Desktop Publishing Pioneer Meeting
Day 1 Session 2: Technology in the 1960s

Moderators:
David C. Brock
Burt Grad

Editor:
Cheryl Baltes

Recorded May 22, 2017
Mountain View, CA

CHM Reference number: XXXX.2017

Table of Contents

Desktop Publishing Pioneer Meeting: Session 2: Technology in the 1960s

Conducted by Software Industry Special Interest Group

Abstract: Session 2 of the first day of the Desktop Publishing Pioneer Meeting focuses on the development of printing related software technologies in the 1960s. Discussion topics range from the challenges of desktop applications in the mainframe era to the origin and development of markup languages, such as GML (Generalized Markup Language) and HTML (Hypertext Markup Language). Other subjects were typesetting and printing methodologies including linotype service bureaus. Some of the new digital technologies discussed were the work done by Rocappi, digital typesetting; digital text entry; digital graphics; special purpose printing languages; Engelbart's activities at Stanford; Hypertext; H&J; document content vs appearance

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
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: David and I will be going back and forth as moderators for these sessions. I'm the bad guy, and David is the good guy. He gets to do the 1960s session. So, David it's all yours.

Typesetting and Precursor Technologies

David Brock: Great. When I saw that the subject was Desktop Publishing technology in the 1960s, I thought we'd better look at the 1950s to set up the 1960s, but I'll try and play against type and not do that. Just one note, if you'd like to make a contribution on a particular subject, just catch my eye if you can and I'll keep a running list. We can manage it that way so you'll know that you're in the queue. I thought we could begin maybe by talking about early efforts to use computing in the automation of printing or other public presentations of text. Are there some early noteworthy efforts that really set the stage for what happened in the 1960s? Does anyone have any thoughts on that? John?

John Warnock: I think there is also a graphics component because a lot of the work in Utah was sort of the visual use of computers. It was inspired a lot by Ivan Sutherland, and the whole research that was done under ARPA was about man-machine communication. I think that's intimately related, and graphics are intimately related with the evolution into publishing.

Brock: There's also, I think, a fascinating element in this of what we just call, loosely, the graphical turn in computing, where text becomes a graphical creature, if you will.

Grad: Again, let me show my ignorance. Someone start and tell me if there was any use of computers or other kinds of these technologies by the beginning of the 1960s in terms of producing typographic stuff, printing, any of those things. Were they using any computer-driven stuff by the early 1960s?

Jonathan Seybold: First, there was software written at MIT to do hyphenation, to break text into lines. The first commercial stuff was at the *Los Angeles Times* and the West Palm Beach newspaper in Florida, IBM Computer in Los Angeles, and RCA Computer in Florida. They were basically making line ending decisions for columns of newspaper text. They would get in paper tape from the wire service, and they'd print it out. Then they'd punch it out on paper tape, and it would be set typically on a line caster, on a Linotype machine. The operator would have to make decisions about how to end the line because it was just an endless stream of text. So the computer programs were written to make that decision so you could feed the tape directly into the machine and it would set line by line by line. That was where things stood.

In fact, that was the genesis of what happened or where the germ started for ROCAPPI because my father had gone to the West Palm Beach paper when it first started up to see what happened there. He then had the idea that this was sort of a germ of what you could do to not

just newspaper text but to a whole range of much more complex kinds of things. We actually hired the programmer from that to be our first programmer.

Dave Walden: I just wanted to say, if my memory is correct, in addition to what was happening at MIT very early on, there was work going on at the University of Newcastle, I believe.

Seybold: Newcastle on Tyne.

Walden: Yes.

Grad: Do others have things about the early 1960s?

Donald Knuth: I remember reading in the Penrose Annual about a big conference that was held, I think, like 1966 or 1967, and they were calling it sort of a revolution in the way of typesetting. One of the contributions was by this guy at Newcastle—his name escapes me right now, but it will come to me.

Seybold: By the way, there was a ROCAPPI Limited that was started shortly after ROCAPPI was started in the U.S. They put money into ROCAPPI, and they got the right to have ROCAPPI Limited. The Newcastle stuff was a tie to ROCAPPI Limited because they had been playing around with stuff beforehand. I recall the Newcastle people coming to visit us in 1965.

Content versus Appearance-Driven Documents

Seybold: I'd like to throw something in here that I think is important because we're going to get tangled up in stuff if we don't do it. I've got a handout here about content-driven versus appearance-driven documents because we're going to have different strains of this stuff. The typesetting strain in the 1960s was all text-based, and it was all based on content-driven documents. The crucial, seminal thing that happened with ROCAPPI at the beginning was the realization that if you put something into a computer, you could separate the content from the presentation for structured documents. Anything that had structure for which the computer can automatically format, you could separate the form in the content. You could then manipulate it. You could sort it. You could change it and so forth, and you could present it in different forms for different purposes at different times. And that was the focus from 1963, 1964 on at ROCAPPI, and that was sort of the driving thing that happened through that strain of stuff that happened. That takes you right through the TeX doc stuff that happened. It takes you to SGML (Standard Generalized Markup Language). It takes you to HTML (Hypertext Markup Language).

What didn't happen at that time was work on things that were appearance driven; the appearance and arrangement stuff on the page was important. Usually, these are things that have to grab your attention. You have to be seduced into looking at things that involve heavy use of text and graphics, and you need to arrange stuff. That's where the graphics stuff that John is talking about became important. We didn't have the computer tools to deal with that in the 1960s. We started to see that in the 1970s. It lagged, not because it was less important, but because the technology simply wasn't ready for it yet.

Knuth: I don't know how many other people had the experience that I did, but I was totally unable to do any kind of thing with graphics until maybe 1970. I had already written the first two volumes of *The Art of Computer Programming* before I had any way to make a picture except with a Calcomp plotter.

