The Final Demonstration of the Xerox 'Star' Computer, 1981

Recorded: June 17, 1998
Palo Alto, California

CHM Reference number: 102737965

Peter Nurkse: I think we're about ready to begin. Seating has sorted itself out. I'd like to welcome you. I'm Peter Nurkse and I'm from Sun Microsystems, and with Jeanie Treichel, also of Sun, we've had a computer history talk series for the last five years. Our very first program, five years ago, back in April, 1993, was right here in this auditorium, on early computing at Livermore. And the total number of people present, including the speakers and the entire audience, was twenty. [Laughter] But this was a very distinguished audience, because right there in the front row, in two seats in the center, we had Gordon Bell and Donald Knuth, sitting side by side, sort of like hardware and software together. [Laughter]

So we've had five years of programs since then. In the last couple of years we've joined forces with the Computer History Museum at Moffett Field, which has the world's largest collection of computer hardware, over 100 tons of computer hardware. [Laughter]

I might say that our second program, after that distinguished first program, was on the 20th birthday of the Ethernet, back in May of 1993, again a PARC occasion, although we held it at Sun, that program. John Shoch was one of the speakers, who also suggested this program here tonight. And since then, looking back over our programs, a couple of more particularly relevant to this evening: just last fall, October 28th, we had a program on the early user interface design at Apple, with Larry Tesler and Chris Espinosa, which directly drew on a lot of the work done at PARC that you'll see in the Star user interface this evening. And our November program last year, that was on Doug Engelbart and his writing on augmenting human intellect. And there we had Bill English, who worked with Doug on early input devices. And he had a whole series of early mice that he pulled out of his briefcase, one after the other. Some were plastic, some were wood, some were metal -- a whole profusion of early mice, which we'll also see in the Star user interface. And remember, the Star comes from 1981, but just that recently, 1981, people weren't sure how to describe a mouse. It was a digital pointing device. They didn't have any better words for it.

About the program tonight: obviously the main center of interest is the Star itself, although there are two Stars running, just in case one falters. [Laughter] I'd like to make a point about the hardware. In my announcement I wrote how Dave Curbow, from Sun, had like twelve [6085] Stars, from which he was making one or two run. But those were a later model of the Star. We're lucky enough for this program tonight to have the original Star, the Xerox 8010, running. And we're even luckier that we have the original software running on it. The [password] code to activate the software was only found by Bob Garner two weeks ago. [Laughter] So, there I think I've said enough about the series, and more or less the main topic of the program, the Star, so I'll ask Dave Liddle to speak now. Dave directed the Star development effort at Xerox. [Applause]

Dave Liddle: How's that? Not that I couldn't make the place resonate entirely unamplified. [Laughter]

There's something very painful about this experience to me, which is that I'm seeing all these people that I haven't seen for many years, and you are unchanged except for a little pigment here and there. And, recognizing that that was going to be true, I promised myself that I was not going to name names, in my discussion, precisely because I'd have to leave people out, and I can't bring myself to do that. So I will name really only one prominent name in the course of what I'm going to tell you about. But I want you to know that I'm just going to try to set the context, and it is a considerable and complicated one, for the development of Star, before my talented colleagues, who really did the hard work, tell you what we did and how we got there. But I'm going to just try and help you understand really what that world was like, the one in which Xerox decided to participate in these new industries and in which, and for which, we developed the Star, and the related network services, and so on.

There's been a wonderful -- what do you call it, when there's more than two people in a dialogue, is it a polylogue, or a multilogue, or something -- anyway, there's been a -- actually, more like a cacophony of e-mail discussion about all the firsts and inventions and new ideas and whether they were or weren't new, or for the first time, but it's really been invigorating. And so a lot of us had an opportunity to participate in that. I'm not going to try and identify those things, but some of you may pick them out as we go along, a little bit later.

But first of all, I want to talk about PARC. Not this PARC that we're in now, at all. The same location for part of the time, but this has nothing to do with Xerox Palo Alto Research Center as it operates today. But just a very small window of its long and glorious history, from 1970 to 1975, roughly. And of course I'm only going to talk very briefly about issues directly tied to certain kinds of information processing and computing problems.

But having said that, I want to mention some very strong, major high points that affected a great deal of what we did. First of all, there was a famous talk given by C. Peter McColough, who was then the much-respected Chairman and CEO of Xerox. In this famous and much publicized talk, he said that he wanted, for the future, for Xerox to be the architects of information. This was a great phrase, because nobody knew exactly what it meant, and there were quite a few interesting things that you could do and simply cite that as the justification. [Laughter] But it also made Wall Street feel a good deal better about Xerox's future prospects, to realize that they weren't just focusing on a sort of reprographic copying, duplicating, and so on.

There was already the early work that had been done by Gary Starkweather and others, leading to this thing that we then called SLOT, the Scanned Laser Output Terminal, which grew up to be laser printing of the kind we know today. And this was very cool, because we really could print very, very interesting documents with it, but it was really hell to produce those documents at that time. There was the splendid project called POLOS, the PARC On-line Office System. I came there first to work mostly on that project, and to help out with the Learning Research Group a little bit. The really nice thing about this was for the first time, we saw how very, very attractive office equipment designed for professional users, rather than for clerical users, could be. High resolution displays with beautiful, carefully rendered fonts; pointing with a mouse; having a really quality keyboard, and so on, and the ability to print the kinds of documents that otherwise we had to get a paper accepted by a publication ever to see our words rendered so beautifully. And now, even if it was entire rubbish, we could see it rendered beautifully [laughter] with these wonderful fonts and so on.

There was a great deal of work at that time, sort of very turgid interesting work going on in the Learning Research Group, which among other things led to Smalltalk and a somewhat different way of thinking about using screen real estate as well as programming. And in 1973 the Alto and the Ethernet independently were invented, but they began to work together relatively soon thereafter. The Alto back then was a prototype for a pad-sized computer, and Chuck Thacker promised us that it would only take a few years before it went from being a featureless cube, 30 inches on a side, to being this pad type computer. The functionality of it, though, was very much what was wanted. There was a lot of other creative ferment going on. I won't bore you with the details, but we had people interested in networking, and protocols, and various programming techniques, and very advanced notions in operating system security, and memory protection, and other things.

In the midst of all this, in January of 1975, my friend and principal mentor Harold Hall stepped up and agreed to start an activity to try to harvest these ideas into a long-term architecture for Xerox's entry into the new world of the automated office. So, he was the one willing to face up to the corporation's sort of mixed and tentative interest in doing this, and saying, "Yeah, we can do it, Palo Alto is a place where we can get started and put together that story in an understandable way, and harness some of these good ideas that have already been developed here in PARC."

Now, between 1975 and 1980, the time period of interest when all of the development was being done, things were very different than they are today. You can't extrapolate backwards from where we are now, and get there. There's been a very big discontinuity in many dimensions that's occurred since that time. Minicomputers were all the rage. If you couldn't do it with a minicomputer, it probably wasn't worth doing, that's what most of industry thought, and even IBM had begun to package things into minicomputer sized chunks for various sorts of reasons. The very interesting and cutting-edge companies were the minicomputer companies at that time. And the idea of the interaction devices that were going to be in front of the users was that they were going to go from being dumb terminals, which at that time still predominantly were teletypes and paper based terminals -- it was a great luxury in that world to have an

actual screen on your terminal, and there were a few intelligent terminals, intelligence of course being a relative term, but it did mean that some of them had a little bit of memory, and could be made to do mildly interesting, sort of quasi-computational things.

But the hot ticket in those days was electromechanical word processing, or "clickety-bang" word processing, as we used to call it back then. IBM was the big lion in this marketplace, and it was one hell of a big business, and they were also challenged by various other people. By the way, if you don't know this, the term "word processing" was made up by the IBM Deutschland Engineering Group in Germany, to be sort of a parallel with data processing. I'd love to know what that German word for word processing was, I'm sure it's a real tour de force of linguistic construction. But in any case, this was then a big business, but it meant, in general, a typewriter imbued with a modest amount of storage and logic, and you could somewhat painfully edit a document, save it onto a magnetic tape or a little card. This was a very big business. The big threat at that time to IBM's dominance in this business was Xerox, with a division in Dallas, which, because they possessed a daisywheel printer from Diablo, across the Bay, a company Xerox had bought, had actually introduced quite a nice high performance electromechanical standalone word processor.

There was no independent software industry. Software was written by the customers, say, Bank of America or something like that, or by the vendor of a mainframe. But not by anyone else. If you ever wondered where the "I" of "ISV" comes from, when the first companies finally started writing software, companies that weren't part of a mainframe company, this was so shocking that a new category was made up and they were called ISV's. But there was no such thing, during this period of time.

The future of office systems was imagined by the pundits of that time to be entire turnkey closed vendor office systems. Burroughs had one, Wang had one, everybody was sort of hovering around this idea of a marketplace. And in particular, there was no personal computer industry whatsoever. Near the end of this era, a few personal computers appeared, as you know, kits and then the Apple, completed packages and then a few others, but there was no really strong industry, it was essentially a hobbyist business at that time.

Inside Xerox, this is very important to understand, that is, this was the climate in which we operated in this period, from 1975 to 1980. Xerox had entered, and then exited, the computer business; that is, they had bought the leading minicomputer company of the time, Scientific Data Systems, which was a competitor with Digital. And at the time they bought it, it produced primarily real-time, shared, very high quality, minicomputer systems. During the era that it was a part of the Xerox Corporation, it made a turn in the direction of providing business data processing systems, so the Sigma Series began to be sold competitively against IBM, rather than, for instance, competitively against Digital, as had the 930 and 940, and other famous systems been sold. This turned out to be a very tough business to participate in, and in 1975 Xerox somewhat painfully left this business, selling its remaining interests to Honeywell, but leaving behind, luckily for us, a large number of really talented people in El Segundo, a very strong capability in electronics and certain kinds of system software, and a factory, and gave us suddenly sort of a base, a place in which this kind of work could be done.

