for printers in a printed representation, we had a way to dump it out in a variant form such that it would preserve the semantics of what you're putting into the document itself, not how it was laid out.

Grad: What language did you use?

Kirsch: It was all written in C. Charles just learned how to program in C, so that's how things were done. It was fast.

Grad: Why was it so much faster?

Kirsch: One, it's written in C. Two, it's written by one guy—he knew all the pieces, where everything was connected. He was just brilliant and he could code for enormous stretches at a time, so he's extremely productive and extremely smart.

Grad: Was it difficult to debug? Did you have a long testing period?

Kirsch: Not particularly. People would beat on it, and we'd log bugs but we didn't have a formal bug tracking system. We got it out to people, and they would send in their bugs and we would fix them.

Charles really didn't want people using the prototype software after a certain date, so there was a time bomb in Prototype FrameMaker 0.6. After a certain date, the software would cease to function because it was too much of an embarrassment to have 0.6 out there. It was just like an ego thing that I didn't want to do that. We had like 1,000 people using this 0.6, and the deadline was approaching. We didn't exactly tell people that this was the case, so we had to ship the software and get it out before the 0.6 software exploded on everyone. We had the whole thing working, and everything was going fine, but like two days before the deadline, the software was crashing and we couldn't figure out why. We'd look at the dump files, and it would be complete nonsense. You'll never guess what caused the problem.

Grad: Tell us.

Kirsch: It was a bug in the Sun hardware. It was the last thing you would ever think: we discovered a bug in the Sun hardware where if you took a page fault it would clear a certain register. We just happened to write our code in such a way that it triggered that. We were sweating bullets. Then we found this thing and said, "Whoa, this is a bug in the hardware." We verified with Sun that it was a bug in the hardware, and we wrote a workaround so that it wouldn't cross the page boundaries. We shipped it just in the nick of time.

Grad: Did you charge them more money at that point?

Kirsch: It was like $2,500 a seat, and we pioneered this notion of a floating license where you'd have one license active and you can use it on any machine. It was sort of a number of simultaneous licenses, so it was a pretty innovative concept at the time.

Grad: Did you port it to other machines besides Sun?

Kirsch: Yes. Steve Jobs had us port to the NeXT box. We also ported it to PCs, the Mac, and X Windows. I think we ported it to Sun Windows as well and the new Windows system.

Grad: Did you try to get it ported to the Mac world earlier? Was it a big enough market?

Kirsch: It was just a question of resources. I think the Mac was probably the second one that we ported it to.

Seybold: One distinction I think is vital here between you and Interleaf. Interleaf was the old model where publishing systems were built on proprietary hardware; Interleaf was built on a Sun workstation, but they sold it as a hardware software package.

Kirsch: Right.

Seybold: You were selling software only. That was the new generation, the desktop model.

Kirsch: Yes. We were trying to be new, hip, and cool. Interleaf had these super expensive sales offices and hired really high-priced guys. We were the total opposites, the Silicon Valley version.

Grad: That's a good story.

Kirsch: We didn't bundle it in. We tried to keep the prices low, keep it affordable, with a lot of attention on the user interface. Kevin Lynch was in charge of the Apple product for us, and now he's the chief technology officer (CTO) for Apple and was in charge of the Apple watch.

<u>Early PostScript Development Efforts</u>

John Warnock: I'm going to start talking about PostScript, and Chuck Geschke is going to correct me. This is just the technology story. When we started, we were going to be in a business model very much like Interleaf, and we were going to work on Sun workstations and build a printing architecture to go along with it. But things changed after we talked to Digital Equipment Corporation and other people. DEC had been working for years and years on a document printing protocol that was volumes deep. We talked a lot with Sam Fuller. We also talked a lot to Gordon Bell, and he said, "Gee, we're really interested in the printing protocol."

Grad: Give me a timeframe.

Warnock: It was 1982, just after we started. We started December 1982, and we started implementing first. We were going to do PostScript, so we had a fairly good idea of what we wanted to do, what we wanted to build essentially. Martin Newell and I put together a language called JaM, which stood for John and Martin, at PARC, and it had a full imaging model. We said, "This would be appropriate for driving a printer." So, we started working on that.

Steve Jobs had hired away a whole bunch of people from PARC, and he learned that we had left PARC from Bob Belleville. He called one day and said, "Gee, can I come visit because I hear you guys do interesting stuff?" He came over and we showed him what we were doing. You have to remember the position that Apple was in; they had mimicked the Alto in the sense that it had the same kind of graphic interface, a much less expensive machine, and had a floppy disc. They were about to announce the Macintosh. They had tried with the Lisa, but the Lisa didn't do so well, so he wanted to build a smaller machine that anybody could buy. He built that, had the little bitmap drawing programs, built bitmap fonts, and had the original Mac operating system working. But the only printer that they had was a dot matrix printer that would print out the screen image, which was total crap, and he recognized that there was no bloody way in hell that he could sell that to a business. When he saw what we were doing, he became incredibly enthusiastic, and he offered to buy the company early on. We said no.

Chuck Geschke: We said no.

Warnock: We said no. We worked with Steve, and we talked with the engineers and decided to go into a co-development arrangement with Apple to build the LaserWriter. Dan Putman, who was a very accomplished engineer and board designer, worked with Belleville and crowd to build a board for the LaserWriter. It had 1.5 Mbytes of memory, and it had 0.5 Mbyte of Mask ROM where the code was going to go into. We signed a contract in November 1983. They were going to announce the LaserWriter in 1985 and the Macintosh in 1984. We changed our business model and decided that it was going to be a big enough business opportunity because of the interest in PostScript to build printing software that was going to be totally secret.

PostScript and Font Technology

Warnock: We didn't have fonts, we didn't have anything, so we hired Liz Crews. We went to ITC first and got the ITC contract. Jointly, Chuck and I convinced Steve that he had to have real fonts, Times Roman and Helvetica. He couldn't have Chicago and Monaco because these bitmap fonts were just not going to fly in a business environment. We really pushed hard and worked together, and we got the first Linotype license in the world because when we approached Linotype, we showed them the device independence and he said, "The same thing you see on the screen, the same thing you see on the LaserWriter, you can see on a 1,000 dot per inch (dpi) machine.”

Plate maker had a head of engineering who was very progressive, and he talked to Kummer who was head of Linotype into taking this leap of faith, which was totally against the psychology of the typesetter world. In the typesetter world, "You buy our typesetter, you buy our fonts. My fonts don’t work anywhere else, and the typesetter doesn't take anybody else's fonts." That's the way they all were, so in the typesetter business, you were sort of married to your installed base and you provided the fonts. We said, " Okay, but what's going to happen is all of the ITC fonts are going to be able to run on anybody who builds this RIP, and fonts are going to become interchangeable." This was very disruptive to the typesetter industry.