Chuck Bigelow: Jonathan reminds me that his father, John W. Seybold, wrote an article in a journal called *Journal of Typographic Research* in 1968 called "Esthetic Values in Computerized Photocomposition." It's very much about hyphenation and justification and all of these things. That was just the second year this journal existed, but he mentions in the article that he saw how the hand compositors and the Linotype machine compositors made these aesthetic decisions in how to hyphenate, how much space to put in things. He was confident that soon computerization would be able to reach that level with work.

Seybold: We were doing commercial quality work by 1964. It's the first job of 1964. We were doing commercial stuff for New York publishers in 1965. And yes, what you do when you start out is ask, how was this done by hand? What are the decisions that people made? And you look not only at the Linotype stuff but also Monotype where the higher-quality stuff was being done. You look at what makes really good composition; Donald did the same sort of thing. We were constrained by the fact that we had a 20-K 6-bit word computer. No random access storage. Mag tape and get it out. So if you look at what we did there's a whole series of programs because we were so limited by the size of the computer. But we were getting quality that exceeded the quality that the commercial typesetting people were doing by about 1966 in terms of hyphenation, justification, better spacing, better hyphenation, and so forth. That hurdle we had passed by that time.

The big breakthrough for all of this stuff for us came in 1968 when we got the first cathode ray tube (CRT) typesetter that could measure everything in one-tenth point increments, which means that, if you're trying to do page makeup of a complex page and you've got headings and so forth on it, you've got to space things out. You could, if you chose to, and we did do this, space out between lines in like one-tenth point increments. You had a chance to sort of justify the page vertically and make things fit. That was much more flexibility. This is going to horrify you, but we also had the ability in extremis to actually modify the typography to help with

hyphenation justification. We started using this —for, among other things, science and health. The book for the Christian Scientists was written by Mary Baker Eddy. She had written it and edited the galleys originally very heavily in hot metal, so the spacing was terrible. But it had to be preserved line for line for line because everything was indexed to a page and line number. We were setting this in a different typeface, in a different technology, but we had to maintain those line breaks. What we did was, beside justification, we allowed the ability to change the set width of the type of the line in one-tenth point increments because it was pulled out and stretched in a little bit. I had a copy of that book for years. If you read the thing, you might be able to find it but I've given it to dozens of people and they cannot pick out the lines where this was done because we did it very, very finely. Now, I'm sure whoever designed the typeface would be rolling over in their grave, but it worked. That's, I think, the progress that we're all going to talk about here; you start out by doing what was done with the old technology and you master that. Then you push further and do the things you wished you could do but the old technology didn't let you do it. I think that's going to be a theme we're going to see throughout this.

Paul Brainerd: Do you know what this is? Any idea? [holding up a logic board]

Brock: Solid logic technology.

Brainerd: It's a logic board from a Compugraphic typesetting machine in 1966. I went to the factory and learned how to fix it. It took two weeks to fix. So this is H&J. It was made up of about 30 or 40 of these cards that hyphenate and justify so it's exactly picking up where Jonathan just left off in 1966–67. Compugraphic came along and commercialized a broad market for typesetters by doing it good enough at a tenth or a half of the price of Mergenthaler or Linotype was selling at that time. They commercialized the typesetting machine on photocomposition to a broader market by bringing the price point down and by using this discrete logic because we didn't have program logic at that point. I had to go back to Compugraphic and learn how each of these boards worked. This survived only because it was in a wooden box. You got your spare parts in a wooden box that was as big as a suitcase and that sat in Minneapolis for many years. When they finally got around to throwing it away, someone thought of me because I started up the cold type composition for my college newspaper, and they sent me this board.

So this is one gate of logic, 8 bits that made up part of the hyphenation justification decision on Compugraphic, which was awful. We got these great big long spacing in the lines, and we had to put in discretionary hyphens in order to be able to get it to break more gracefully. It was awful composition at that time. But this is what did it.

Seybold: I just want to say the only way you're going to get acceptable hyphenation in the English language is to use a dictionary. I mean you can do logic, but you've got to use a

dictionary. It can't be syllabification. It's got to be broken on what the proper hyphenation points are in the word, which is not just breaking up syllables.

Bob Sproull: There is another theme here, especially in the 1960s, having to do with whether there were general-purpose imaging devices being available. Some of the early photocomposers were not, in fact, general purpose. They would happily set type shapes from various mechanisms, but they weren't prepared to combine text and graphics. But there were others that largely were for other applications. I'm reminded of III, Information International Inc., founded by Ed Fredkin in 1962. They were building 16,000 × 16,000 CRTs for high-resolution exposure for film. A lot of them were military applications but computer output microfilm, for example, was one of them. They were also imaging at that resolution for things like OCR (optical character recognition). Now, III itself was not particularly in the publishing business, but there were people using their equipment and playing with it in a variety of ways. Ultimately, I think in 1968, they came out with a thing that they really did target for photocomposition but as an imaging device that was a thing called the FR 80. And this was in the computing world where there were computer engineers, as I think somebody alluded to, who had the hubris to think they could generate the images just fine, thank you. FR 80 was a godsend because suddenly you could get really good resolution for whatever it is, the bits you could produce.

The Role of Government Funding

Brock: Interesting that you mentioned military uses of that technology, because one question I had in thinking of computers and automation in dealing with large volumes of text was about the role of the government. I wondered if that's a path in the story, if anyone has any thoughts about that.

Bigelow: The government printing office was the first institution in the United States to install a high-speed cathode ray tube typesetting machine built for them, a sort of special purpose by Linotype and RCA.

Seybold: There was at a government printing office, a Linotron 1010 and there was also one at Wright Patterson Airforce Base.

Brock: When was this?