Xerox at that time was by far the largest vendor in the facsimile business. But Xerox was, by the middle of this era, a five or six billion-dollar company, and the fax business was a very small, few hundred million dollar, and business. And not very digital in its nature. The fax business we know today was almost completely restarted from scratch by the revolution in digital logic. Xerox had gone from really dominating the copier and duplicator world to suddenly being under a great deal of very stiff competition from very, very tough competitors, including Kodak and IBM and Ricoh and Canon. And some of Xerox's longtime patent protection had expired. Somewhat worse than that was, Xerox had to agree to an IBM-like consent decree with the Justice Department limiting a number of things that it could do, both as to pricing but also as to competitive statements it was able to make in the marketplace and so on. So suddenly we had to go to this very sort of Marquess of Queensbury, dignified, uncompetitive behavior in the sales force, which was not an easy adjustment for the company to make at that time.

We had a word processing operation, largely in Dallas as I had mentioned. It was troubled. It had a hard time bringing it to profitability. They had a hard time focusing it down as a business, and there was much energy going into what will we do to fix that operation. The Printing Systems Division, which by the end of this era had begun to build laser printers based on much of the work done here in PARC, particularly work that used very clever character generation technology, and other things. It was very focused on selling computer printers to MIS departments, that is, big, really beautiful printers that were hooked up to great big mainframes, and produced two pages per second of very high quality output. That was the business that we were in. Not anything to do with casual office laser printing and so on.

At this time Xerox was the perhaps number two or three largest participant in magnetic storage. They owned Shugart, which pretty much owned the floppy disk business, and had a very strong position in the sort of Winchester disk business, and they owned Century Data Systems, one of the big players in packaged disk drives. So there were a lot of things that were true of Xerox at that time, and many problems to be worked on, or itches to be scratched, and they simply don't exist now. This list of stuff that I'm showing you here is -- that was the big deal. Those were the things that we were actually focusing on, and that were affecting our choices about what businesses that we were going to be in. Ever heard of any of them before? I doubt it.

Now, when we started this organization, we moved around Xerox. A lot. In 1975 Harold Hall and I, and then Gail Tofani, and then a small growing group of other people, were part of the Information Technology Group's OIS project. By 1976 a quote, division, had been started, although it wasn't really the size of a typical division, called the System Development Division. By 1978 it had been shuffled to be put together with some other loosely information-related products in order mainly to try to hold on to one particularly self-centered executive, who, after they reorganized the corporation to keep him, wound up leaving to run Amdahl anyway. And then in 1980 we were kind of snatched from the jaws of death by Don Massaro, who still thought it was a good idea even though other people didn't, and that it deserved in some way to see the light of day, and so we were lashed together with the Dallas office products division. And were able finally to get our products built and into the marketplace. It was a case where the product and the ideas were so strong that even though we never could quite have a good home for them, it's also true that no one wanted to stop them in their tracks, because they had an obvious merit and an obvious value even though they were an imperfect match with most of the business objectives of the corporation.

What were we supposed to be trying to do? We were trying to do an overall system architecture that would last ten years. With its first implementation being to support electronic copying and printing. And then, for professionals, who were doing document creation and transmission and storage and so on. And then finally, as a long term overall successor business, when people would no longer be allowed to cut down trees, and there wouldn't be a paper-based office automation industry any more.

In 1975, at Harold Hall's urging, I wrote a document called "The OIS Architecture." This described these essentially seven points - there was more detail, of course, but I won't put you through all that. The main point, though, was in the parentheses you see what the ultimate implementation was, that we finally did, of these particular points that were in this architectural document. That, with some pain, we got people to agree, was a good long-term approach for us to take, so the idea was we would have a packet-oriented local area network. We had already had the three-megabit Ethernet as its inspiration, and this was to be named the Xerox Wire, and eventually we renamed it Ethernet because that was a better name.

No one used the term "client/server" back then, but on the ARPANET there was the concept of user and server. Everything was hosts, of course, but you had a using host with a bunch of terminals, and a serving host. We worked out the way to have that host be a workstation belonging to one person, i.e. a client as it's now called. And of course a structure of a number of servers and server protocols.

A scalable CPU architecture, in the old sense of the word architecture. Now that there are so many people who do architecture that don't know very much about it, [laughter] we talk about architecture as if it were design, like the way it is in a building. But in those days architecture meant not how the box is put together; that's what it did not mean. It meant everything but that, namely, the instruction set and the

things that happened when you executed them, no matter how you put the box together, no matter what the underlying technology was, or how big or small any register was, or anything like that. It was the implementation-independent design, which we did in a document called "The Principles of Operation."

Operating system research had just begun to go from the baroque to the rococo at this time, [laughter] and we managed to avoid the worst excesses of operating systems design, but we did design a very, very modern operating system, and quite sophisticated for its day. Now, the fact that the VM in there stands for virtual memory shows you what the world was like back then. Memory was still relatively scarce, even though we were thought to be the most reckless and profligate people in the world for having 512K bytes, when we went out the door. Whew! And when you heard people talking about 192K machines, those were 16-bit words, and we couldn't quite get it in 192K, but I didn't let the pressure off until the last minute, fearing that it would get even bigger.

The development environment at that time was also a modern and sophisticated one that we were very proud of, and it was clearly far ahead of what had been done up until that time. Bear in mind that I'm not naming any of the things that happened after this time. I'm just talking about the stuff that we described in 1975 and shipped by 1981. The servers that were out there were print and file and mail and what today we would sort of call some combination of routers and bridges and gateways, our communications server stuff, including inter-networking.

And then finally, this idea of the Star, the 8010 Professional Workstation. Why the Star, itself? Why, particularly, that last piece? Here's what we were trying to do. I tell you this to put in context what you are going to see demonstrated. At this time, we were really trying to make the first product that professional users would actually like, because it added more to their life than just giving them a way to get memos typed, and so on. And in particular, we wanted to overcome the idea that this was somehow an unacceptable thing for professionals to do, by making a really useable and really powerful workstation. Naturally, what we hoped they would then do was to create these elaborate, kinds of self-serving documents like we did in PARC. We had some beautiful typesetting for equations, for example, because if you got people to typeset their equations, then you knew for sure they couldn't print this on anything but one of our laser printers. [Laughter] And that was the whole idea: we wanted to induce people to produce subtle and elaborate documents that would then favor our ability to make money by storing and managing and printing them.

And finally, we wanted the brand named Xerox not to just be in the copier rooms, and seen by purchasing agents and people who were responsible for duplication; we wanted to get it onto the desks of management in the Fortune 500.

So, these are the things that we were trying to do at this time. This is what the Star was about. Now, Bob Belleville and Bob Garner are going to tell you in more detail about the hardware design aspects of the Star. [Applause]

Bob Belleville: Personal interactive software like Star was unknown outside the research facilities in 1978, the year Dandelion was started. Apple Macintosh began to catch up, perhaps, to Star's capability about a decade later. Microsoft Windows took almost 15 years. Inside Xerox PARC, however, we were quite used to this level of creature comfort. At SDD, a Xerox product group spiritually attached to PARC, almost everybody had an Alto. First built here at PARC in 1973, Altos furnished most of the computational horsepower for our culture. We had e-mail, and FTP all over the world via the ARPANET, computer-aided impressive office automation and software development tools, and computer-aided design facilities that took us from schematic to prototype nearly automatically. Our mission at SDD was to take some or all of this to the world, except we were going to redo it all from scratch.

Alto was mature, and there were about a thousand machines in use in 1979. Basically a 16-bit machine, Alto had a 600-by-800 pixel monochrome display, arranged vertically like a sheet of paper; a 3-megabit Ethernet; a removable hard disk; and up to 512K bytes of RAM. Alto was implemented in small and

medium-scale TTL logic. The CPU was microcoded, and all microinstructions were completed in a 170-nanoseconds clock cycle. That's about six megahertz. Main memory cycles were about one microsecond, and by running 32 bits wide, that gives a total memory bandwidth of 32 million bits per second.

In addition to emulating various instruction sets, microcode in the Alto was also used to implement much of each peripheral controller. The hard disk controller had, for example, only a 16-bit word buffer. Since in those days disk data rates were 1.5 million bits per second, microcode could leisurely read this buffer once every ten microseconds, and small buffers meant low cost.

Star was to be a much more powerful workstation. Star's display was to be 1024-by-800 pixels, some two pages, compared to the Alto, and consumed 51 megabits of memory bandwidth. Included too was a 10-megabit Ethernet, up from 3, and 10-mega[byte] disk drive. In addition, Star's application software with overlapping windows and multiple fonts, and the Pilot multitasking, virtual memory operating system were both written in Mesa. They were to be emulated by this same processor at the same time. Alto, in short, wasn't nearly fast enough.

I had studied our situation for my boss, Bob Metcalfe, and was talking with him one afternoon in 1978. I was lamenting the problems we faced for Star, to build such a piece of hardware that would support such a thing. Bob's mind was wandering, in the way it did when he had already gotten the point. [Laughter] But he looked at me suddenly and said, "So what are you going to do about it, just sit here and whine?" [Laughter]

SDD had no charter to do processor hardware. The last machine I had personally designed was my home-brew 8080 in 1976, and I was at the very bottom of the SDD org chart. My coworkers, Roy Ogrus, Ron Crane, and Robert Garner, were working on the new 10-megabit Ethernet. Robert had heard tell that Butler Lampson had sketched out the design of a new kind of CPU, on seven sheets of yellow paper. Butler's idea would make it possible to build a high-speed machine with relatively small I/O controllers.

Alto had been based on the [TI] 74181 accumulator chip. Butler's design, which was later called Wildflower, because they were expected to come up everywhere [laughter], used four 2901 four-bit CPU slices. He had a preliminary data sheet from National Semiconductor which convinced him that the machine could be built with 100-nanosecond cycle time and a 300-nanosecond memory cycle. That would be a ten megahertz machine. Best of all, Wildflower's design could guarantee memory latency for I/O controllers and get us the speed we wanted with Alto's simplicity and low cost.

The genius really of Butler's design in the concept was to synchronize the scheduling of microinstructions, not only with the main memory, but also with peripheral controller microcode. Microprocessor tasks were scheduled in three instruction groups called clicks, and in each click, microcode had exactly three microinstructions and one memory access. Clicks were arranged into a round, with each controller able to demand only a fixed number of clicks. In Dandelion's case, there were five clicks per round. The disk controller had one click per round assigned, so the hardware for the controller could be designed knowing that one memory access was available every 15 clocks.