We started the implementation, and there were some difficult problems. The font problem was probably the most important because if you do as Butler Lampson said, the most straightforward thing, and scan convert a letter and put it on a 300-dpi machine from an outline, it looks terrible. We knew we had to solve it. In the past, people would put the letter outline up on the screen, and they would figure out what bits to turn on and turn off inside of the letter to make the letter look good on the printed page. This was done by hand.

Geschke: For every point size and for every resolution and orientation.

Grad: I've been told that, as it gets bigger or smaller, you can't just scale it up and scale it down.

Warnock: You can't because there's not enough dots in one stroke.

Grad: That is because it was a bitmap.

Warnock: If it had been a 2,000 dpi, it would have been okay. Even at 600 dpi, it would have been okay, but on a 300-dpi printer, if you didn't have these bitmaps, the page looked crappy. This simple fundamental idea, it's so simple, we didn't patent. We kept it as a trade secret because it would be too easy to work around. The basic idea was don't figure out what dots to turn off and turn on, you modify the curves in the letters so they line up with the grid, so the frequency of the verticals in the letter line up with the grid. With a Bézier defined curve, all you have to do is transform the control points; you don't have to transform anything other in the curve, and you don't have to subdivide the curve down at all. All you have to do is transform the control points.

Knuth: There was another idea floating around to move the grid, to distort the grid. There was this company on Page Mill Road that I visited that worked on this.

Warnock: In some sense, they're equivalent, but it's a nonlinear grid. In other words, if you modify the curves, then you can get the x-height and the x-base to be the same because you have what we called blue lines that would change x-heights. Then you had the tube staffs of, say, an M that you would modify by one pixel by the curve that it would make perfect scan conversions.

Butler Lampson: You had to know that the character should be the same width.

Warnock: Yes, correct.

Lampson: That had to be part of the definition.

Warnock: Yes, those were called yellow points. There was a yellow point on every vertical line, and you knew that those had to line up with the grid. We coded this up and tried it, and the results were bloody amazing. It just worked like a charm, but we noticed one other thing: diagonal lines are too heavy. You squeeze the grid on diagonals; you don't squeeze the grid, you squeeze the curve.

Geschke: Squeeze the curve, yes.

Warnock: So, the whole trick in rasterizing fonts is to modify the curve, not the grid, and then all the diagonals show up. If you rotate the letter, none of it matters because it's not beating against the frequency of the grid.

Grad: You say you patented this?

Warnock: No, we didn't patent it.

Grad: Oh, you kept it as a trade secret.

Geschke: Yes, because we would have had to disclose the technology.

Grad: Of course. That was one of the reasons I objected to patents.

Geschke: We figured it's better to give the employees stock options and make them owners, so they won't give it away.

Lee Lorenzen: So the secret never got out?

Warnock: No, not until the 1989 Seybold Conference where we said we'd publish the type one spec.

Commercial PostScript

Grad: When did it get running commercially?

Warnock: This was the technical thing that worked. Now it almost worked on the screen, screen resolutions were so low, so when we got challenged by Microsoft and Apple in 1989 at the Seybold Conference, we went back to Adobe and said, "We're going to out invent the bastards." Bill Paxton figured out how to get the font-scaling technology to work at screen resolution. He not only did that for Latin alphabets, he did it for Japanese and Chinese, which was a tour de force in programming.

Grad: What was the trick?

Warnock: On the screen, you want to look at the rasterization and you want to look for anomalies of bits. He would say, "Gee, if you see this anomaly, kill the bit," because scan conversion at low resolutions had a tendency to be too dense.

Geschke: Basically, throw away bits that looked like they were particles.

Warnock: He would keep the continuity of the letter.

Lampson: There's an amusing story about this in the context of Microsoft. I don't know whether this was a good decision or not, but the Microsoft guys were worried about screen fonts and decided that they needed a great deal of flexibility in controlling things in difficult situations. They decided to allow you to turn the character descriptions into little programs, and the little programs were written in X86 machine language. This turned out later to be a terrible security problem.

Grad: Stop for a second. So, the biggest stumbling block in terms of your getting this to be an operating program was the handling of the fonts?

Lampson: They already knew how to do PostScript, so they didn't have to invent that.

Warnock: There was one other issue. I bow to Don Knuth because what you have to do to build a reliable scan converter for an arbitrary polygon is you have to do it in fixed point, you cannot do it in floating point. There is no bloody way that you can take a floating-point representation of the polygon and reliably scan convert it. That was another thing that we learned, and it's a lot of very difficult, very tight code that does that fixed-point conversion for very long integers.

Grad: What did you write all this code in?

Warnock: All of this was written in C.

Geschke: The other challenge we had is we had only half of a megabyte of space and it would be a Mask ROM, so if there was a bug, you're screwed. We built an escape patch, which we very seldom used, but we could actually go in and put something in the printer in some RAM memory that would allow it to fix the bug.

Warnock: The other largest Mask ROM burn to that period had been done by NASA for the space program. This was a huge amount of ROM, of code to put into a board.

Grad: What machines were you using, and what language were you using to do all this work?

Geschke: We wrote in C. The machine was this specialty board that had a 68,000 processor on it that had 1.5 Mbytes of RAM because the bitmap of the page was 1 Mbyte and then you needed some working space. You couldn't really make the scan conversions of the characters fast enough, so we had to build a font cache in which we sort of remembered the scans.

Warnock: Since it was locked into the raster by changing the curve, you could replicate it to any other raster position, and it would still be good.

Grad: You felt that you would get total portability in that regard?

Warnock: Yes. You could do it at any resolution, and that's how it worked.

Matthew Kirschenbaum: Was there a Bézier curve used? Was there any connection between PostScript and the work that you were doing at Evans and Sutherland earlier?

Warnock: Yes, because Martin and I actually developed the basic language of JaM and the way the stack architecture and everything was done on a PDP 11 at Evans and Sutherland. When we started Adobe, we licensed that technology from Evans and Sutherland.

Kirshenbaum: Was there anything conceptual, was that simulation work that you were doing?

Warnock: Yes, we were building database models for flight simulators.

Kirshenbaum: It was only the programming language that was in common?

Geschke: Only the language.

Warnock: Plus, some knowledge about graphics and mathematics and stuff like that.

Lorenzen: Back to the font breakthrough that you had on rasterization, these blue lines and yellow lines, did you basically shift to the font designer the responsibility to give those hints?

Warnock: No, anybody could do that.