Seybold: This was in the early 1960s. Now, these were typesetting machines. They were actually a hybrid machine. They had a linotype use stationary grid with photographic images on them for their phototypesetting machines. They had a grid with the images on it, and they basically scanned out the character on the grid and rewrote it on the cathode ray tube. It was basically replacing the optics of a photographic typesetting machine with electronics instead. It

was not digital type and it didn't do graphics. But it was higher speed than you could get by doing mechanical stuff, and that was the point. It was done for high speed, high-volume stuff.

Walden: Perhaps it is not quite relevant to your question, but I'm sure that a lot of the early systems like RUNOFF and things on the TX0 and Whirlwind, even if they weren't the point of the project, were funded under government contracts.

Brock: Does that directly feed into the sort of developments that we were just talking about?

Walden: It hits your question about the military supporting things.

Butler Lampson: It does in the sense that the computer equipment in those days was extremely expensive, and people couldn't afford it usually if they didn't have government contracts. You would sort of nibble around the edges of the government contract and play with the equipment in ways that you wouldn't see written into the contract.

Seybold: Most of the development work that was done by commercial firms, and there was a wall. We applied twice for NSF (National Science Foundation) grants to do multilevel math and were turned down both times because we were "profitmaking." Now, we were running a loss, okay, but we were still a profit-making enterprise. Therefore, we didn't qualify from the grants. They were given to some nonprofit organizations who produced nothing. Except for that sort of stuff, there wasn't any government money. Linotron 1010 was government money, but aside from that, this was all commercial stuff.

Early Graphics and Imaging

Warnock: Early images at Utah were done on an 1108 that communicated to a PDP-8, which drove a CRT. You put a Polaroid camera in front of the CRT and colored filters to make colored images. You would scan one line at a time, and it would take a minute or two to make a color image. But we weren't interested in the speed; we were interested in the technology to make the image and how the image was created. The only displays we had in the 1960s at Utah were line drawing displays. I can't remember—I think it was III or something—you could make line drawings, but they were not interesting from our point of view in that we wanted to make raster images. I think all of your characterizations so far are correct, that I know of.

Brainerd: It's an interesting parallel between the software and the hardware. That's why I wanted to bring this along, but I didn't make my point. So two were paralleling each other; the software people were trying to take advantage of the programmable pieces of it. Until we got rid of the discrete logic, we really couldn't advance to the next level in terms of the speed of

operation and storage and all of the things we needed to produce things like postscripts and graphics.

I was following Ivan Sutherland at Utah way back then. I was interested in what he was doing, but it was so expensive that none of us could afford it. It wasn't until the 6800 came along that we had the address space to do graphics in terms of the personal computer revolution. The Intel 880 really didn't give us that addressing space that we needed in order to be able to start the revolution that became desktop publishing later on. So there's this hardware advance with the software programmers taking advantage of this.

Also, I agree with Jonathan. I think the government research was largely in the research labs, in the ARPA network; that was the fundamental basic research. It wasn't on the commercialization side of it. That was done by private firms.

Seybold: That circuit board reminded me of the RCA 301 computer we had at ROCAPPI. We had this terrible problem that every once in a while it would do something strange and we couldn't figure out what it was. We had the repairman by over and over again, and we couldn't find out why suddenly it would just go crazy. But we learned that if we walked around—this was 19-inch racks—if we walked around to the third rack and kicked it in the back we'd fix the problem. For about a year, we ran that way. Finally, the guy was back there one time, and he said, "Ah ha," and we came around. It turned out that he'd left a screw there, and the screw would flip over and short circuit between two circuit boards. When we banged the back, it would pop back up again.

<group laughter>

Brainerd: Just one more story: at the first West Coast Computer Faire held in San Francisco, Captain Grace Hopper was featured as a fellow of the museum. I had the privilege of going and listening to her way up in the top of the civic center. A little tiny room with 20 or 30 of us using every last inch of space—I was sitting at the door, I remember. She was talking about the origin of debugging—that's why I mentioned that earlier—because the early vacuum tube computers had that same problem with reliability. They'd literally have to get in and dust, debug because they would bring all of this outside air in, and the bugs were getting in and then shorting out the circuits. So Captain Grace Hopper was one of our heroes from that era.

Document Tagging

Brock: In preparing for today, I was very fascinated by this point that Jonathan brought up about the separation of the content from final appearance, if you will, for texts. I also thought that maybe it was the recency effect because Bob and I were doing an oral history with Ivan

Sutherland that reminded me a bit of things in SketchPad for graphics. I wondered if this conceptual move of separating the content, if you will, or the definition of a thing, from its final appearance is something that is common to both the text path and the graphical path that come together in desktop publishing. Is that something that someone in computers would naturally occur to? Is that sort of akin to things that happen in a database? You can have all sorts of reports, but your data remains unchanged. We'd be interested to hear people's thoughts on that.

Seybold: On the tech side, it seemed to me to be blindingly obvious. It also always seemed to be obvious when you talk about line graphics, which you can describe mathematically. Where that breaks down is images, because images typically are captured at a resolution and at a pixel depth. With images, you're starting with something finite. Now, you may desample that. You may change it, and you may manipulate that. But you're still starting with something that does have sort of finite dimensionality to it.

Lampson: Also, it doesn't work for layout. It's been a dream, I think, that's still unfilled that I could take a tag document and do a high-quality layout of it on devices with different form factors and different screen sizes and so forth. Certainly, in the 1960s and 1970s, nobody had a clue how do that. At the very end of the 1970s, we started to work on that problem. And I think people have been working on it ever since, only with different levels of success.