Clicks not required by the controllers were given to the Mesa interpreter. This scheme gave low-cost controllers, DMA, microcode-based virtual memory, and complete flexibility for the design of Mesa's byte coded instruction set. Another way of looking at Wildflower's architecture is that the CPU acts like the system bus in modern machines.

One addition we made to Butler's design was the addition of an Intel 8085, which we kind of snuck in the side. It was to manage all the low-speed I/O, such as keyboard, mouse, serial communication, floppy disk, and real-time clock. The 8085 also took care of the job of initializing the main CPU, to put it into operation so we didn't have to have a huge amount of logic to support that function. So, as a result, we ended up with an early dual-processor system.

Both Bob Metcalfe and Dave Liddle were ready to be behind us in this adventure, but they needed a document and lots of slides to support our claim that we were doing the right thing. And to reinforce what Dave's just been talking about, this is the environment, the same period of environment he's talking about, and the kind of, as Dave was saying, no possible way of projecting, of interpreting this back from the current environment. In those days things were very simple. Microprocessor technology was not in the state that it is now, by any stretch of the imagination. And the personal computer, as I said, there was hardly any such animal.

Clearly, our only choice to achieve Star was a custom processor. Dandelion was an order of magnitude beyond anything currently available or envisioned for the near future. By the way, we picked the name Dandelion because PARC had been picking project names after very aggressive beasts. Notice the aggressive beast hidden at the end of a common wildflower. [Laughter] Dandelion's inception came at the end of the bipolar era. Motorola's 68000 was introduced the next year, in 1979, and signaled the end of bipolar machines. CMOS speeds and densities grew very quickly from then on. Dandelion shipped in 1981, the same year as the introduction of the IBM PC. The Apple Macintosh, the one that shipped in 1984, was also started that same year using the 68000 as its CPU.

The team on Dandelion, and I'm going to actually put up a slide while Robert Garner's coming up, to show the people that were involved in the very earliest part of that design. They were a very remarkable group. Our challenge was to have a picture on the screen by Christmas of 1979. This was accomplished by feeding the output of a scanner into memory. [Laughter] Between 1978 and 1981, while the workstation was moved into production, the design was enhanced to support file servers and print servers. So here's a look at the people, and we'll get Bob Garner to come up and show some of these wonderful pieces of hardware here. [Applause]

Robert Garner: Thank you. I think actually many of the people on this list are here. This feels a bit like a reunion. For me, it's been going back and reawakening parts of my brain that I thought were gone a long time ago. [Laughter] It's been quite an experience.

Thanks, Bob, for that introduction. I'm going to do a little hardware show and tell. I'm going to try to set up some context as well. You've heard from Dave Liddle how this was an era of machines that are totally different from what we know today. I found some old slides in an old box, and had them scanned in, and I'll show you up here to give you a little bit of a feeling of what the context was like.

I think it was Gordon Bell who remarked many years ago that the design of a machine is based not only on its architecture, but also on the financial and the cultural aspects of the organization that builds it. And this was no where else [more] true than here at Xerox. And at the time, well actually today, a very powerful computer fits in our pocket. You know, my ten-year-old understands that. But in the mid-1970's, a computer of comparable performance -- if you inadvertently bumped into this thing [Slide shows a room-sized DEC PDP-10] you'd smash your nose. So, what today fits in your pocket looked like this in the mid-1970s. And these were the real computers. These are the ones that we were all supposed to aspire to, and build, coming out of school. Not these little things you can put in your pocket today.

So this was a much loved machine, the PDP-10. A 36-bit machine, a little under two MIPS in performance. What I like about all this is that everyone now knows these terms [laughter]. And about a megaword of memory. Not a megawatt of power but a megaword of memory. [Laughter] A very much loved machine. The designers of the Star software would have loved to have had hardware of this power. But the technology wasn't quite there.

Now, a machine that was a little closer was the PDP-11. This is the Model 60. [Slide shows a large DEC PDP-11/60.] It was a 16-bit machine, with little more than half a MIPS of performance, and a quarter megabyte of main memory. It turns out that these features were almost precisely what the Star [Dandelion] designers were looking for in performance. Now, the only side effect is that it's not going to fit under your desk, obviously. So something had to be done.

I notice Dave referred to the two very sexy I/O devices on the side there, the character terminal and your own personal printer. So this was kind of what we wanted, but still it was a little too big.

So, what made the Star hardware unique? Bob talked a lot about this. Really it was based on the Alto, the designers here at PARC of the Alto realized that really what they were designing was in essence an appliance. I called it an office appliance, here. You know these days we hear about computers as appliances, but I think this is actually the first example, because it did a very fixed function. A very simple fixed function, not a real computer opened up to the user. So one of the first observations the researchers at PARC made, was it didn't need a general-purpose, universal I/O bus. Not necessary. And those universal I/O busses were in the PDP-10, in the PDP-11. But you didn't need that in this system. In fact, if you wanted to expand functionality, you did it via the Ethernet. So this was kind of a breakthrough idea that led us down this path.

So, again, like Bob talked about, it allowed us to have very tiny I/O hardware, in fact most of the I/O operations were done in software, in microcode. And basically it gave us state of the art performance just as good as those previous minicomputers, with a third of the power, weight and cost. So the Star was actually a tremendous achievement for the time, that all of us are very proud of.

And Bob talked about this, and I'd like to show a picture of what it was all about. Butler Lampson, basically, and some of his cohorts here at PARC, came up with this idea of reworking the memory system so that the I/O devices basically were divided into five slots, or rounds, like a round-robin. Each time the device got hold of a slot, it could move one memory word, and it could execute three microinstructions. So this shows the five slots, one for the disk, two for the Ethernet, one for the display, and one for the slow I/O. And so this guaranteed that if a device needed to access memory, it always knew it could get there within two microseconds. And once it got there, it would always get guaranteed bandwidth, one megabyte per second worth of bandwidth.

So this is actually a "correct-by-construction" hardware design, with respect to the I/O. As long as that box ran at the clock rate it had to run at, the I/O was going to work, and not have one of the terrible side effects that many of those minicomputers had at the time, which was when too much I/O happened, they tended to drop data on the floor. So, this was a very innovative design, "correct by construction", which is a phrase I dislike, but it's actually true in the case of this machine. We knew it would work as long as it met the cycle time.

I'm going to do a little show and tell. This is where we get a little physical here. I've already got my fingers poked several times in the backs of these boards. [Holding up an 11" x 15" PCB.] This was the CPU card for Star. On the top – it's 16-bit wide, so there are four four-bit bit slices, from AMD, the 2901's that Bob referred to. And on the bottom were four thousand entries of 48-bit wide microinstructions. All of this ran at 137 nanoseconds cycle time, or 7 megahertz. All that, of course, and more, fits on a single chip today, so it's kind of disheartening to see. [Laughter] But it's good stuff. Okay.

This is a quarter megabyte of dynamic memory [Holding up another 11" x 15" PCB]. A quarter megabyte! [Laughter] A quarter! Eight times as much memory fits on one of these chips [today]. [Holding up a 32MB memory SIMM], if I cover up the other ones. So eight of these [boards] fits on one of these [chips] today. These are 16-kilobit DRAMs, and today they're 16-megabit DRAMS, so there's a factor of 1,000 in the number of bits per chip stored there, per chip stored here. So, that's the industry for you. Again, it's very humbling.

What I almost forgot about the memory is that Dave Liddle mentioned it supported virtual memory, so even though the memory itself was only a quarter megabyte -- by the way, it's nice to hear, Dave, that you finally left the gate open so that the poor software people could run with more memory, because I remember it swapping too much. Later, with the 64-kilobit DRAMs, it could get up to a megabyte and a half of memory. But it did support 8 megabytes of virtual memory, which seemed like a lot at the time. And one of the amazing things about this board is that it supported error correction codes, so if there was

a single-bit failure it would be corrected, and that actually was a feature that didn't show up on the [IBM] PC you bought later that year.

Now, the Ethernet. Of course the first Ethernet was done here at PARC, the 3-megabit Ethernet. We worked on the follow on version, the 10-megabit. This board [holding up another 11" x 15" PCB], even though it looks big, is amazing. The top half of the board is the Ethernet controller itself, and demonstrates how small the hardware was, in the case of the Dandelion. You have to realize each of these little black chips maybe has eight flip-flops in it, and four NAND gates, so really, small-scale integration. So the Ethernet controller was only 88 chips, whereas in the minicomputers that I had showed you, there's more like hundreds of chips. In fact, I'll never forget the VAX-11/780 Ethernet board, it was as big as your washing machine. Just for the Ethernet. When DEC and Intel came and looked at and saw our Ethernet implementation, they just about collapsed, because -- maybe they thought we had an advantage. [Laughter]

So, anyways, one of the stories here on the Ethernet was, I was designing the -- it was called the Xerox Wire at the time. The target was 20 megabits a second. And I designed a board for a predecessor machine called the D0 or the Dolphin. And I laid out the board, and it didn't fit. So I went to Bob Metcalfe and I said, "You know, Bob, I can't get all the chips to fit at 20 megahertz." And I opened up a -- remember those picture catalogs of parts? -- opened up a parts catalog and there was a Fairchild 10-megahertz CRC [Cyclic Redundancy Check] chip. So I went, tail between my legs, to Bob. I said, "Bob, if we run at 10 megahertz, this all fits." So that's why Ethernet runs at 10 megabits a second. [Laughter]

Now, we get into some fun stuff. Disk: this was the Shugart SA-1000 -- uhh! [Laughter at straining to lift the 25-lb, 5" x 9" x 14" drive.] Which was the industry's first low-cost 8-inch disk. I'm not sure how much it weighs, but it's low cost, a thousand dollars. They're actually very reliable. The designer of this disk might be in the audience -- I don't know if Joel Harrison is here. I met him recently at Quantum, and he told me lots of secrets about the disk. He said, "They still work!?" [Laughter] Apparently they thought the heads would have welded to the surfaces by now. Some customer he said they had, said they'd never buy them because they wouldn't last 20 years. Well, he's going to tell that customer they're still working. [Laughter]

We had one fun experience with it, which was, the disk ran synchronously with the CPU. Totally synchronously. And you had to supply the clock data to the disk, and so that was tied to the Dandelion's clock. So when this system was sent to Japan, where they have 50-Hertz AC motors [laughter], it didn't work. And they learned they had to replace the pulley and the belt in Japan, so there were two stock versions of those, for 50 Hertz and 60 Hertz.