Lorenzen: Anyone could do that once you knew the trick basically?

Warnock: Yes.

Warnock: You would get it from ITC. You would get a font, and all you would do is draw the lines on vertical stems and draw where the x-height is.

Geschke: We hired our children to scan the output from the type, and then we wrote a program to do conversion on the Sun workstations. It worked well enough to give the designer a place to start.

Grad: You had to have pros actually finish it?

Geschke: Someone who was in the profession and who knew about font technology.

Warnock: On the market today, there may be 20,000 fonts.

Grad: How many did you have to add before you felt it was usable?

Warnock: The starter set for the LaserWriter was 35 fonts.

Grad: How many sizes?

Liz Crews: Any size you wanted.

Grad: It was totally flexible?

Warnock: Totally flexible. You can rotate them, you can stretch them, you can do anything you want with them.

Grad: No longer the limit of 10 sizes you mentioned before?

Crews: No, the 10 sizes were for the bitmap fonts.

Kirschenbaum: Today, is a new font still done manually, or do we have tools now that can do all this stuff automatically?

Warnock: I'm sure that they have really simple tools to make this a piece of cake.

Kirschenbaum: Are humans still involved, or can it be done totally automatically?

Warnock: There still are font designers.

David Brock: Font designers deliver it in the format that contains it.

Warnock: Yes.

Seminal PostScript Contribution

Seybold: Let's reinforce how seminal PostScript was. Until this time, every output device had its own language and its own set of fonts, and they were a limited subset of what was possible. They had their own counting systems, so if you wanted to drive as we were doing in the 1960s, if you wanted to drive to different output devices, we actually invented a virtual machine and treated everything as a subset of that. The common practice was not to do that but to write more directly for the individual machines. These were typesetting machines; they could do type, and they couldn't do anything else. This was a huge, huge problem.

When John first contacted me about PostScript, my reaction was either this or something like this has got to be the future, because this is how you get output device independence and resolution independence. Without that we can't really go where we need to go. This was absolutely seminal to everything that happened, and the thing that was really great about it was that you guys can see this not just as type output but as page output, the ability to describe everything that would go on the page. It took us the rest of the decade to catch up with PostScript in terms of all the subject we put on the page. PageMaker couldn't do it. In fact, QuickDraw on the Mac couldn't do what PostScript could do. That problem was only solved when the Mac got PostScript.

We'll talk about it later, but Illustrator, Photoshop, and all those tools came later in the decade. PostScript had the ability to do that stuff from the beginning.

Warnock: And we left design room for color. That was really critical.

Seybold: Yes.

Geschke: We also realized that printing is an international business.

Seybold: Yes, exactly.

Geschke: I began travelling to Japan, and I went to the leading printing company there, Sha-Ken. We wanted to license their library as we had done with Linotype, and I could never get to the second cup of tea with the lady who was running Sha-Ken. Her husband had passed away and she was running it. Turns out at the beginning of World War II, Sha-Ken and Morisawa were the two people running the largest printing business, and they disagreed about the Japanese approach to the war. Sha-Ken who supported the war stayed in Tokyo, and Morisawa went to Osaka. So, we went to Osaka to see the number two guy who also had a very well-renowned library. We eventually convinced him that we could help him take over Sha-Ken's business if he went with us. We got the license agreement and started working with them building Japanese typesetting equipment. In effect we converted PostScript so it could handle Japanese, and they are now the most successful printing company in Japan. It's the only company with whom I've ever had the wife of the owner invited to dinner at a Japanese restaurant, which is a sign of a kind of respect that they have for Adobe.

Thomas Haigh: When you were discussing the challenges with PostScript and how cool it was, the focus was really on the fonts and rasterizing, but of course PostScript is a programming language rather than just a markup language. Did making it a real programming language have any kind of important influence on how it was used or your success?

Warnock: I have had some strong biases in my life. I have never been a fan of declarative protocols, where you declare things rather than program things. Procedures can always bail you out of corners, so that's just the way it is. Having it be a programming language saved our bacon 100 times because Apple not only had their own printer interface, they had their own graphics interface.

Geschke: QuickDraw.

Warnock: The trick is you write a PostScript program that takes QuickDraw and renders it. The Mac didn't have enough memory but the LaserWriter did, so the QuickDraw emulator was done down in the printer, and every customer had their own protocols.

Geschke: Windows.

Warnock: They wanted to take their protocol and translate it so they would either write it in C or they would write it in PostScript so they could send it their protocol and it would print.

I have to tell you where Acrobat came from because PostScript has the property. After we had signed up our 35th or 45th Original Equipment Manufacturer (OEM) for making PostScript printers, we started working with standards, and it started to become a standard for printing. All of the output in the world had PostScript drivers, so you could drive a PostScript printer. Every document produced on Earth has a way of producing PostScript. Now the cool thing is, before we announced the LaserWriter, one of our greatest sample pages was an IRS form that I hand coded in PostScript—all the dashed lines, everything, half tone blocks, the whole page composition was hand done in PostScript. When you ran that on the LaserWriter, it took two and a half minutes to print because there was a lot of execution involved in printing. Steve Jobs said, "No, I want to use this in the introduction of the LaserWriter. What can you do?" Well in PostScript, you can redefine what an operator does. A move operator normally moves the cursor wherever it is to a position, a closed path operator closes it, and a fill path fills it. Those are the basic operators of the graphics machine. If you redefine all those basic operators, they don't do any graphics, they just write themselves out. If you have two coordinates, you write out coordinate one, coordinate two, move to a file. It turns out that that's a linearized PostScript program that has no loops in it, and it has no procedure calls in it; all it has is graphic commands and we called it binding. To make a PDF file, you ran it through a program that linearizes the graphic execution. You can separate the pages, and you can do all the things we love about PDF. That's where PDF came from. Anyway, PDF is our biggest revenue generator.

PostScript vs. Other Approaches

Marc Weber: How was PostScript different from other page description languages? Did Imagen and HP have one at that point? What was the very early laser printer business like?

Warnock: The other printers all had sort of brain-dead protocols that drove them: "Here are my letters, and here's the size of my font. Go down here and put out a line." That's essentially what the HP PCL printers were: "Here's my line. Here's where you put it. Here's the letters that are in it."

Weber: Imagen was the same?

Warnock: Yes.

Seybold: They were more like what had people been doing for years for typesetting command languages. That's the mold that PostScript broke.

Lorenzen: Given that you had worked on Interpress and other similar types of things at Xerox PARC, was there a challenge getting out of Xerox to start this company with the rights to what you needed?

Warnock: Interpress and PostScript have nothing in common other than us.