Seybold: Within limits. For example, one of the things that we did was the King James Bible. We had keyed every possible variation of the bible and had a bible database. You could have red letters, not have red letters. You could have diacriticals and not have them. You could have footnotes, annotations, of all of these things. Now, bibles were published on very thin paper and they were published in various sizes. They were done mostly at that time by Monotype in Holland and Ireland. I don't know why. In terms of pricing, I found that there was a nice elastic curve there. If we could set a price that was like half of what it was to Monotype, people would do many versions of the bible. So we did this. At one point, some significant portion of the bibles produced in the U.S. came from that one database. They were all different page sizes. They were different typefaces and everything else. Now, that's a fairly simple document because it's all text. But it's a difficult one to do because you have very short verses, so breaking the columns so that you don't leave an orphan and so forth was tough. But the programs handled that just fine. In fact, it was so reliable that I had taken a chunk of the bible that we could process with different parameters. It would tell me how many pages you would have within half a page over the entire bible. The reason that's important is that people wanted this to come out to an even printing form, so you wanted to know exactly how many pages it was going to make. Within limits you could do that. The problem you're talking about…

Brock: Is that it wasn't too complex.

Seybold: The more complex it gets, the more difficult that becomes. We are still facing that problem.

Markup Languages

Grad: Markup languages, when were the first ones that you're aware of, Larry?

Larry Tesler: RUNOFF, MIT. That was the model for Pub to begin with and for other markup languages that came out around the same time; it was based on an MIT work called RUNOFF.

Seybold: When was that?

Tesler: 1963. It was written by Jerry Saltzer.

Seybold: In 1964 we were using a dollar sign plus a letter as tags. It wasn't a markup language. It basically was a tag that said, "This is a unique element," which means it can be typographically unique and you can define that separately. It also means you could access that for data-processing purposes. You could even have elements that would typographically look the same but would behave differently. We did a map index for National Geographic where you had to sort things with the same name. They had a whole elaborate sequence of places, rivers, cities, towns, and so forth and a sorting sequence. By knowing that this was a river rather than a town, you'd know how to sort the thing. The previous one we did in 1963 was simply typographic. We used a letter beta plus a letter for typographic stuff, but we discovered within six months that wasn't sufficient. It had to move to something that was more general than just typographic. That was very specific to what we were doing. We had to go later in 1965 or so to a dollar sign plus two letters because 26 wasn't enough.

From that, a guy named Bill Tunnicliffe looked at that and understood the separation of form and content. He understood what was happening. He said, "We need to make a language for document exchange." He developed that with a committee of people formed by the Printing Industries of America subgroup that he called GenCode, which was then extended into what became GML, Generalized Markup Language, by three people at IBM. That then became SGML (Standard Generalized Markup Language) as a direct sort of step in that. The difference was that what they were doing was giving the tags names so that it would say what it was, whereas ours was totally generic. If you're going to exchange documents, you'd sort of like what it is. The downside is that the language now becomes much more complex and has syntax and opening/closing parameters and so forth, whereas the simple tag thing didn't require that.

Walden: Jerry Saltzer's memo on RUNOFF mentioned several prior systems that I know had markup in them. I'm trying to remember who the author was—Peter Sampson maybe did one of

them. I'm not sure. If there's a provision in this conference to submit later information, I have the whole sequence written down somewhere, easily available, of the ones up to RUNOFF and then on to the IBM systems that Jonathan mentioned.

Grad: Let me make a comment about that. We transcribe the sessions as they are, but we can put in a variety of ways for people to access additional material either through personal accounts or other contributions. Whatever you have that will help enrich what we can provide to the historians, please do so. We'll take care of that. Okay. I'm sorry. Don.

Knuth: I have a similar remark as Dave. I think one of the important predecessors of RUNOFF was Michael Barnett's work at MIT. He published a book in 1965 called *Computer Typesetting, Experiments and Prospects*." I think Michael claimed, with some justification, that he was in a way the grandfather of computer typesetting.

Seybold: I think what's fascinating here is that once you say I'm going to put something in the computer, and I want to be able to manipulate it in some fashion, then you immediately hit the question of markup to identify things. Everybody hit this and everybody came up with solutions for it because it's the first problem you encounter.

Lampson: I think something that has been partially clear in the discussion so far is the distinction between markup that gives more or less explicit instructions about what you're supposed to do—indent this paragraph by a quarter of an inch, etc.—and markup that tells you what kind of thing it is. The decision about the mapping from what kind of thing it is to what the typographic effects is separated out. I think it would be interesting to hear more about how that distinction evolved. I mean, obviously when you're tagging something as a river you're way over on the "what is it?" side. But I don't know about things.

Seybold: We encountered that right at the beginning. The first jobs ROCAPPI produced were a product directory and a lawyer directory. So immediately you see, "Oh, I've got data elements here." You have to move almost immediately; you move from saying I'm doing typographic markup to saying that I'm doing data element tagging. I have to say that happened within months.

Lampson: By conscious example, there was no such notion in RUNOFF.

Seybold: Yeah. Right.

Lampson: The only kind of tagging had to do explicitly with the layout.

Sproull: I agree. Just speaking from my own personal experience, when Pub started it was by and large the explicit command style, and people wrote macro packages that implemented the style much the way LaTeX become the style package for TeX. Then people like Brian Reed went and made a whole document production system in which styles were centrally the main markup—the only markup you were expected to use as a customer, as a user. The translation from that to actual typesetter stuff was written, if you will, in a separate vernacular. It was the same language, but it was not expected. It was just like LaTeX; you were not expected to delve into LaTeX every time you wanted to write a new little report. At least in the computer world, which is separate, I think, from Jonathan's themes, I think Scribe was the first time that I, speaking for myself only, the first time I saw a cleanly designed, cleanly separated entire system built around that philosophy.

Lampson: By the way, we see this today in HTML, where there's a huge conflict.