Now of course today -- this is 8 *mega*bytes [lifting the SA-1000] -- this is 8 *giga*bytes [showing a new 4" x 6" x 1" disk]. [Laughter] So, yeah, a thousand X. Software better be a thousand times better. [Laughter] To be honest with ourselves, this obviously wasn't enough for a network system to store all those images that Dave was saying needed to be printed, so the Dandelion could also support bigger disks, Trident and larger Quantum disks, which had to be in a housing all by themselves.

All right. The display was 808 [rows] by 1024 columns, 38 frames a second, and you can't change that, so actually when Dave gives the demo, there's going to be a little bit of flickering you're going to see, because we can't adjust the frame rate on these guys. Remember they're tied to the CPU clock; everything's running synchronously. The innovative thing about the display is, there are 32 extra lines at the top and bottom and sides, which are called the border pattern, and we could make that equal to the background pattern of the rest of the display. I had an interesting explanation of this. Maybe Dave can [corroborate] this. Apparently, in some cultures, if you have a picture of someone with a black border, it implies they're dead. So they didn't want to have a display with black borders, so one of the nice features about the display is that you can make the border be any color you'd like. No dead pictures.

And again, I've said the Dandelion is synchronous with the display. The CPU cycle equals seven pixel times on the display, the memory equaled 21 pixel times, and the disk equaled six pixel times. So it was a pretty unique machine. (Have to do something about the scan rate.)

Finally, the IOP that Bob told you about. So this is where we use this Intel chip. Actually, [holding up another 11" x 15" PCB] this is an AMD chip. [Laughter] Kind of like an uninvited guest, in some sense. It was definitely okay for running low-speed I/O, but over time I guess it became something of a Trojan horse. It could handle this [floppy] disk. So as part of this board, the 8085 ran the 8-inch floppy drive, which we all recognize. Which [holding up the 25-lb, 5" x 9" x 15" floppy disk drive] has also gotten dramatically smaller, these days. The 8-inch [floppy]-- I guess all computers had to have a floppy, even though there were networks.

The other things we had on here were the keyboard controller and the LED maintenance panel, which is underneath there, [pointing to front of desk side pedestal] which tells you what state your machine is in, and connections to the laser printer, comm[unications] ports, and TTY ports. So actually it was a pretty capable machine, we just didn't tell all the customers they could take their Star hardware and hook it up to a laser printer, and all that kind of stuff, that the same hardware could do that.

Question: How much data on the floppy?

Answer: Oh, 800 kilobytes, as I would guess. I think the double density were 800 kilobytes. [Inaudible comment from the audience.]

That's it for my hardware show and tell; my arms can't take any more. Although I guess I could show -- this was the original Ethernet transceiver [lifting a 2" x 3" x 4" metal box]. This was an evolution of the 3-megabit design. There are probably some up in the ceiling, I'm sure, still here. [Laughter] In fact, I'm sure there are some Dandelions probably running here at Xerox, and these machines are probably homing in on them right now. [Laughter]

This machine wouldn't have been possible if it weren't for the revolutionary network design environment that had been created, and I stepped out of school and walked into this environment. I thought I'd walked into a science fiction novel. At Stanford I was using punched cards, and all of a sudden here are the Altos with their bit-mapped screens and networking, and I thought I was on drugs. [Laughter] But I wasn't, and what this environment allowed us to do was to take advantage of these tools, and we actually ran like a small startup. One thing I don't remember are the stock options. [Laughter] But it allowed us to work around the clock. With e-mail, you could say what you had done, and when a person came in and you weren't there, could read it and take over where you left off. And we actually did. We worked around the clock, and so we actually ran this project -- my father asked, "How can you work around the clock, when you can't communicate with people?" "I just send them an e-mail message." "What? What's that?" So, it allowed this kind of productivity -- and so actually we, from the start of the design to having hardware running just took us a year, which is kind of standard for a complete system, and then another year to announce and ship. And the hardware, like Dave mentioned, was actually announced first as a file/print/communications server, in November of 1980. And then April 27, '81 as the Xerox Star 8010 Professional Workstation.

I found an old slide, and took it down and got it scanned. This was our debug environment. It shows -- this is our favorite windmill situation, where we had the six boards, the single station, where we could access all the boards simultaneously. This was stich weld wiring technology, so we could turn a board around really quickly, like in a couple of hours. Remember, no software simulation environment. The Alto ran this great user debug program called Burdock, which allowed us to single-step the machine and set breakpoints and whatnot. I think that person is Dan Davies [pointing to slide]; he couldn't make it here tonight. But we were really proud of that debug environment. There's actually a person, I've learned, at Apple who's trying to recreate this environment. So I'll let you know when he gets it running.

Moore's Law has been operating since 1959. The Star was -- I know they sold it for much less than the list price, I just wrote the list price up there [$16K], I'm sure Dave could sell you some at a great deal. [Slide: List: $16K Weight: 70 lb. Power: 1.4 KW]. They were a little expensive, they weighed a little bit, and they took all the power out of your one outlet in your office, but again, they were a third the cost, a third the weight, and a third the power of a machine which really had the comparable performance at the time. And so they were actually revolutionary for their time.

But you know, now, memory chips are a thousand times denser, the CPUs are a thousand times faster, and so I have wondered whether software's a thousand times better, but one thing I've noticed, looking at these statistics over the years, having worked on RISC machines since here, was that the I/O has only gone up 10X. So even though the processor and memory's gone up 1000X, the I/O, the disk drives, and the Ethernet, are lagging. So, if you keep making the software which page faults and uses the networks, it's not going to run that much faster really, and so actually when you see the demo that Dave will be giving next, you'll see it's actually pretty quick. Pretty capable.

So finally I wanted to say, if a picture's [worth] a thousand words, I wanted to say a thousand thanks to Xerox. Xerox put the investment, they put the time in, they allowed the team to be creative -- you know, walking into this kind of environment was a magical experience for everyone involved. Xerox really deserves a lot of recognition for doing that, for allowing us to put into the marketplace a pioneer product which was totally new, a new concept for a PC, a new concept for a workstation, basically the world's first bit-mapped, networked PC. A thousand thanks go to Xerox for allowing us to do that. So, that's my presentation. [Applause]

Well, next Dave Smith's going to give us a demo, and Dave was one of the human factors designers on the Star.

Dave Smith: Thanks, Bob. I'm going to give about a 30 minute demo of Star from a user perspective. Here we have a couple of the first version Stars, the 8010. There are two, because there's a definite, non-zero probability one of them will break in the next 30 minutes, and we'll switch over to the other.

Down here is the version 2 of the hardware, the so-called 6085. It's about half the volume, or even less, of the first version. It came out about five years later. And down here we have the version 17 of the Star [laughter] system, which is a little smaller even. Here we have a Star with the skins off, so after Bob's tour of it you can probably identify a lot of the pieces inside there.

So I'm going to give you a demo. I'm going to pretend I'm a new user, and I'm going to set up a working environment, and I'm going to do a multimedia document. And I've done this a lot of times, and like the talk says, this is my last time [laughter]. I've got to get out of this business. And in fact it's hard to give a talk on Star, because more and more, when I do, people say, "What's the big deal? Everybody does that."

So one thing I want to reiterate, from a user's perspective, is thinking back to those days of the mid-70. All the screens, all the terminals that people used in those days were text-oriented and they all used command line interfaces. So our challenge, as a designer, was to try to bring the computer into the office professional world, instead of trying to get the professional to become a computer user. These guys were all pretty computer phobic, and most had never used a computer before.

So I'm going to talk about, I hope pretty forthrightly, Star's strengths and weaknesses. Some of its strengths still aren't in products 20 years later, and some of its weaknesses are pretty legendary. So our strength was, this was a breakthrough product, which made a huge improvement in computer usability by non-computer professionals. Today the basic interface ideas are in use by over a hundred million people worldwide. It had the first visual point-and-click interface. It was the first object -oriented interface. It was the first use of icons. It was the first interface to identify and follow what have since become standard human-computer interface principles. It used the first commercial Ethernet. It used the first commercial laser print server.

But, another way to put a spin on that is, a weakness was, it tried to do too much all by itself. It tried to do too many new things. It did some well, and some poorly, and really all of them had to be done well for the thing to be a success. From my point of view, I'd like to point out another weakness that I think was the biggest mistake we made, and that was to make it a closed system. As Dave Liddle pointed out, there was no independent software industry at the time, so it made more sense, in context, but in retrospect that was the biggest mistake we made. For example, the Macintosh toolbox was a huge improvement over Star.

Another weakness of it was even with all the impressive performance that was built into this little machine, it was still drastically under powered, and Star would have done quite well on today's machines, thank you very much. But the software was about 10 years ahead of its time for this hardware. Basically the machine was about the power of the Mac Plus, I believe, is that ... [about 1/2 MIPS]?

A couple of things I want to point out: it's a two-button mouse here. Why two buttons? Because user testing showed that one was too few, and three were unnecessary. [Laughter] The Mac has two buttons, too, for that matter; I always said the second one was on the keyboard. And the PC has three or four or five, or as many as you want, and I won't say anything about that. [Laughter]

Now, the keyboard was pretty innovative, and if I was to point out the biggest reason for Star's user interface success, it's this keyboard. In particular, in addition to the standard typing array we had three banks of function keys. They were hard-labeled with their meanings. The most important were these on the left side; in fact the bigger ones were the most important of those, Delete, Copy, Move, and Show Properties. Those four generic functions were based on observations of what fundamental principles were in computers. In a programming language those were the operations you do: you copy data structures, you delete data structures, you move data structures. So we elevated those to the level of the user interface.