Lorenzen: Other than the two of you.

Geschke: Yes.

Warnock: Because I had implemented a language that we had originally done at Evans and Sutherland, we were confronted by the Xerox lawyers and we went to meet with them.

Geschke: Confronted is a little strong.

Warnock: Well, they were concerned. They said, " We understand you used JaM," and I said, "Yes, we did." And he said, "Well, JaM is proprietary to Xerox." I said, "First of all, there was a course taught by somebody at PARC, at Stanford that taught JaM so it was divulged." And I said, "The second thing is it's a variant on a language that we developed at Evans and Sutherland, and we have a license to it." The lawyer sort of went white, and that was the end of the discussion.

Geschke: He knew that Xerox did not have a license.

Grad: He just was making sure you were telling the truth.

Knuth: I want to reinforce one of the things that John said about floating point. PostScript is not only a beautifully general programming language, but it also is extremely difficult to implement. If you didn't have an authorized Adobe implementation, chances are that it had bugs because of strange things that can happen with rounding errors and the algorithms for manipulating graphics are extremely tricky and subtle. All kinds of special cases can occur. They do occur once ever few days in the nongenuine application. That was an important thing for their company that they were the only ones who knew how to implement PostScript even though the semantics of PostScript were public knowledge.

Geschke: At one point, about three or four years after the LaserWriter was introduced, someone counted up 70 clone developers of PostScript who were trying to take away our business, and none of them ever succeeded.

Knuth: Inside the machine you had the RIP program that renders the PostScript. Anyway, I had a page once that took two and a half hours to compute.

Warnock: Oh, I've had pages that take that long. <laughs>

Knuth: I have a very strong memory of visiting Xerox PARC to see Butler Lampson, I think maybe John was in the office. I was going down the corridor and I looked inside the office and there was a terminal that had a great big capital letter B on it and they were going around the outside of that B with splines. I said, "That's the way I'm going to do my fonts," because I wanted to typeset my book and I needed fonts. First, I tried the Stanford AI Lab, and I found out that if I tried to use a TV camera to look at a letter then a tiny adjustment in the light would change the width of the stems by 10 or 20 percent. It was a completely impossible to do any quality work. At Xerox they had good cameras, so I asked Xerox, "Could I come visit you and make my fonts by measuring it, by sort of replicating what I thought I'd seen Butler doing?" They said, "Sure, but then we own your fonts." <laughter>

I was just going to get some mathematical numbers out of it. I thought, that's mathematics, but they said, "No, you have to keep that." Since they told me I couldn't do that, I had to design Metafont.

<laughter>

Warnock: The other thing is that there were a whole bunch of competing technologies in the font world. Bitstream and Apple were developing their fonts with segments of circles, so they were second-order curves. Ivan Sutherland taught me about Bézier curves, and Bézier curves have the property that they are wonderful inside of a computer to generate. They're very recursive, they're just wonderful, they're very, very fast. It was an obvious choice to build the entire font library out of Bézier curves.

PostScript, Apple, and Adobe

Crews: I wanted to add another point too. There were no applications when we first launched the Apple LaserWriter. We wanted to show the LaserWriter with PostScript to the publishing world, so we collaborated with International Typeface Corporation to do an announcement in New York the day after the LaserWriter was introduced. Those people expected type specimen sheets, the old-fashioned ones where you showed every size. If PostScript hadn't been a programming language, I could never have generated those type specimen sheets. I wore out the LaserWriter at Adobe trying to produce those because we had to show them exactly what we were talking about, show them that what you see is what you get on this piece of paper that was produced on a low-resolution as well as a high-resolution typesetter.

Seybold: In traditional type design, when you scale a font to different sizes, it's not a linear scale. You want bigger x-height and a sturdier letter at smaller sizes, and you want something more delicate at larger sizes. Could you address that?

Warnock: Yes. We had an idea how you could have one font, essentially one sort of master set of curves and parameterize it in such a way that you could get multiple masters. In other words, from the canonical design, you could automate the production of all the condensed versions. Not necessarily italic versions because those are generally different designs, but Adobe put out a whole bunch of multiple master fonts that were parameterize Béziers.

Hanson Hsu: Earlier you mentioned that in order to compete against TrueType, Apple and Microsoft's collaboration, you innovated your way out of the problem and brought the technology to scale, outline fonts down to screen size. Was that Adobe Type Manager?

Warnock: Yes.

Hsu: What was necessary to turn PostScript into display PostScript for the NeXT?

Warnock: It was essentially just the same code. Essentially, it was the same code that we licensed to Steve.

Grad: What was the timeframe for that?

Warnock: Steve Jobs left Apple in 1985 and founded NeXT, and Steve and Adobe have always had a really good relationship, except when we implemented applications for other machines.

Grad: Are we still talking about the late 1980s?

Warnock: Yes, this is 1985.

Bigelow: I'd like just to fill in a few historical details. One is the debt we owe to French mathematicians, I think Don will know this, but in Hamburg in the 1970s, Peter Karro used software to distort outlines to grids, but it was done offline in order for font manufacturers like Linotype and others to then create their bitmaps. Much of his font library was later used by Adobe. Peter Karro never had the idea of putting this in a machine that would do it on the fly. He called the program PASA because he took one pass to figure out the parameters of distortion and then a second pass to make the bitmaps. He used Hermite form of cubic splines rather than Bézier splines so there was a convergence here, but what he never thought of was to actually put them in the printer, in the raster image processor.

In 1984, Mike Sheridan who was a type manager at Imagen, met with Peter Karro. I was there and Mike said, "Have you ever thought of putting this in the machine so that Imagen could have scalable fonts, outlying fonts?" Peter Karro often said no before he'd thought it through. He said, "No, I want to tell you about font servers," that we could have these servers that would be serving fonts, so the possibility that he would have taken that next step was then dashed.

Five years later he did make a program to do that, and it was in the running of programs that Microsoft was looking at before they adopted TrueType, but it was a different mathematician and a different implementation, but he also liked the cubic splines and the concept of distortions. Karro also developed a predecessor to multiple master technology, and he said that for years he tried to distort a single font to do this properly. He never did until his partner in their business, a high energy physicist said, "Well why don't you just interpolate?" From there, Karro did a number of demonstrations for font manufacturers, and the first typeface to actually use this interpolative method was Matthew Carter's Galliard, which came out in 1978. He flew to Germany and spent many a sleepless night with jet lag as they coded in the interpolations. They digitized in Hermite splines a lightweight and a bold weight, interpolated the weight in between which we called bold and extrapolated black. When Carter looked at the black, he said, "Well, I'm starting over now and I'm going to redraw the black because the extrapolation was not of sufficient quality to make a real typeface that aficionados would like.”