HTML pages are just, as Bob says, a huge mishmash because some of the work is being done by saying this is a "paragraph" and there's a separate style definition that says what you should do with it. Some of it is being done by saying put this at this pixel.

Seybold: That's because we are using a language that came out of content-driven documents and trying to produce appearance-driven documents with that language.

Warnock: I hate HTML, for the record.

<group laughter>

Warnock: I think HTML is a piece of crap.

Lampson: HTML is one of the great engineering triumphs of the 20th century. But that's not even consistent with what you're saying.

Seybold: If you understand the heritage from GenCode to GML to SGML to HTML, then you understand exactly where that came from. Tim Berners-Lee did not intend it to be used the way it's being used now. At CERN they were using SGML. The documents that he was producing were SGML documents. HTML was originally a subset of SGML. It didn't get full SGML capabilities until much later on. But that was what he was doing. He wasn't doing the sort of documents you're talking about.

Grad: This is not 1960s.

Seybold: No, this is not.

Warnock: It's a hold over.

Commercial Desktop Publishing

Thomas Haigh: Yeah, I have a question. Computer output microfilm. I don't know if that was the late 1960s or more into the 1970s. My question is, does that in any way relate with the broader publishing story or is it a niche application, a niche technology that just would be used by big corporations so that they don't have to deliver the reports on forklift trucks every week?

.

Tesler: As far as Pub goes, in 1971 it was just for monospace line printer devices. But there was a device command that you could use if somebody took the open-source code of Pub and wrote something to define that device. That was done immediately; I think in 1972 with the FR80. The FR 80 was microfilm addressing the questions.

Seybold: I'm seeing two strands of development here: one was the stuff that came from that heritage, which was the stuff that the computer people were using to publish from computers. The other one was the stuff that's happening commercially. It has always been my assumption, I'd be curious to hear from the people here, that the commercial stuff was the heritage that ended up at desktop publishing, that the stuff that was happening on the computer documentation side was sort of a separate track, but the evolution of desktop publishing came from the commercial side, not from that. Paul, is that…

Brainerd: Yes, that's true. I was aware of it, but I never looked at it because we weren't doing structured documents. Our focus was page layout and integrating the graphics component, plus the text in the markup languages. That whole train of text that we've been talking about on the computer science side really had little or no relevance. I was aware of markup languages because when I started we marked up all of our text and went to those very systems or at least to the people that needed the markup language, but it was the commercial markup like you talked about.

Brock: I want to ask about the place of the minicomputer and to understand these general trends and developments across the 1960s. Does anybody have any comments about the importance of the interrelation of these two things?

Seybold: The early H&J programs when you're just doing hyphenation justification; any other manipulations were run on the smallest computers people could find. IBM 1130 was the dominant one. IBM out of Cleveland had software that ran. Their other company systems integrators in Sacramento wrote software for the 1130. When the PDP-8 came out, it was a natural so people started to write software for the PDP-8 as well. Now, this was just hyphenation justification. This was just doing the simple tasks for breaking lines. Those machines were the dominant machines that were used for that by the end of the decade.

Brainerd: I'll just second that and tell a story about it. My first real computer was the 1130. And then the PDP-8 right after that. We had core memory boards at that point so it was either 4K or 8K, and the board was about this size. I was down pulling these boards, and in those days you did everything. The other thing was the telex machine because it had a paper-tape reader on it, it was commonly available, and it could read eight track paper tape, which is what the publishing industry was using. Combined with a telex machine that could be the reader for the PDP-8, the hyphenation programs were available from a number of sources going back to the 1130.

Grad: The question there seems to be different threads. One is sort of the newspaper thread, what do you do to produce that kind of thing? I was thinking also about newsletters, all of these other limited news extended documents. Were they on anybody's screen during this point in the 1960s?

Seybold: On newsletters, I mean, those sort of documents prior to PageMaker were done with phototypesetting and X-ACTO knives and paste. I mean, you might use, as you got in the 1970s, especially, a phototypesetter that could do hyphenation justification for you. Often they had screens, and you could see stuff on it, but you had monospaced screens. The actual page makeup was done by hand.

Brock: Wax service.

Seybold: Yeah.

Brock: I remember wax.

Knuth: Don't forget rub on letters. Letraset said everybody would have pages and pages of these things to use with their X-ACTO knife.

Warnock: When Letraset visited Adobe and saw Postscript, their life flashed before their eyes.

<group laughter>

Hypertext

Brock: I want to discuss or hear about the story of fonts in the 1960s from a technological perspective. Before we do that, perhaps given what we've been talking about with markup languages and HTML, let's consider that the electronic form of a text might be its ultimately format and ideas of hypertext or collections of electronic documents, hyperlinking. Comments on that perspective?

Seybold: One of the most exciting things to me was when we encountered Ted Nelson, who was then at Brown University, and the hypertext concept. This fit—I mean this was just another dimension to what we were doing in that you could add these links because we were in a world of electronic documents at that point that were abstract in that they were device independent. They were content. They were appearance independent. This was a revolutionary idea, that you could link these things together in an electronic world just seemed to us to be the future. Again, you start out by replicating what was done before, and at some point technology gives you something you could never do before. This was something you could never do before. To use it was an absolute; one of those insights that you encounter and you say, "Oh my God, why didn't I think about that." It is so obvious and so powerful.

Lampson: It was a very exciting idea to people that saw Ted Nelson's hypertext stuff or Doug Engelbart's online system. I think the most interesting aspect of it from our perspective today is it went nowhere for 25 years. People tried it repeatedly, and it never caught on until the Web. I have a clear idea of why it was that the Web succeeded when it did, which is the Internet had to be big enough to make it pay off to make a homepage. But that does not fully explain why the hypertext and NLS (oN-Line System) ideas went nowhere for 25 years. I think it's still a bit of a mystery.