Across the top we had some that dealt with text, like Bold, Italics, Underline. And down here we had some others. So, the effect was, you could actually operate Star for quite a while by putting your left hand on these keys and your right hand on the mouse, and if you were left handed we apologize. [Laughter] And operate Star that way. It was two-handed operation. It was sort of modeled after what Engelbart was doing at SRI at the time. Only he was using this strange little keypad that user testing showed that mere mortals couldn't learn. [Laughter]

But you could do this. So now, I'm going to -- one of the points I'd like to make here, from the user interface perspective, is that this interface has still not been surpassed in many ways today, and one of the ways it has not been surpassed is in the very few commands that you need to do complicated things in Star. I'm going to create a multimedia document using fewer than a dozen commands. In fact, my entire demo is going to use fewer than a dozen commands.

All right, what else? Oh, there's another principle: we really liked this idea of Seymour Papert's about powerful ideas. [A] powerful idea is something of great generality that if you apply pervasively, can make a system both more powerful and simpler at the same time. And as a human interface developer, that's a very important pair of criteria. So one of the powerful ideas we adopted was the notion of copying. Copying is a paradigm for operating the system, and you'll see that in the demo. We made copying pervasive, in Star. Copying is easier for people to do than creating by scratch. That's why people like my son here, who could take tests, loved multiple choice tests; all he has to do is figure out the most likely answer, rather than the fill-in-the-blank tests, which require you to generate something from scratch. By the way, that's the reason people don't like command line interfaces, is because you're faced with a blank screen and you have to generate a command line with all the switches and everything from your own memory.

Okay. So, let's do a demo. Okay, the first thing we're going to do is login to the system. Star was networked from day one, [comment from audience about projected screen "wiggle"] -- yes, we apologize

for all the shaking. If you were to come look at the Star screen afterwards, and you're all invited to, it's really rock solid. It has to do with the mismatch with the scan rates. This is an interlaced screen, and the camera just can't get a sync rate that will stabilize it. So, the screen I'm looking at is not wiggling like the one you are, and we apologize for that. So come up afterwards.

Okay, so I have to log in to even do anything with Star. But, I only have to login once. I've now identified myself to the system and every server I go to or any other host anywhere on the network -- I no longer have to login, it will negotiate for me. Now look at the top of the screen here. [Laughter] See that white thing up there? That is not a menu bar. Let me repeat that: that is not a menu bar. This is an area for messages and prompts and things. Way over here on the edge is a pull-down menu, sure enough, but those are infrequently used commands that we didn't have time to get rid of. [Laughter, applause]

Down here in the lower right corner [interruption] -- good man. I have a monitor here too. This is not part of the Star hardware, by the way, this little Sony monitor here. -- this is our handle onto the whole world of the network. It's what we call a directory icon, and I'll open it. And it opens up into a window, and here are all the categories of things that are available to me as a new user. Remember, this is my first use of Star. So I'm going to say, let's see now, I want to get a working environment together that will let me do some real work.

Okay, basic documents. That sounds promising, so I'll open that up and see what's in there. I'm doing that by double clicking on it, or there is an Open key on the keyboard that will let me single click it and then push Open. Okay, now here's where I'm actually going to use, these generic commands. I'm going to select this thing and say Copy, and then I'm going to move my cursor over here to the right and deposit it. So, that was a use of the Copy key. That got me one of these things out onto my desktop.

Now we're going to go back over here and copy some others in. [Q: You didn't call it the desktop back then, did you?] Yes, we did call it the desktop, and in fact it was very deliberately called a desktop, and the icons chosen were very deliberately chosen to be familiar to the office professionals. So the way we chose them was, we looked around the office: well, let's see, there's some folders, and some documents, and file cabinets, and, well maybe not printers, but there's a bookshelf, and a wastebasket, and so forth and so on. And so those were the icons that we built into Star. Trying to get the computer into the user's world.

Okay, so I'll copy out a folder. What else am I going to copy out? Here's something, a records file, this was our equivalent of a database. And so I might want to use that. Here's a spreadsheet, so I'll copy that out, what the heck. All right, that's all these things, let's go back and look at our categories again. Hmm. Okay, Filing. Well, let's see what's in there. A bunch of file servers. And you can see we've named them, like universities. I don't want to do that, I'm just going to go on. Okay, Mailing, that sounds promising, I'm going to want to do some mail. All right, let's see what we've got here. Okay, an Out basket, that sounds promising. So I'll copy the Out basket and In basket -- I'm selecting these things in every case [by] just saying Copy. [Q: You're not dragging, it's not a click and drag?] No, it's a noun/verb/destination. [Comment: You have mail!] [Laughter] Gee, from 1980. [Laughter] I hope the issue hasn't expired here, I'd better read that, I'll read it a little bit later here.

Okay, let's see, we're going to want to do some printing, so I'll get a couple more things out. I think by now you're getting the idea. This is a browsing interface, to the network. Basically, this use of the directory icon allowed us to bring Star's interface ideas to the world of the network. It wasn't until Andreessen did, what was it, Mosaic? What was the predecessor of Netscape? Mosaic, yeah. Until he did that, we had a similar interface to the network.

This is probably enough of that stuff for now. I'm just going to close my little directory. So now let's take a look at this whole screen here. So this is what I've set up. I set up some documents, some folders, a database, spreadsheet, mail, printing. Ready to do some work. So let's go to the blank document and -- now here's where copying becomes pervasive. The way you work is, you make a copy of an existing

document, and then you edit the copy. So therefore every document becomes like a form pad, a source for new documents. There's no special concepts needed.

So I'll open this -- let me see, before I do that, I want to show you another key I haven't used yet, the Show Properties key. I said that Star was an object-oriented interface. These little pictures are not just pictures, they're objects that have semantics. They have both state and behavior, like any object, in object-oriented programming. And when you say Show Properties, it will show you the state. So, this one has things like a name, and a couple of information-only properties, so I'll call this the "History Demo."

All right, so now my little blank document becomes the History Demo. Now I'll double click on it and open it up. And sure enough, it's blank, just like the source that you -- now let's go up and look at the top of the window again. Okay, there's three commands here: this question mark is for context-sensitive Help, this closes the window, and this causes a paginate, or page layout, to happen. Way over here on the right is a pull-down menu of commands to deal with tables. And even further, there's another little pull-down menu of infrequently used commands, that we didn't really want you to pay any attention to.

Let's go up and look at this top thing. This is still not a menu bar. [Laughter] This is it! These are the commands necessary to construct a Star document. Okay, so I'll just start doing some stuff here. Okay, this is very tiny, so let's -- the first thing we should do is make it larger. I'm going to select it and hit the Show Properties key, because like anything, text characters are objects. So they have attributes. They have both paragraph and font attributes, so, these are the paragraph ones. I can make it centered, double width, double height. Let's go look at the character properties. Okay, let's make it the Classic font with serif, 24 point size, and bold. Okay. So when I say Done now, that thing I selected turns into the properties. [Applause] Well, a very nice audience here -- you're all invited to the next "last talk." [Laughter] Actually there's the penultimate last talk, and in April we gave the ultimate last talk, so this must be the post-ultimate last talk.

Okay, so now I'll type some more stuff.

Let me see, this is too big, so I'm not going to select this whole paragraph. Now, I could bring up the property sheet, that's what the top row of function keys let me do, as they're accelerators for having to go to the property sheet and click those properties. Okay, here's the top row of function keys. So I can do things like I can make it centered or not centered, I can make it larger or smaller, just by pushing some of those keys. And so common operations could be done using this top row of function keys. And I'll show you, in fact, that those are virtual keys that are remapped as the context changes.

Okay, so now I'm going to show you a new thing here, a new function called the Keyboard, the virtual keyboard, when you push the Keyboard key, and when I do, you'll see that the top of the window here, that top row of function keys, of which there are eight, get re-mapped to these meanings. So I'm going to show you the key by pushing the Show key, and down at the bottom of the screen, the virtual keyboard shows up. And, I can say, let's see some office symbols. For math, for logic symbols, Greek letters, what else? Italian, Spanish, Russian. Or, it may be the Dvorak keyboard layout, if you'd like to set it to that, and type with that one.

But what I want to do now is, I want to go to the special keyboard set, and these are smart characters, or smart objects. And one of the ones I'd like to do now is insert a graphics frame, so I'm going to hit the A key, which does a graphics frame, and back up here in our document a little frame will appear. [Inaudible question from audience] The question is 'could you click on those virtual keys with the mouse?' and the answer is yes. But before I do this, I'm going to move this keyboard window off to the right side of the screen. Okay. Now, when I select the frame, this graphics frame, you'll see that the top row of keys automatically got re-mapped to graphics type things, like stretch and magnify. So I'm going to push the Stretch key, and now I can adjust the size of this thing, okay, great.

What I want here is a bar chart. So I'm going to go to this second document that I copied out earlier, this basic graphics transfer sheet. This is modeled after the rub-off sheets [Letterset, for example] that existed in the day. These were little sheets, transparent sheets, with graphical symbols on them, and you would put them on top of a typewritten page and scrub on the back of it with an eraser, and it would transfer the symbol onto the page. Well, we did the same thing, only electronically. We have little electronic transfers here. So, we have some simple ones like lines and rectangles and ovals. So I'm going to select this one and say Copy. And, using that same command that I've been using all along, the Copy command, I can get a copy of it, and I can stretch it since it's a graphical thing, and I'll stretch it out a little bit here. All right. Let's see, let me move this thing so that I get it where I want it. I've got a kind of sticky mouse, so it's hard for me to do this. Ah, that's basically good enough.

So that's a very simple little graphic transfer. Here's one that's maybe more interesting. Down here is a bar chart. This is a complex object but I deal with it in exactly the same way: I select it, I say Copy, and I click where I want the copy to go. And I can stretch it, using the Stretch command. So even though it's got a whole lot of semantics to it, the interface is the same.

Okay, so I'm going to stretch it out so it's oval. Now, it's an object like any other object, so it has attributes, and so I can use the Show Properties key to see what they are. And here you'll see how some pretty fancy stuff -- it's got some spatial properties, and it even has a table of data. So what I'm going to do is, I'm going to fill in the data table. I'll type some demographic data for Curbow and me. Let's see, what I want to do is column by column, so I'll hit the Next .... Okay, so the first thing maybe we'll enter for you guys is our heights. Dave is about, I don't know, 75 inches tall, and I'm about, oh, shall we say, 50 inches tall. [Laughter] And we'll get even more fanciful here because we'll do weight. And Dave is about 215? Going for the gold, huh, Dave? And I'm about 150. [Laughter] There's my little data table, and I'd like to plot this data into the bar chart, that's what bar charts do, right? So let's go back to the spatial properties here, and get this chart looking the way we want.