There was this parallel development, but it never took the crucial step to make it in the printer and embed it in a programming page description language.

Half Toning and PostScript

Haigh: You haven't said anything about how PostScript handled the bitmap page elements that were sent to it. Was there anything challenging or innovative in that, or was it relatively straightforward? Pages can have got bitmaps as well as vector graphics and text. Was there anything interesting or challenging in how PostScript dealt with those?

Geschke: One of the biggest challenges with doing photography on the printed page was to do the half toning, and we in fact struggled quite a while with that. I think I mentioned earlier today that my father was a letterpress photo engraver, and I recall bringing home a sample of some of our first half tone work and showing it to him. He took out his loop because photoengravers always have a loop, and he looked and he looked and he said, "Charles, that doesn't look very good." When my dad called me Charles, that was bad news. I said, "I understand Dad but we're going to get it." He said, "Hmm, okay."

About six months later, I knew the rosettes were perfect, in fact better than anything I'd ever seen come off of his proofing press when he was still working. I took it to my dad and said, "Here, Dad, here's another sample." He took his loop out, and he looked and he looked and he looked. With a big smile he said, "Charlie, you finally did it." <laughter>

To me, the high point of my printing career was the fact that I could show something like that to my dad.

Warnock: I think we probably have 30 half toning pens on digital. I don't know the exact number but there's a lot.

Adobe Illustrator Technology

Grad: Are there any other significant new things you developed during the 1980s?

Crews: Oh, yes. Illustrator.

Warnock: Illustrator was a breakthrough product. We said, "Gee, there was a really nice program at PARC called Draw that Patrick Baudelaire did, and he used splines." I was of the opinion that the spline technology didn't give you the kind of control that you needed to do very precise illustrations. You got the curve and you mushed it around until it looked okay. I suggested to Mike Schuster, who was one of our programmers, "Why don't we just take the PostScript imaging model and turn it into an illustration program where you control the Béziers? You have handles on Béziers and you crank them up and down." There's a learning curve to Illustrator, but once people master Illustrator they love it. That's where that came from.

Lampson: PostScript for non-programmers.

Geschke: John's wife is a graphic designer. Once she got the LaserWriter, she wanted to use it, but she wasn't a programmer. John was having to program in PostScript to do her illustrations for her and that motivated Illustrator.

Knuth: I can't resist mentioning that my Metafont program has its own way of rasterizing. My books today are printed with bitmap fonts. My algorithm controls every bit in them, and I send it to PostScript and it gets printed with exactly that bitmap as a type three font. Unfortunately, Adobe has never blessed type three fonts; they put them in sort of kicking and screaming, When I convert to a PDF file, I can't search, I can't find any of the words that I did with the letters that I digitized myself. Therefore, when I do the printed version of my book, I rasterize with the way that I built in. But when we have the PDF version, I have to go to fonts that were designed by somebody outside of my control.

LaserWriter Technology

Grad: We've talked about the laser printers and LaserWriter. Is there a technological story that we should be discussing? You started a little bit ago about the technology that Starkweather brought to that. Were there other major changes that made the use of the LaserWriter effective?

Lampson: The biggest change was that you could afford enough memory; you had a low enough resolution and you could afford enough memory that you could compose, you could have a whole frame buffer. That meant you didn't have to worry about timing issues when you were rendering a page.

Grad: What was different about LaserWriter versus the other stuff that was around?

Warnock: Memory was cheaper.

Lampson: That was the main thing.

Grad: Why?

Lampson: As time goes on, Moore's law says things will get cheaper.

Warnock: When we announced the LaserWriter and Linotype 100, the fact that we could output to a 1,000 or 1,200 dpi machine was mindboggling to people. You can't have a frame buffer for a 1,200-dpi machine; you couldn't buy enough memory to do that. You had to build a display list that you could sort and you could buffer into band buffers and then rasterize out to the Linotype machine. There was a fair amount of technology.

Lampson: Was the Linotype machine asynchronous?

Warnock: Yes.

Lampson: This technology was developed at PARC to run synchronous printers where it's much more difficult because you only have a fixed amount of time to process each band.

Warnock: They had a direct driver motor that was driving the film, so you could stop and start.

Seybold: There's another key thing about the LaserWriter: the problem with xerography was, and anybody who had copiers in the 1960s and 1970s knows this, that with your copier came your serviceman because they break. They had to be fixed and you had to have service for it. Canon took the problem and packaged the things that were going to break into a consumable they could sell like razor blades. That made it possible to make a really low-cost copier that could be sold in mass numbers without having to have a huge field service organization. That was the basis on which the stuff got built.

Grad: I'm still confused. Who owned the technology on the LaserWriter? Was that Apple?

Geschke: The printing device was owned by Canon Technology.

Grad: Isn't that a later point in time?

Warnock: No, Apple bought them from Canon and put their own electronics into it, which included PostScript. We had the intellectual property rights to PostScript and licensed it to Apple.

Grad: In other words, Canon did the first technology work. I thought you were saying that work on the laser printer was done here.

Richard Ying: Canon makes a copier. If you take the copy out of it and drive it with the electronics, then it becomes a printer.

Grad: I have these multipurpose, multifunction machines, and I don't know the difference.

Font Selection

Seybold: The second thing I wanted to say was that the really crucial thing here was the deal with Linotype because Linotype owned the standard font library. There was no question about the quality of Linotype fonts; anything less than that, there would have been a question about commercial quality. To have Linotype and to have a high-resolution Linotype output typesetter as part of the package when it was introduced was absolutely crucial. Even if most people never go beyond laser printers, there is always the possibility to go to a service bureau and get commercial quality, high-quality output from it with the same type fonts that are used by the experts.

Warnock: Steve always gets the credit for appreciating typography because he took a calligraphy course in college. I'm sorry, but I think we were a forcing function.

Geschke: Fair enough.

Crews: We had many typefaces because we wanted to make sure that they had a range of faces that would work for a lot of published applications and not just a limited selection of display typefaces. For example, besides Helvetica and Times Roman, which were probably the most popular, and Chuck can add on to this, there was both sans-serif and serif. We added Palatino and put in ITC American Typewriter, and everybody sort of wondered why. That was because the majority of the people were used to a typewriter look, but we didn't want to put in Courier.

Warnock: It wasn't fixed pitch.

Crews: No, it wasn't fixed-pitch. It wasn't Courier or Roman PS. It was ITC American Typewriter typeface, and it looked like a typewriter face.

Warnock: Courier was there as well.