Seybold: They depended upon people being connected; you had to have a bunch of people connected, and you had to have lots of documents. It's a critical mass here. You would have never got to critical mass until the Web.

Lampson: It's a little surprising. I mean there were plenty of organizations in the 1980s that had hundreds of machines connected together with LANs (local area network). You'd think the critical mass would have been big enough. I don't think we really understand why it wasn't.

Knuth: Okay. Ted Nelson had a wonderful idea, but he didn't have wonderful algorithms. His presentation included all links that were two way. Not only would you be able to read somebody else's page, but they had to know that they were reading your page. I don't think there is any

good way to implement that. But he was not an implementer. He hired people to implement his system, Xanadu, and they couldn't figure it out. I believe that's probably a good reason why it didn't thrive.

Lampson: I can understand why Xanadu per se didn't thrive, but that doesn't explain why the idea didn't go anywhere.

Knuth: Two way…

John Scull: It did. It didn't get to the scale until the Internet, but with CD-ROM and the emerging multimedia in the 1980s, not in the 1970s, with Bill Atkinson, HyperCard, and there's a variety of things. I mean I was fortunate to be at Apple and doing desktop publishing. That was the closest thing to media and stuff. Everybody who was doing anything with hypertext were all coming in to show me stuff. That actually led me ultimately to shift careers and get into the multimedia space because it was happening. It was happening all through by 1984 and 1985, even before we were starting to see all sorts of stuff. I know if Alan Kay was here, and heaven forbid if Steve (Jobs) was here, he'd probably have some real strong comments on that as well.

Brainerd: Yes, I was going to pick up on what John was saying. Until we had the hard disk drive, the spinning hard disk drive—before that the board I was taking out of the PDP-8 was a core memory board with 4K memory. It didn't give you a lot of opportunities to do online interactive stuff until we had that big disk and CDC and others commercialized that. Seagate got it down into smaller form factors and we were able to fit it into the personal computer. That's what enabled the hard drive and the CD-ROMs. All of those storage technologies really made a big difference here.

Bigelow: There was an example that was pointed out, that until the Internet was big enough and there were opportunities there, that these hypertext markup languages weren't successful. It reminds me of a story anthropologists report from the Philippines, which had an indigenous writing system. They never educated young people in it. They didn't have to go to school like we did. It was mainly used for writing love letters and love notes, and it said it was marvelous how quickly young people learned this when they came of age and they needed to get a date and meet people of similar interests. I think in the Internet perhaps money is of greater interest in our time, but still the motivation is there that things take off when you really have a desire to do it.

Grad: Let me introduce Marc. Introduce yourself because Marc just joined us late. He was given bad instructions by David as to when he should be here.

<group laughter>

Grad: Marc, talk about your background a little bit and your interest here.

Marc Weber: Sure. I'm Marc Weber. I'm curatorial director of the Internet history program here. I've been studying the history of the Web since 1995, starting at CERN in Geneva with the Web team there. I co-founded the first two organizations in that field. Before I did that I was a computer consultant, mostly software testing, some UI (user interface) and technical marketing and training writing. I actually wrote a few manuals for Canon page-description languages in the late 1980s associated with Luis Trabb-Pardo of Imagen who some of you must know.

In any case, with hypertext I think there are two factors that kept it from scaling. One was that hypertext really comes into its own when you're reading on the screen, when you can follow the link instantly. Until you had a critical mass of people reading on the screen, that just didn't make a lot of sense. The other factor is that the hypertext systems tended to be proprietary, so the people were asked to convert all of their existing documents to something. There were some beautiful ones, but there was a high buy-in factor. The Web took off to a large extent because there was no buy-in factor. It was easy to install. Hypertext proved to be tremendously popular once people didn't have to put much effort into getting it. So those would be the two points I wanted to make.

And sorry, addressing something earlier, HTML was originally meant to be device independent as Jonathan was saying. It was really intended to be used on everything from a voice browser for people with a hearing impairment to small screens, teletypes, whatever. It reacted very badly to getting turned into basically a way to let to layout type. It started with a quite different intention. That was largely because the original Web was meant to have people editing as well as reading. It was meant to be something that everyone would write their own documents in. Once it moved into the professional realm of programmers and graphic designers, that's when they sort of pushed it in a different direction.

Walden: There are many things, I think, that supported the Web. But my view is that the Web let amateurs do navigation. The point-and-click interface let every grandmother and school child do something they couldn't do before, which maybe leads to where suddenly there's a big market for putting up your homepage or your website.

Paul McJones: Speaking to Butler's point about finding an organization that would have had an internal customer base, I think if you go back to the PLATO system at the University of Illinois, it was originally for computer-aided instruction. In fact the students could write their love letters, and there were huge discussion groups. Ray Ozzie was there. And then I think there was VAX Notes, and Lotus Notes was a little bit of the flavor of it, but these were sort of closed communities.

Costs and Complexity of Mainframe Computing

Grad: Question. You all talk in terms of what was needed to make desktop publishing work required a small machine. We had the ability on the large computers in the 1960s to do any of these things pretty much that you wanted to do. Why didn't it really move ahead a lot faster?

Seybold: If you're talking about large computers in the 1960s, you were talking about a ton of money. The machine we had was 20K 6-bit words and mag tape. There was no random access storage. That was $3,500 a month rental in the 1960s.

Grad: Driven by IBM, who could charge you more.

Seybold: Right. We did it through IBM because this was before the 360, first of all. Secondly, a 360 was beyond our budget.

Grad: From 1960 to 1966, the 1130s, the 1401s, all of these things were there and then we had time-sharing. You had the ability to buy a little piece of it. I'm just surprised that with the creative abilities of these people and the needs that some of these things didn't move along. I was working on applications. I didn't do it. It didn't even come to my attention.