So when I switch these little tabs, or pages, it probably wouldn't move so far, there it is. I don't like that. I want just this kind of thing -- I want the bars to be side-by-side, and a little space in here. And, oh, by the way, that one color's kind of beating on the screen, so I'd like to change the weight color say to be black. And now when I say Done, it will apply these changes to the chart, and there you are. [Applause]

Let me just center this guy again. I wish my mouse wasn't quite so sticky. Okay, well, we're done with this little transfer sheet, and that was pretty painless, you know, the first time I had to create a chart, maybe against a deadline or something. I'm going to just insert one other kind of thing here. Using our special keyboard, we're going to insert a[n] equation frame, which we do with the C key. Now when we insert this equation frame, it automatically positions the caret, the typing point, inside the frame, and I can type some standard ASCII, like F of X equals -- I apologize for not being able to make this larger, this is the largest font I can do with this version of Star.

And now I'd like to, say, maybe do a quotient here, so if I go to the equation frame now, and if I go to the special keyboard now, the fact that the context is an equation frame gives me a different set of special characters, equation-type characters. So, it's context sensitive, so I can say I want a fraction, and in the numerator I want a summation, and you'll see it positions the caret automatically in the lower limit of the summation. And I can type I = 1 and then hit the Next key. Now the Next key -- actually I was using the Next key, I forgot to tell you about it. When I was stepping through the rows and columns of the table, I was doing that with the Next key. Now I'm going to use the Next key to step through the parts of an equation. So I hit the Next key, it jumps to the upper limit, and I can type 100, and hit Next, and I drop to the right side. I don't know if you can see it or not, maybe you can see it in just a second ... but, I'm going to say X sub-I squared. There's a couple of things I'd like to point out. First of all, can you see that the letters are in italic and the numbers are in Roman font? Which is proper equation formatting, and I did not have to switch the font to get that. Also, the superscript is directly above the subscript, not offset to the right as in word processors, which again is correct equation formatting. And it does that for you automatically. So I'll just type some more stuff, some random - see, a product, say over J, and we'll say it's a Y sub J. And now the Next key I'll eventually get to where I want to be, which is in the lower limit. For

this I maybe I'll insert a[n] integral, from A to B, and let's use some Greek stuff here, so I'll choose the Greek special characters, say, rho of theta and then the rho, I guess, the theta.

So, I just created a mathematically typeset correct equation. [Applause] This is very rewarding, Dave. [Laughter] The last clause of that statement was going to be, only using the special keyboard -- I didn't even use the Copy or those commands this time, all I used was the special keyboard feature. And the smart equation objects.

All right. In the interest of time -- I was going to insert some things like fields, and some other objects, which I could do, into this multimedia document. One thing I do want to do is get rid of the frame border around this equation, this graphics frame. So the graphics frame itself has attributes, just like the objects inside it. Its attributes have to do with its border and whether I want to center it and so forth on the page.

And now I'm going to -- now here is where a weakness comes in, in Star -- I actually, to get this to look the way it will look when it's printed, I actually have to expressly paginate it. The hardware simply wasn't able to keep up with interactive pagination. Sorry. But, let's look back here at what I've said.

Okay. So, I created a multimedia document, properly typeset and laid out in a WYSIWYG fashion. If I were to print that, this is exactly the way it will show up. And I did it with fewer than a dozen commands, as I claimed I would do. If I wanted to print this thing, I mean to, say, well, print it, or to mail it, what I would do is drag it to one of these icons. So, in fact, here is where -- I guess I forgot to point out, when I was talking about icons, we divided the icons into two classes: data icons and function icons. The function icons operated on the data icons. The data icons were documents, folders, record files, spreadsheets. The function icons were mailboxes, printers, file cabinets, and so forth. The way you got a function to operate was by using the Move key. You selected a data icon, said Move, and then clicked on a function icon. So I've just invoked the Mail command by using the more generic and general Move command. So all I have to do here is type an address, say Done, and it will go off and mail this thing for me.

So, I want to sum up here. There's a couple of lessons that I think bear thinking about, 17 years after Star came out. 1,000 times later in machine speed, memory density and disk density, Star's interface still is simpler and more consistent and more useable than the systems we see today. [Loud applause] In fact, the most pleasant experience for me in this whole thing -- actually Dave Curbow has had to do all the hard work of making this hardware run, all I've had to do is get familiar with Star again. And the most pleasant part of all that is finding out, all over again, how well Star has survived, how well the ideas have held up in the intervening 17 years. It doesn't seem like an antiquated system. In fact, if there were any way I could use this thing [laughter], I would. But there's no where I could print it -- nothing to print it on, nobody who can read this e-mail. This e-mail, by the way, that I just sent, I have equations in it, I have all sorts of non-ASCII stuff in it. It doesn't matter. It just all went. [Comment: Maybe somebody will write an emulator.] [Laughter]

So, Star had many fewer commands than today's systems, and it didn't do it by having fewer functions. It just had fewer commands. And the way we accomplished that, I hope I illustrated, was through the use of generic function keys, through objects that had property sheets, through virtual keyboards, and through smart objects that would do a lot of the hard work for you. We also had taxonomy of icons, data and function icons that led to simple ways of doing things, namely, you move to a function icon and that handled a whole bunch of commands there.

And lastly, this notion of a Copy paradigm meant that I could do quite fancy graphics if somebody had created a little graphics transfer sheet for me to use. All I had to do is then copy those things into my document.

Okay. That was pretty much the demo. Now Dave Curbow is going to talk about some of the technologies other than this interface. Obviously the interface has been picked up by all personal computer

manufacturers and most workstation manufacturers. But there are a lot of other technologies in Star that have also been picked up by other companies, and Dave is going to cover some of those. Thank you. [Applause]

Dave Curbow: So, the version of Star you saw is a version that they were working on when I arrived in '83. And if Robert [Garner] was blown away when he came here at the hardware design, I was really blown away because I came from a mainframe operating system background just before this, and I'm very proud to have been able to work on this project for several years. My job in this whole thing has kind of been, I had some hardware in my garage that I got from Allen Freier and some more people, and in the course of my work I managed to figure out how to make it work again, so we were able to do a demo for the Chi Conference, human interface [Human-Computer Interaction]. In fact, I became a human interface designer at the end of my Xerox stay because I was so impressed with the user interface that I wanted to do a better job for everybody else. So I went to Apple, and worked there as a human interface designer. Well, long story.

So, we went to the Chi Conference and everybody was really impressed and John Shoch suggested that we do this again, and so since everybody has covered all the interesting things I thought I'd tell you about the -- well, not the interesting, but some of the things that people don't really notice. Can I have the slides, please? Thank you.

All of those wonderful things Dave showed, with the equations and all that, they work because we had not just ASCII, and also we supported -- oh, gee, let's see, in the very first release we had Russian, French, Swedish, lots of other things, and then like six months later we supported Japanese. So at the end of '82, I believe, is when J-Star came out. It may be tough to see, but this is actually Japanese typed on a screen and printed. And in the course of the next few years the Xerox character code standard, which became Unicode really, Unicode came from this, supported every major language.

Some little known facts, some of the other innovations you may not be aware of: directory services. We called them Clearinghouses. They derived originally from Grapevine here at PARC on the Altos. Now, 18 years later, they're all the rage, with LDAP and Microsoft Active Directory. Of course Novell did support directories for a number of years, and they were, I'm told, extremely similar to Clearinghouse.

Courier, Remote Procedure Calls, you know, originated here. Of course it originated with the Altos first, and we productized and shipped them. Printer protocols, Internet printing protocols, that you guys, if you were really into printing, you may care about, derived from here. CUSP. CUSP was Customer Programming. Basically it was end-user programming. It was intended to allow users to program. Well, it actually turned out that, more like, consultants programmed, but anyway. It is an English language programming tool that was extremely popular with some of our major customers. I'm told that Lufthansa Airlines did everything from menu planning and everything else using CUSP. That's kind of metamorphosed into metaphor capsules, slightly different, and Apple Script, a project I worked on at Apple for a while. The intent was, again, to allow users -- well, really, consultants -- to be able to automate stuff on the system.

Boot servers. Well, boot servers, if you used an Alto you downloaded things like Maze War, and other games, and applications. In our product days we shipped something called Boot Server so you could boot Installer and utilities and things like that from the network. That's metamorphosed into application servers that now download things like Java applets. WebLogic and Kiva/Netscape do this now, very popular. Only, so many years later.

Remote access: the ability to, from some place remote like your home or some place out in the field, dial in through a modem and be connected transparently to the network at the other end. Caused Allen Freier no end of grief, trying to make it all work, but it really looked like you were on a very slow Ethernet connection. And of course that's extremely popular now in the nomadic world that we live [in], where

everybody carries a notebook computer and they want to plug in, but they have to dial in to work to get stuff.

That's just a few of the things that we did. I want to say thank you for all here who helped make this happen, like Cathy Ching, and Stella Timmons, of Xsoft, who provided me with some floppies that they managed to find, as well as some hardware. Allen Freier, who helped make all this hardware work in the first place, and of course Tony Stafford, who has a garage, I'm told, full of Metaphor machines. These in fact are 10 megabyte, very first 8010s. So, they're very old. We're very happy that they're still working. He also managed to find, in his garage, a box of software that when he left Metaphor -- Metaphor got rid of the Stars, whatever it was -- the stuff went to his house, for some reason [laughter], and it has -- he had a box of software. In this box we found software from 1983, 1984, which we actually last night loaded floppies, from 1984, and it still loaded, and still worked. We did have a bit of a problem with product factoring, trying to find keys to unlock it, but a scrap of paper was found with a password that worked, and so we were able to last night product factor things and make it all work. So we're very thankful to Tony for stashing the stuff.