Crews: But it was beautiful; it was really nice

Seybold: Just one example of how small this world was at this time: the reason Palatino got in there was because you and I liked Palatino. It was that simple.

Lampson: The reason that Times Roman got in is because I chose it at PARC, which was actually not a very good decision. I don't like Helvetica that much, but I do like Times Roman.

Warnock: I disagree. You made the right choice.

Lampson: Really?

Warnock: Absolutely. Times Roman is a universal font. Helvetica is a universal font.

Knuth: It was my impression that Monotype was also one of the world's greatest libraries of fonts in those days. I visited Redhill at the time; at one point he insisted that Optima would never be made digital, and he had a special thing built in to the PostScript font, which I had to actually break into and disable since it could never be used at less than 100 dots per inch resolution.

Seybold: Monotype was the standard for the highest quality output. During the photo typesetting era, Monotype lagged. Monophoto was a really strange machine and did not make photo-touching machines that were competitive in the commercial marketplace. By the time PostScript came along, for most people who were active practitioners in their field, Monotype had become the standard for comparison. Yes, there were particular fonts that always gave problems. You mentioned one, and Souvenir was another one. It had these little delicate curves to it that were absolutely just a bitch to do at low resolutions; you just couldn't do it right without making it look clunky.

Crews: We did put Optima in.

John Schoch: I had a question that's been on my mind for a long time. I remember the people who worked on the 9700. It was all about two pages per minute, super high speed, trying to keep up with the printer multiple pages in flight, so you had to have the image ready because you were launching the paper from the paper tray before you had the image. It was a whole bunch of synchronous problems. The commentary on the LaserWriter was always, "Oh, they can stop between pages," so if they have a really tough problem they just wait. They get the page done, and then they do the next page and restart the printer. The LaserWriter works and that's great. What happens when you try to step up the performance on a PostScript design to a really high-speed synchronous printer with seven pieces of paper in flight?

Lampson: More buffering.

Warnock: Memory is cheap.

Geschke: That’s an answer that wouldn't have worked in the 1980s and certainly not in the 1970s.

Larry Tesler: At one point Gary Starkweather and Xerox were in control of the IP for laser printers I think I heard. Then jump forward and Canon is the company that has the IP. That is the way I interpreted it.

Grad: I thought we agreed nobody here was an expert on this topic.

Tesler: Is that how it concluded?

Seybold: The only thing I know about that was that Canon did patent the single cartridge, which is why its competitors ended up having to break it in two pieces, having two separate cartridges.

Lampson: They missed a bet there. They could easily enough have made a two-cartridge version right and patented that too.

Bigelow: Who chose Century Schoolbook for the LaserWriter 22 set?

Crews: I think it was Allan Haley.

Bigelow: It's important because it links in a completely accidental way to the fact that Ginn and Company originally sponsored the development of Century Schoolbook in 1919.

PageMaker Technology

Paul Brainerd: PageMaker came about because we were all without jobs, as I related before. Eastman Kodak bought Atex, trying to diversify its business model into non-photographic products. Charlie Ying and I went to Rochester, went up to whatever floor it was up on the top, and Charlie was an advisor. They were making some really good technologies. I was running the West Coast plant doing all the microcomputer development including the graphics display terminal that was going to be used for display advertising. They made a decision after acquiring Atex to close down that research facility, which was a little crazy given the fact that we were the future of the income stream. Nonetheless, one day I walked in and they told me that this was going to happen. I laid off 120 people, walked out the door, and never looked back. They gave me the stock that allowed me to start Aldus.

We had 17 engineers working on graphics and typography and composition, and we had all the specifications written for the newspaper industry that I'd had in my mind for years, but now we had to start all over. Our thoughts were that we'd build on top of the microcomputer revolution in terms of the new personal computers coming to market. The problem of course was there wasn't a graphics-oriented machine at that time, there was only the IBM PC, and we were not too excited about the graphics card that came standard in that because there was no bitmap graphics in it.

I had known Jonathan and his father John since Volume 1, Issue 1. I'd been a subscriber, and I talked to John after starting our company in 1984. He said, "I can't say who to talk to, but you need to go to Silicon Valley and set up two meetings: one with John Warnock and the other at Apple Computer with a guy named Bruce Blumberg." I took his advice, signed a nondisclosure agreement, went to Silicon Valley, and met with John in his office. I can still remember sitting in his office; I looked up and he had a page of the Gutenberg Bible there. I said to myself, "Hmm, this guy knows type." <laughter>

"He has respect for type, he has respect for the history of type." I had the same thing in my background in design and newspaper production and typography, so I was impressed that he had a page from the Gutenberg Bible. Then I went over to Apple and of course learned about the Macintosh.

I've always said desktop publishing is based on a three-legged stool. There is PostScript and the technology we've just been talking about, the fundamental breakthrough in resolution and independent output, and license to a large number. Everything up to then was proprietary systems with their own connections to a particular thing, so that was an open standard. Apple had the Macintosh, and we had the application. Because of my background in terms of having worked in the industry and knowing what people wanted to do, we were focused on small short documents. The first version only supported 16 pages, but to integrate text and graphics on a page and put it out to a printer we needed the LaserWriter. Those three legs of the stool all came together because of Jonathan telling us to get together.

I had four engineers that I picked from all this group. The other thing that happened is we went over to Apple Computer in Bellevue, Washington and we explained our idea to a salesperson who worked in corporate sales. Most people we talked to about desktop publishing or the idea of publishing on the Macintosh or on a desktop computer really didn't understand what we were up to, but this guy did. Lo and behold, one afternoon this guy shows up and pulls out two prototype Macintoshes out of the back of the trunk, brings them in, and says, "I think you guys should play with these things." He just showed up. This was not top down from Apple. It wasn't working with John or anything else. It was bottom up from a salesperson in a regional sales office that just thought that we had a good idea. He brought us those two Macs and we were so excited. We finally had a development environment with a bitmap graphics display and a graphics language that we could work with, so we didn't have to build everything from scratch. The GEM operating system and Windows didn't exist yet, and that would have taken us an extra year to get to the application level in order to be able to get the page layouts working. Our goal was the introduction of LaserWriter in 1985, so we had one year to get there. We'll go into the business stuff tomorrow, but mostly all the story is about market, sales, and how we went about rolling this out and building a marketing plan for Apple at a time when they really needed it. We'll deal with that tomorrow.

From a technology point of view, we had these two Macs and the Pascal development environment in the early days running on a Lisa. We only had 64K of memory, which was really not adequate. When the 128 Mac came out, that was really when things got going. Originally it was written in Pascal. That changed I believe with System 7 to C and then to C++, so we were always having to rewrite because the development environment was changing around us.