Seybold: The thing that I wanted most was not a more powerful computer. The thing I wanted most was a window into that computer. I wanted the visual even for doing text stuff, the visual window into the computer. You could go in and change something, correct something, rather than doing this batch—that was what we really wanted.

Grad: The 2260? We were doing online stuff. We were doing real-time stuff. We were starting to build real-time systems. And yet, I didn't come across it. I was building applications for everything, and this never came to our attention. I was at IBM then.

Brainerd: Well, to answer your question, I agree with Jonathan, one reason it didn't was cost. We're talking an order of magnitude of cost in a computer room with raised floor. I was there with the 1130s and stuff. It was a big deal to maintain that whole infrastructure. You're paying thousands of dollars a month to IBM or whomever to just run it. Second was control. We needed to get the technology to a point where we give the controller back to the customer, the person that's the creative, the window into the computer that John was talking about. We have this intermediary because of the cost and the complexity.

And then the time, because the reason that desktop publishing was a revolution is because of an order of magnitude in tech shift all at once. We had the technology. We had the software.

And we had the control over the process that took so long and was so tedious before. You put those three things together and package it. What I did with PageMaker and desktop publishing, when I came up with the term, was bridge that gap because I understood the technology and what was possible. Then I worked with the software engineers and the programmers to bridge the gap between the two. I was the translator, and the focus was creating that marketing message, which John Scull and others did.

Matthew Kirshenbaum: We've touched on this, but I'd be interested to hear a little bit more about screen technology and how important it was to sort of have that surface where you could see the text but sort of not see it at the same time because it was never going to be the same as what you got on the page. So screen technology, hardware, resolution, how important was that and when did it start to happen?

Brock: Before we turn to that question of screens, I just want to check with Tom and Larry that we're not diverting from the thread we were on before. Maybe we can loop back to screen in a moment, Matthew.

Haigh: I had, I think, a similar question or observation to Matt. Of course, Burt raised this question: why didn't people start desktop publishing with mainframes in the 1960s? Obviously, yeah, one reason is the enormous expense. It's not economically viable. But my impression is that, because of the fundamentally different way in which people interacted with mainframes and even with time-sharing systems, you couldn't get that responsiveness visually of like dragging and dropping, moving things around. The same kind of way that made VisiCalc such a compelling application, the rapid scrolling, the instant responsiveness that you didn't get with say a time-sharing or a mainframe-based financial kind of package. As there are people here who know very much better more about this on the technical side than I do, I think I'd turn that into a question. First, could we imagine a different world in which people did produce highly responsive visual applications on mainframe or time-sharing systems? And second, basically, do you agree that that is an inherent precondition that you couldn't get without having your own processor and a frame buffer?

Brock: Before Butler responds, we're kind of intermixing responses to an earlier question with new questions. So we have screens, kind of interactivity. Larry, you wanted to say something.

Tesler: Yeah, I think a lot of what I heard is likely that there are technologically limitations. It's not like the day they go away all of a sudden there's an infrastructure around or critical mass. There could just be a lot of things going on at the same time, and investors have to make choices. By investors I mean both these VC (venture capital) type people outside of large companies and the executives and boards of large companies. I think there was at least one period of time when the investment community looked at business applications as being the

only way to make money. You're not going to make money on consumer applications because no one is going to pay for anything. In the 1960s and 1970s, if you asked somebody to put their own hard earned money behind this, they'd have said people aren't going to do that. Maybe they would. And maybe we could find a way to get a lot of eyeballs. We'll find a way to pay for it.

Storage Technology

Richard Ying: We talked about quite a few different technology areas and why hypertext did not take off for 25 years. One I think is important but has not been mentioned was the storage technology. Back in the 1960s, you know we took it for granted that everything on a hard drive was readily available. But back then what was available was not really online; we can call it near offline, it's mag tape. You had to actually get somebody to mount that foolish tape for you, and couldn't do that with the hypertext to send a message to somebody to mount that tape. There was a technology called DECtape that came out in the early 1970s that was a random access mag tape that actually divided the tape into small chunks. There was a directory in the front of the end of the tape, but every time you had to access the file you had to spin the tape back to the beginning to get the directory because there was not enough memory to remember the directory structure. So you couldn't mount the big tape in there because it would take you 20 minutes to get to the directory structure. So they have these itty bitty little things. I think that HTTP did not take off because we didn't have cost-effective online real-time data access until much later.

Grad: We were actively building online systems in the late 1960s with these big disk drives. The cost, I think, may be the factor that is key here as well as size. But technologically we could handle speeds, we could handle the complexity. Yet, as I say, I was asked to build out some applications, but this never came to my attention.

Brock: It has to be a very high-volume application to pay for the technology in the 1970s.

Grad: Newspapers are high volume. Newspapers are big money.

Warnock: The UNIVAC FASTTRAN weighed 5,000 pounds and had about 5 Mbytes of storage on it in the 1960s. The IBM 1311 disks were huge, horrendously expensive.

Ying: A disk in the late 1960s was taller than me, six feet high. It was single platter. It had steal bolt holes on the base, mounted on the ground. We didn't know what the bolt holes were for until we were debugging the system. Every morning we came in the disk moved.

<group laughter>

Ying: It was precessing based on the Earth's rotation, and the bolt holes were to lock down so that it doesn't precess. <group laughter> And that was about 20 Mbytes.

Brock: There was a story that Stanford had a fixed disk from Librascope that had so much inertia that if it got loose it could roll all the way up to the top on the hills.

Warnock: The first interactive environment I went into was at PARC with the Alto.