And of course to the hundreds of people here who worked on Star, of which I was just one of those people. Thank you for coming, and now we'll do -- oh, there's two more things I should tell you. If you didn't sign the poster outside, the people who worked on Star, please do so, there is a poster somewhere. And the Xerox Star Retrospective, which Jeff Johnson and people worked on, Jeff has kindly provided a few copies if you're interested in that. And now we'll do Q & A. [Applause]

Nurkse: Let me say something about the Q&A session. I've heard some of our history programs, that questions and answers go on too long, and I'd like to point out first of all it's valuable information for the historical record; we tape these programs for that, so I'm not inclined to stop the questions and answers at any point. And anyone who wants to leave at any time is perfectly free to. You don't have to -- now, or during the questions and answers.

Question: Those who do not remember history are compelled to repeat it. And I'd like to suggest that even though many of the good ideas used in Star have been published and published well, that there are also important aspects of it that don't really translate their value to subsequent generations except in the form of code. So I'd like to ask the following two questions. Number one: Can anyone make a substantial case why Xerox should not put the entire source code of Star in the public domain? And number two: were Xerox to do such a thing, would it be possible to find it? [Laughter, applause]

Liddle: Peter's question is, is there any substantial argument as to why the Star source code ought not now to be put in the public domain, and secondly, if there were an agreement that that were an acceptable thing to do, could anyone lay hands on the original Star source code or a facsimile thereof in order to do that. I can't possibly answer either of those questions, except to say I cannot myself automatically see a reason why it couldn't be, as some other classical source code is, in the public domain. But that would be an issue for Xerox to sort out. The other question, though, about the availability of it, who knows? I imagine it must be ... [interruption: I think he has an answer to that.]

Curbow: Well, in fact, up until about two or three months ago Xerox was still supporting the products. There'd been a new virtual machine implementation built that ran under Windows. It runs very fast, actually. And they tell me that they have gigabytes of stuff, but they're not quite sure what versions, and stuff, because it was just a small crew that was finishing the maintenance. They stopped supporting it just recently. I suggest you talk to somebody at Xerox. [Q: Who?] Mark Weiser is nearby. [Laughter] [Inaudible comment and reply.]

Question: You said it was a virtual machine?

Curbow: There is a virtual machine implementation. It is a Xerox product; it was sold until about two months ago.

Question: It runs on?

Curbow: It runs on PCs, Windows.

Question: Is the Star program year 2000 compliant? [Laughter]

Liddle: There's a wonderful answer to this, actually. Although it's not year 2000 compliant, it does have some fairly nice features about dealing with dates and other things like that.

When is it -- it hasn't blown up yet, though. Is it year 2000 when it hits the wall, or is there something earlier?

Curbow: I think the time stamp actually blows up around 2020, or something like that.

Garner: I think there is a problem with the configuration software, which, if Allen can rewrite it -- it doesn't let you configure a new Star. I tried entering 00 and it said, "No way." [Inaudible comment] That's your code? Okay. [Laughter] That's just the code to set the time on the machine. It had a few idiosyncrasies where it really couldn't keep time forever. But we tried.

Smith: So it is year 2000 compliant, it's not year 2020 compliant. [Laughter] Ron Crane?

Ron Crane: The [one word inaudible] has time protocols, a 32-bit number, which is the number of seconds since January 1, 1901, and that 32-bit counter rolls over around 2020.

Question: I have a question regarding Chip Witch, who's the person who designed it? Xerox is still using it. It's a wonderful program that I see the designer was [inaudible]. Do you know anything about it?

Liddle: What was the name of the program?

Reply: Chip Witch, we used it in El Segundo for IC design.

Liddle: Sorry, we just used [name, sounds like "Seal".] [Laughter] And we weren't allowed to design ICs.

Question: I'm Charles Irby. [Applause] One of the things I thought it would be useful for Dave to comment on is the design methodology that we used, and the fact that the design is based on abstract notions of [inaudible] functionality rather than the specific benefits [inaudible].

Liddle: That's an interesting point. [Portion inaudible] Charles asked about some comments about the work that was done on the design methodology. Dave Smith referred earlier to the fact that it was done in a principled way. In fact, I commissioned a, you might say, a task force which Charles Irby chaired, as it were, to develop a methodology first, so before we did the design we decided on a methodology for user interface design, so that we'd be able properly to manage both the process and the abstractions that we were going to be working on. And then, as the user interface design was done, in stages, and to attack different tasks, there was a very sort of consistent approach. The information display was designed independently from the function invocation, for example, which turned out to matter a great deal. And both of those were designed independently from the user's conceptual model. And this was the first time that anything like this had been done, either the development of a methodology per se, or then a principled user interface design. That's one of the reasons why Star, from a usability point of view, is such a great improvement over most of its successors, is because we actually did it by designing it using a set of principles rather than just sort of randomly copying what someone else had done, or kind of debugging it into existence. So that turned out to be very important. Charles himself and Dave Smith, Eric Harslem and Ralph Kimball, along with Linda Bergsteinsson and a number of people from PARC, who were still in PARC proper but who happily helped us by joining our task force, and I include Larry Tesler, Charles

Simonyi, Bill Verplank, a number of other people as well, so it was a big and very focused effort. Some people thought, "Gee, it's a long period of time just to decide what you're going to decide, or to design how you're going to do the design." But I feel that it really served the process and the subsequent industry very well.

Question: It seems pretty clear that one of the major shortcomings of Star was the thing that you cited about a turnkey closed model, as opposed to somehow seeing in your crystal ball that the independent software market was going to come into existence. There was some speculation at PARC at the time that such a software market would come into existence, but of course it was all just speculation. I'd be curious about what you think, and what Dave Smith thinks also, about whether that was something that at the time was just impossible to foresee, that the Star programming would have been overreaching itself, or whether looking back on it, you feel like, gee, I wish we'd gone the other way and tried to make an open platform.

Liddle: The question was, 'in retrospect, doesn't it seem as if you could have thought that you should provide more of a programmable open system, even though there wasn't an independent software industry at the time?' We actually did make the XDE, the Xerox Development Environment, in such a way that it could be packaged up and shipped. And we did begin shipping it to customers about, if I remember, about 18 months after we launched the product. It was, however, designed for fairly serious, hard-core programming use, and we did not understand a lot about what it meant to export and support that kind of an environment. But, it actually was always a part of the plan to do that; we just did not put it in front of the other objectives. Later on, of course, some of the objectives for the program changed, but I generally feel that if it had happened earlier, we might have been more a part of jump-starting a better, earlier, independent software industry. The first few years of the PC software industry were pretty rocky, largely because of the tools and so on that were available. So, I see it as an opportunity partially missed.

Curbow: Actually there was [inaudible] in '83, this project called Phoenix, which was presenting the Tajo tool kit, Dave mentioned it, I think he mentioned today, he may have mentioned last time. There were times when we didn't have a tool kit installed, that the [inaudible] on the screen required everybody drawing on the screen. Well, in Tajo there's a tool kit for doing that. And, in the Phoenix project, a tool kit was built, originally, and it became Viewpoint, and Viewpoint was an open, tool kit based architecture, intended for third parties to develop software for. We had developed Mesa, and it required more learning that what most people had. About '84, '85 we started shipping Viewpoint to allow people to do just what you suggested.

Smith: I would like to make one more comment about the previous question, Charles Irby's question about the design methodology. Dave Liddle is correct, that before we did the design, we did the design for the design, and it's interesting that the principles that we came up with are no big surprises to people today. Anyone in the human factors business, anyone who goes to a SIGCHI conference or reads the journals, it's all pretty standard stuff. It was doing user testing. It's analyzing the tasks that users are doing, in their context, trying to figure out what objects they're using and what actions they're performing on the objects. And in fact we went out many times in the early days to various sites and decomposed in excruciating detail the tasks that people were performing, went back and did a design that would enable you to accomplish those very same functions, but hopefully in a much simpler and more improved way. I think it's very revealing from a historical standpoint that the SIGCHI organization formed after Star came out. The first SIGCHI conference was in 1982.

Question: I have some information that might be interesting to the speakers. I work at PARC, and just last week I had a visit from a colleague in Fuji Xerox, the Japan branch of Xerox. He told me that [inaudible] global dealer J-Star, and [inaudible] Japan is still active, and that Sony Corporation this year had selected it for a system need that they have. And then for good measure we opened up [inaudible] the system here, and the network icons, and I dropped a document on this guy's printer, at Fuji Xerox, in Japan, and it worked. So ... [applause]

Liddle: Speaking of J-Star and Fuji Xerox and so on, I very well remember when Bill English and I, and Joe Becker were there for the launch of the Fuji Xerox J-Star. Bill and Joe, of course, had been there in Japan several years getting that program working. And I was not very used to the way the literal translations from Japanese turn into English phraseology. After the launch, it was on the day of the launch, the booth was absolutely packed with people of all kinds, particularly competitors, people from other companies and so on, looking at the product. I remember one of the Fuji Xerox marketing folks translating for me the review that was written the next day in the paper. He said -- it was clear, in real-time he was thinking how to phrase this back to me, and he said, "The Fuji Xerox Star is as different from conventional computer interaction as clouds from mud." [Laughter] And I thought, boy, you can't do any better than that! [Laughter, applause]

Smith: Hang on just a second here. I think that while we're talking about the international features of Star, and how they were a big element in the sales that we did have, we should recognize the years of hard and very creative and dedicated work of the man who did, who was largely responsible for the internationalization of Star, and that is Joe Becker. [Applause] Without whom a lot of these tough issues, especially like making a selection in Arabic, which starts like this, and if there's an English word, it goes the other way, and you go back to Arabic and the selection will go the other, and he made it all work.

Liddle: I remember that. And then he said, "And what if the writer is writing in Arabic, and they're quoting something from Hebrew?" [Laughter]

Question: [inaudible] I've been very curious about what kind of tools were used to create these programs other than the traditional Teletype things, the old word processing, assemblers, compilers, and others. Were there any new environments for coming up with more real-time interactive, low-level machine language, native machine language code? The only things I've seen require you to stop the program before you make changes [lengthy comment/question inaudible]

Curbow: You could -- at some point where you realized you screwed up, you could back up and say, no no, do it like this, back up down the call stack, force the call differently, passing different variables, and go in and say, no no, skip over this and do that, and this was in '87, something like that. It's just amazing tools available that, according to my friends who are still writing code, tell me that they're just now starting to see this stuff in some of the new Microsoft products. I'm probably the most nerdy person in this group, and I can't answer the questions any more because I'm doing human interface. But you find like -- oh, this man right here, Charles Haynes, who was the debugger person, who put in [inaudible interruption] -- and other things. Talk to him afterwards, okay?