QuickDraw versus PostScript, there were all these standards issues that we interjected ourselves into from time to time. But our story was more working with the users based on my background. The vision for PageMaker was basically me working with the engineers because I had done pasteup, X-Acto blade, and so forth and I'd work on all the controls. The floating toolbox was the first floating toolbox, and it was really difficult to implement. The whole idea of the controls at the end of the page and having those go up and down, window shades we called them, to indicate where you wanted to end the line of type—all those interactive controls we could build on top of QuickDraw.

<u>ITC Announcement in New York</u>

Brainerd: It took us a year. We demonstrated it at the LaserWriter announcement, and people were just amazed with the output quality coming off. I hired a graphics designer from downtown Seattle named John Anderson. He had been in the business for 20 some years, and we were working with MacPaint, MacDraw, and Word to try to put together pages that looked half decent graphically. We managed to get three pages done and those became the hit at the initial announcement in terms of people in the graphic arts industry, and then we worked with a really top-notch designer to really do the marketing materials that we'll talk about tomorrow. I was also at that ITC announcement in New York. Liz, You were actually in attendance. Do you remember?

Crews: I planned it.

Brainerd: You planned it, but Steve was there.

Crews: Yes, Steve was there. It was important to have John Warnock, Chuck Geschke, and Steve there because of the Apple LaserWriter and also Wolfgang Kummer because he was head of Mergenthaler, and ITC because they ended up being the studio where a lot of people expected to have new type. They were putting out newsletters, etc., and Aaron Burns had the contacts with the publishing industry. We felt at Apple that if we didn't introduce the LaserWriter with its capabilities and its connection with Linotype's output capability to the publishing industry, if they found out about it later just sort of haphazardly, they would never understand what we were accomplishing. That's why we went to ITC to do it.

Brainerd: We went to ITC in New York to this fancy place. I understood the industry, and we knew what was going to happen, so we were trying to navigate and get their endorsement in some kind of way or at least not their outright resistance. We were all working off the script in terms of messaging. Then Steve walks in the door and basically says, "We're going to put you out of business." <laughter>

He didn't stay on script, he didn't do anything. He just said, "This technology is going to put you all out of business." I looked around the room and all these highly esteemed people in printing and publishing were like, "Who's this kid from Silicon Valley telling us about our business?" He just let them have it. I just couldn't believe it. We had many fun stories of Steve. That was definitely not what we had planned that day.

Crews: I don't remember it quite that way. I remember that Steve sat in the corner a lot and was very quiet.

Grad: What was the result of the meeting?

Crews: I think the result of the meeting is we introduced exactly what we wanted to do; we showed them the technology was there, that we were going to make breakthroughs in publishing opportunities. I think that the industry that was most impacted was the middle man, the composition people. I was on the National Composition Association Board for a while, and they realized that they were going to go out of business if they didn't "move on" with this technology. Some of them did, and some of them went out of business. But I think that it was a successful meeting.

Technological Shifts to Desktop Publishing

Seybold: Here's a precursor to that. When the Mac was first introduced the year previous to that, we had arranged to have a demo area for it at the Seybold Seminars in March. We had two ladies from Apple going around the country from the announcement in January; they had spent the subsequent weeks going around the country trying to promote the Macintosh. I'll never forget that they walked in to the Santa Monica meeting just totally bedraggled; they were beaten down because they were getting no response from anybody. This was before the LaserWriter and PageMaker, and suddenly they walked in to a group of people who got it. The reason was that everybody in publishing, the professional type people, had gotten this vision already in the early 1980s of the future of a WYSIWYG workstation. The people understood that's where things were going to go. Here was a baby WYSIWYG workstation, and they could see the potential for that. Right at that point it should have been clear to Apple that this was their market.

Kirshenbaum: One of the questions I had when I was doing my history of word processing was when publishers began accepting electronic submissions from authors. It turned out that it wasn't until much later than you might have thought for many mainstream publishers; it wasn't until the mid-1990s. The reason for this turns out to be that you had an enormous technological base installed at the publishers and they weren't willing to convert to new technologies on any kind of rapid timetable. I'm looking at my old research notes; Simon & Schuster had $6 million worth of Selectric typewriters, and Penguin Putnam didn't get their first computers until 1995. I'm wondering about the sort of disconnect

Grad: One of the things that tomorrow we'll talk about with Aldus is that the market turned out to be different from the market they had expected for things like what you're just describing. I believe that's the case.

Seybold: Here's a note for you. As far as I know, the first author to give a novel manuscript in electronic form to a publisher was John Saul, who wrote horror mystery story type things in 1977. He had written it on a Lanier word processor. After that, all these convention people were talking, and all the other authors wanted to have him talk about word processing. The authors got interested in this stuff fairly early on.

Kirshenbaum: Yes, that timetable is exactly right. That story is in the book as well as other authors from right about that same time, but it took much longer for the publishers to get there.

Grad: I want to ask a stupid question. When did PostScript and PageMaker suddenly made this a viable business proposition?

Brainerd: Up until then if you went to a print shop, the investment required was about $100,000 for a typesetting system with terminals and so forth. We brought that down with the LaserWriter with one or two Macintoshes to $10,000, so that was an order of magnitude reduction. Most importantly, we made it easy enough to use without markup—in other words, as much WYSIWYG as we could get into it at the time so that people didn't have to have specialized training.

Grad: Was the quality anywhere near as good?

Brainerd: The quality was fantastic, as long as you had a printer because of PostScript and the promise of Linotype. When the Linotype 100 came out, we could meet anybody's quality standard at that point.

Seybold: The commercial systems that were being sold for $100,000 were not only expensive, they weren't suitable for the smaller types of jobs that PageMaker was sold for.

Brainerd: We'll talk about this tomorrow, but our customers ended up being office users, newsletters, religious organizations. I got a phone call one day from one of our religious customers in the Midwest and I said, " How many of these are you printing?" They were doing offset printing, but he said, "We're printing 600,000 of these a week." I said, "Oh my God." I didn't think that that was possible, and suddenly I realized that we were really going to have a fundamental revolution here.

Grad: It was a religious experience, is that it?

Haigh: There's a version of the history of Apple that says that Steve Jobs went on a vendetta against the Lisa and insisted on replacing it with the Macintosh, that he deliberately crippled the machine with only 128K of RAM, no hard drive support, no slots, etc. Then only after he was fired and they released the Mac Plus and put back a hard drive (particularly the Mac II) and slots could PageMaker save the company. I wonder if that had never happened, and there'd just been an improved Lisa with a faster processor, is there anything about Lisa that would have been a worse platform than the Mac to have developed PageMaker on?