Displays and WYSIWYG Capabilities

Lampson: Yeah, I want to just pick up on this business about time-sharing and mainframes and all of that. One of the reasons the Alto was so successful was that prior to that, getting a display on a mainframe that would give you any semblance of interactivity was either impossible or hideously expensive or extremely special purpose. We tried repeatedly starting in about 1964 and running up until we thought of the Alto to solve this problem, and we never could succeed. I think to the extent that that was important for this application it's a sufficient explanation for why the things that Burt was asking about didn't happen.

Scull: I was actually just going to say I think the people that could probably give us the most insight earliest are the guys at PARC because that's where everything was ripped off from. I mean Steve and others just needed the graphical interface. You needed fonts and the ability to cut and paste graphics and fonts. You needed all of these fundamental things. You needed a laser printer. You needed to have a way to describe it so that it went from what you saw on the screen to what came out of the printer (i.e., Postscript). You had all of these enablers. You needed to have some kind of communication between them. If you didn't have all of those building blocks, only after that could you have the smarts about which was the software that took advantage of all of that, to actually have applications be it page layout or more—you know, TeX doc or some other or whether it was presentation material. Then all of a sudden you had everything. Until the foundation was laid, it's no surprise. It just happened and it was around 1985. It's magically all boom, the big bang theory, they all came about. You had a lot of really cool things happen.

Kirsch: Yes, I'll give you my perspective. I was a student at MIT in 1974. Shortly thereafter—I'm not sure what the date was—I remember seeing the manual with Butler Lampson's name on it because MIT got these things called Altos. Even though we had all of this computing, the Alto was really transformative because not only did we have Altos but we also had this thing called a Dover printer. The Dover printer could actually print out the stuff that you were creating on the Alto, and it was all WYSIWYG [What You See Is What You Get]. To me, I think that was the transformative moment; that's where things could really start to take off because you had access. It wasn't that the machines were expensive. It was the access. You could have these

machines, but you'd have to have reserved time on a terminal. The terminals were line by line or they were text terminals and not WYSIWIG displays and so forth. To me, I really think the thing that was transformative, in my view, was the Alto plus the Dover printer; you could actually create these documents and share them. Of course, we had networking as well at the time, so those things all coming together is when things really started to take off.

Chuck Geschke: In 1972, I was writing my thesis at Carnegie Mellon. I believe I was the first person to do an electronic thesis generation. I did not use a typewriter. It was some software running on a PDP-8 with a DECtape for memory and hooked up to a Xerox XPG which…

Seybold: I had all of the right ones, just the wrong order.

Geschke: An XGP, a Xerox Graphics Printer. Because I had some mathematical-like symbols and the font didn't have them, I actually went off to a line graphic display and was able to generate my own characters and add them to the font. I produced my thesis. The pieces were just starting to come together. It was possible to do those kinds of thing just like we were doing with the Alto at MIT, and we were doing it at PARC.

Knuth: Can we access your thesis on the Net now?

Geschke: I made the mistake of never copying it from the DECtape when there were still DECtapes around. I can do it for you but I'll have to…

Knuth: I mean CMU (Carnegie Mellon University) probably has university microfilms, and all thesis were scanned.

Geschke: I can just scan it for you.

Knuth: Okay. I mean I think for the world, not for me personally.

Weber: I'm sure the Computer History Museum can read the DECtape. Do you still have it?

Geschke: No.

Printer Technology

Knuth: But I was going to talk about the XGP (Xerox graphics printer) also. I mentioned earlier the Penrose Annual. The Xerox graphics printer actually came out of Rank Xerox, another division of Xerox in England that had invented this device. It was digital internally but the original

device scanned and then reproduced xerographically, but it didn't have digital input. I don't know exactly how it happened, but somebody in America added this extra thing that the computer could supply the rest of the bit padding. Now, because the thing was so funnily designed, it had different resolution. At the edge of the page, the resolution was about 100 dots per inch. In the middle, it was about 250 dots per inch.

Grad: That's not true.

Knuth: The other way?

<group laughter>

Lampson: It read the same resolution across the page.

Knuth: No, it didn't.

Lampson: Well, you had a broken one then.

<group laughter>

Knuth: You could print the letter A over here, and you could print the letter A over here, and it would be different. Our eyes adjusted it.

Lampson: Run by the specter of two, that's what you're telling us.

Knuth: Okay. Maybe it's a factor of…

Lampson: Maybe it was 10%.

Knuth: No, it's more than 20%. I swear.

<group laughter>

Knuth: Anyway, the Xerox graphics printer was really a revelation. Dover was very consistent and much better later on, but that was several years later on. I know the XGP was important because every once in a while the XGP at Stanford's AR lab would go down. On those days the parking lot would be less than half full.

<group laughter>

Lampson: The XPG was originally built as a high-speed fax machine. The reason that it was available for your applications was that it was a complete commercial failure as a high-speed fax machine, mainly because the high-speed phone lines were much too expensive.

John Schoch: It's the LDX (long-distance Xerox); you can look it up and find pictures of it. Two refrigerator size units. One was a scanner. One was the printer. It required a leased line in between. It had a scanning CRT. It had role-fed paper. It was sort of an abomination. That was an early fax machine, LDX, but then someone said we'll just take the output side, put a digital input on it, and all of a sudden it becomes a laser printer but originally a CRT printer.

Knuth: I still insist that it had variable resolution, but that didn't make any difference for the fax machine application.

Lampson: To the parking lot.

Grad: I take the privilege of age, again. Look, this is very fascinating because we've taken about an extra half hour on the 1960s, which basically you've all told me really had no significance at all going forward. It was a total waste of time. You're all younger than I am, and I'm still saying that the 1960s did provide something and that's where I'd like to end up this session. We'll start the next session here and think about this question as we grab lunch: What were the key elements that came out of the 1960s that underlay what happened then at Xerox PARC and elsewhere during the 1970s? We're going to start our next session with that theme.