Brainerd: We never considered the Lisa a viable option for our market.

John Scull: The decision was already made inside of Apple that the Lisa was going away. By the time any of this came about, the Lisa was already gone.

Brainerd: It was dead.

Scull: The Mac was either about to be launched or had been launched, so it's kind of a moot point.

Cost Reductions in Printing Technology

Haigh: But the previous cost point was $100,000, and I know the Lisa price came down to about $5,000 when you consider the cost of the LaserWriter and the system as a whole.

Warnock: Actually, it could have been.

Tesler: Compugraphic developed a software called the Personal Composition System. When they announced it, they made a lot of promises about future versions and what would be in them. They bought millions of dollars of Lisas from Apple that they could resell, so they would resell it with some kind of printer and some of their boxes and software and the starting price was $40,000.

Knuth: Can you clarify what you get for $10,000 versus $100,000? I'm thinking about the quality. I mean if I'm just going to have a Canon laser printer instead of a Linotronic, I don't see how I can consider that to be really publishing.

Brainerd: There was a pathway, so for most of our customers 300 dpi was fine for a newsletter internal publication, although not for all of our customers. Some of them needed higher-quality output but we had the Linotronic 100 with PostScript, and we could provide that whole continuum of quality of output.

Knuth: How much did the Linotronic cost?

Brainerd: The customer didn't buy the Linotronic; the Service Bureau bought that.

Crews: They went to the Service Bureau.

Knuth: Okay, the server would do that.

Lampson: You sent your PostScript file to the Service Bureau to get it printed in high quality after you had iterated 12 times to get it to be the way you wanted on the LaserWriter.

Brainerd: But you're going to get exactly the same output that you got on the LaserWriter, so you didn’t have that proofing process or lose control of the process. Again, it's about saving time, saving money, and having control, the emotional control.

Crews: Remember, the Linotype produced film that you then took to a printer. What you did was create master camera ready art; you could look at your proofing capability of the LaserWriter. It was like camera ready art, so you knew that when you sent it to the typesetter that you were going to get exactly that; it was black and white, and you didn't really have much proofing problems at all.

Knuth: At Stanford we got an AlphaType that cost us $25,000 and then Apps—whatever it was called and I don't remember the cost. I might as well mention that our AlphaType was used by Andy Bechtolsheim to print the circuit boards on the first Sun workstation. He designed his own font to do the typesetting of the circuit board.

Seybold: The APS 5 was a $100,000 machine. That was a breakthrough because previously machines of that sort had cost over $300,000, and this was a different order of magnitude.

Geschke: This may be more appropriate tomorrow, but it sort of fits into this conversation; you shouldn't assume that the Apple sales staff was all that happy with LaserWriter because they couldn't imagine selling a printer that cost two to three times what the computer cost. They were going to rebel. They had a meeting in Hawaii in October and they said, "We can't do this." Steve said, "I don't care what you think, you're going to do it." He put his foot down and they went to market.

Warnock: The economics were really interesting. The parts cost on the Apple LaserWriter was $7,000; it had no margin. Steve was selling it at cost and he had enough. He told everybody in the management meetings that we went to, "The price of memory is going to come down. That will give us margins." So, when it was announced, it was a $7,000 machine that cost $7,000 to build. About two months later the price of memory dropped, and that was the major cost component in the device. You know the guy really deserves his bad reputation, but he also had an enormous amount of vision.

Grad: Good point. Paul, do you want to add anything further to the story on PageMaker?

Brainerd: Again, I think it's more about a marketing and sales, and we'll get that tomorrow. Some really critical things happened right at this break point.

Grad: One technological question. Did you have any competition?

Brainerd: Yes, all kinds. They were coming out of the woodwork.

Grad: Wrong question. From the technology standpoint, was there a different set of alternatives that did what you were doing?

Brainerd: In the software sense, yes, Microsoft in fact had a Macintosh typesetting program. It was developed by a third party, and Microsoft brought it in-house. Guess who the product manager was? Melinda Gates. Melinda wasn't married to Bill at that point, but Melinda was the product manager. It never went anywhere inside of Microsoft, but we were scared to death.

Grad: When did Quark come out?

Brainerd: Quark didn't come in for another year or two.

Lorenzen: You mentioned some of the core developers had just lost their jobs, and you got to pick them up. Who were the core engineers? I know Jeremy Jaech, and then I guess Ted Johnson came later during the Windows version, but who were the first core engineers?

Brainerd: Jeremy Jaech went on to form Visio and then join Microsoft and other companies. Mike Templeman was probably one of our key developers. Dave Walter worked on the user interface, and Mark Sundstrom. Those were the four engineers. We all got in my car and went down Interstate 5 because we thought we were going to sell to newspapers because that's what we already had the specification for. We realized on that trip that the newspapers bought by committee, and their timelines for getting a decision made were typically 18 to 24 months. We didn't have money. If we had waited that long, we'd have been long out of business, so that changed my mindset entirely; we needed to focus on this smaller publishing market and make it as easy to use as possible, really abandon that traditional old typesetting market.

Changes in Market Economics

Sproull: I don't know if closing comments are appropriate, but I just want to echo something John just said. We've talked today a lot about technology, but I think there's something very important that we haven't said much about: there were huge changes in the economics over the period we've been talking about. There'll probably be more opportunities to talk about this tomorrow, but I would urge everybody to mention what things cost, and if you need to normalize it, normalize it by an annual salary of the user or something. The technology was fun and nice, but all of this stuff I think we're talking about today, you could argue was done at the precipice of affordability. Affordability is not just what it cost but what the market you are trying to sell into can afford. The Alto was at the precipice of affordability in some sense. All of these were explorations at that edge and perhaps as things mature, toward the end of tomorrow perhaps, as we all get tired, we'll no longer be at the precipice of affordability, but you cannot talk about this without understanding the incredible, exponential changes in the exponent of the prices. It makes a huge difference.

Grad: Of hardware?

Sproull: Everything.

Lampson: Hardware, actually. Software doesn't get cheaper.

Grad: Yes, the software didn't get a hell of a lot cheaper.

Lampson: It did actually.

Sproull: The whole price model, the whole business model for fonts changed.

Lampson: Software did get a lot cheaper. I mean look at Microsoft Office: the components of it sold for $1,500 in the mid-1980s.

Grad: This is a subject I have never even thought about.

Seybold: The hardware price and the software price are related because as hardware prices go down, the market gets much larger. Therefore, you're selling software in much larger volumes. The whole pricing model changes, so to some extent, the hardware prices drive software prices.

END OF THE INTERVIEW