Question: How long were sales significant, and did the machine actually end up on the desks of high-level Fortune 500 executives?

Liddle: The question is, 'for how long were sales significant, and did the machines wind up on the desks of Fortune 500 professionals?' I can answer the second part. I was only here through most of 1982, and then went off and tried to put similar but different machines on the desks of Fortune 500 professionals. Not to print documents, but to do other funny data-like things. It is certainly the case that the primary customers were the biggest customers you can imagine. That is, it did indeed go virtually always to professionals. As a matter of fact, it was hard to get the sales force to quit trying to sell it to word processing people. Word processing people were proud of their ability to withstand pain, you know, [laughter] and to know all these complicated commands, and they were not the target. And it definitely went, in modest numbers I admit, to that very target audience. Who can say what was the period over which Star was sold on the various different platforms, including the Sun? I don't know if any of us have the right answer for that.

Question: I fell in love with your design, and I want to say thank you. But I also wondered, who invented the Undo command? [Laughter]

Liddle: The question is -- the question following a complimentary statement about our design – 'who invented the Undo command?' Now, as far as I know, Warren Teitelman invented it, Undo command in InterLISP, or its BBN LISP prior name, I believe that's true, and it was culturally very strong here in PARC, and so it was not optional not to have it in the system. I think it's fair to say that the ability to make text-editing documents un-doable flows from J Struther Moore's piece table invention, which was the thing that made it possible to have a text document in which you could still undo it, by not irrevocably committing the data structure. That, at least, would be my assessment. What else?

Smith: As far as Undo goes, we may as well come clean here. [Laughter] Dave's trying to get a picture of the keyboard -- if we could switch to the camera for a second. There is a key, labeled on the keyboard, Undo. It's one of the function keys on there. And we thought that, as designers, that was enough to ensure that it would actually be implemented. [Laughter] Unfortunately, in our naivety we were wrong, and it was never implemented, and Star did not actually have Undo. [Laughter, applause] However, we did have one thing that we did win, when it comes to Undo. We used to have a contest: can you find the sentence in the Star functional spec that causes the most implementation work? That was the sentence: Every command can be undone by pushing the Undo key.

Question: I understand that, when you talked about names for the product, that one of the original ones was to be Daybreak for the Star. How did the term Star, how did the name Star..

Liddle: The name Star was made up by Bob Spinrad, who was at that time Vice President of System Development. He was the head of SDD at that time. And he and I were once again trying to sell it to yet another skeptical senior executive, and we had just been put into the clutches of Dave Culbertson, who happened to be a sailing enthusiast. And so Spinrad said, "Let's name it after a sailboat class, a one-design sailing class. Maybe that will cause him to be interested in it." And [laughter] Star -- well, you know, it wasn't very easy to get Dave interested in things, and -- the Star was both a decent, interesting one-design sailboat and also a tolerable name for an office appliance. We looked at a number of other ones, but you know, somehow, Lightning didn't quite do it, Sunfish, everything else, but Star seemed okay. That's where it came from.

Question: Do you have any idea how many were actually produced? [Comment: "Daybreak" was the name upon launch.]

Liddle: Something like 30,000.

Question: About object-oriented GUI, and I'm a strong supporter of it, but recently you may have noticed there's something about backlash about object-oriented, but maybe it's sort of a muddling, would be a better word for it. With the longest perspective perhaps of most folks, on object-oriented GUIs, has your faith in object-oriented GUIs been shaken at all?

Liddle: Not badly enough to call them GUIs, but we know [inaudible]. [Laughter, applause]

Question: Out of curiosity, if you were to look at the descent lines from the Star, what would be the first, and maybe some of the second layer ones that you would consider to be obviously descended from the Star, at least in idea, mimetic concepts if you will. I'm thinking particularly that the Lisa might have been the closest.

Liddle: There were a few that were licensees, that is, there was a product done at Sun called, I forget, there was a toolkit done at Sun, under license from [interruption: Open Look.] Xerox -- Open Look, thank you, which was a direct descendent. Similarly, the product that we did at Metaphor, we actually had a license for as well, and it was a clear, direct descendent. Lisa had some similarities and some differences. It was more similar than the Macintosh was. Obviously there was, you should pardon the expression, Next STEP. And Motif. What was that? Yeah, the PERQ had some similarities and some differences as well. What else? Other guys can think of some.

Smith: Actually, this article by Jeff Johnson that Dave referred to, which are up here, has a very nice chart in it that has quite a dependency graph of Star and everything, so you're invited to come up here and take a look at that.

Question: A lot of hay has been made over a certain visit by Steve Jobs. He showed up here and looked at everything: whether that exists or not, maybe there's a story there or maybe not. I'm more curious -- especially in the last ten years -- about how Xerox invented it and everybody else got rich off of it. And I'm just curious, what was the -- what can you say about the attitudes and feelings inside Xerox, inside PARC? Watching these other machines, garbage or not so much garbage, whatever, making a lot of money. Was it the proud child of our ideas, was it the goddamn [inaudible]? [Laughter]

Liddle: We just thought it was a great idea. We just simply wanted our ideas to go out there in the world, and we weren't at all concerned with commercial success. [Laughter, applause]

Question: My name's Ken Pier. I started off before the Star, in that division, Dave hired me and I've been at PARC for about 20 years. I just want to say this in public: two of the inventions at PARC make Xerox over two billion dollars a year in gross revenues. Just two of them. So we're very proud of the money we've made for this corporation. And they are the DocuTech and the DocuPrint printers. [Applause] Dave, I thought you were going to tell one of my favorite stories, and I just have to substitute a couple of words. Remind me, that Star I think was unveiled at the 1981 NCC for the rest of the world, and the booth was packed, and the Japanese press was there, and they went back and wrote all about it. And when it was translated to me in just the same way, the translator looked at it and said, "This says that the Xerox Star disemboweled the competition." [Laughter]

Liddle: Yeah, I decided to leave that part out. That's true. Bob?

Question: Dave, I heard a rumor a long time ago that when the Department of Defense was designing Ada, they came to Xerox to ask if they could use Mesa instead, and we said no. Is that true?

Liddle: Bob's question was that he had heard that long ago, 'when the DoD had that high-order language thing on the street, what eventually became Ada, and that they came to Xerox and asked if they could, quote, have Mesa, and that Xerox said no.' I don't think that's quite correct. I do remember someone asking permission for us to give a briefing to the Ada Committee about a number of the features that were in Mesa, because we hadn't at that time really written widely about at least some of the features that were in Mesa. And we did give, we did sign off on that briefing happening. I think what you may have heard is that we were asked if we would like to take essentially an ARPA contract to, as it were, harden Mesa to be that Ada, and we said no, because we didn't want to do contracting. We were actually trying to construct these commercial products. But that's my recollection about it. Peter, do you remember anything about that?

Answer: That does ring a faint bell, that there was interest in Mesa as part of the Ada effort. I don't remember whether -- and I'm pretty sure there was some Xerox political reason why Mesa was not submitted as one of the candidates in the Ada competition. The person who would probably know the most about that is Butler Lampson.

Liddle: Yeah, and maybe [name inaudible], I remember her coming to me and asking how much can we ... [interruption: Dick Sweet, do you know anything about that? Reply: no more than what you said.]

Question: What about Niklaus Wirth, when he was at PARC he patterned a lot of Modula-2 after Mesa, is that correct?

Liddle: The question is, 'did Niklaus Wirth's visit to PARC, and his exposure to Mesa, have impact to how he designed Modula-2?' He certainly has said that, I think he's quoted that as one of the sources of his ideas.

Question: The Xerox [inaudible] from Star clearly made text-based interfaces obsolete, command lines obsolete, and hasn't really been bettered in 17 years. Some day it's obviously going to be obsoleted too, and I was wondering if you had any ideas or thoughts [inaudible].

Liddle: The question was, given that the Xerox Star and the interface approach that it sort of exemplifies and embodies still seems to be a leading form, and the strong successor to the old text-based interfaces, he was really asking about opinions about things which will compare to the Xerox Star as 'clouds to mud' [laughter], that is, what is it that will obsolete the graphic user interface?

Smith: I think the answer is pretty clear: it's Windows 98. [Loud laughter] As one of the designers of the Star interface, I am a little disappointed that in the last decade we haven't really pushed much beyond this desktop metaphor, with its documents and folders. I mean, even at Apple I was unable to get them to willingly go beyond this metaphor. It was just too successful for too long, and developed too much inertia. I would like to see it pushed in directions of information retrieval. One of the neatest things about the Internet are all the search engines that are out there. Those are just great. And there's no reason you can't do that on your local machine as well. And there's a lot of areas that we could make progress in, and there hasn't been sufficient innovation in the last decade, and I'd say that's pretty clear.

Garner: I'll second this as a hardware designer. You know, having the MIPS rate go from a half to five hundred, and seeing the user interface barely change has been an incredible disappointment. I remember, on the Star software, if you wanted to run the spelling checker, that was a big deal, and you waited a while. In fact, we had spelling checker servers. [Laughter] If this is the biggest change that's happened in ten years -- it's now PowerPoint does spelling checking while you're entering the characters. I predicted that would happen some day, I remember. So that's the only change I've seen now, in 17 years, is that spelling correction is as you type.

Question: How about Paginate? [Laughter]

Garner: Don't do that! Don't do that! [Comment: Actually Word still has pagination, it's call Fast Save.] [Laughter]

Question: [inaudible] I spent some time at Silicon Graphics working with the Nintendo game machine and technology and so forth, and even though I'm not a video game person, if you look at the number of children that are growing up with amazing skill at playing these very, very fast video games, and you think about some of the navigational technology there, and some of the methodology for moving around in the space, and doing things in that space, especially if it's interacting with you in kind of an entertaining way, it seems to me that there's a germ of a new UI paradigm in there somewhere. I don't know quite what it is yet, but it's certainly not a video game per se, but I think we can build on some of the same...

Garner: Certainly today we can build our microprocessors with things like the 3D acceleration for free essentially, so I'm really -- there was the InXight Group here, which I visited a little while ago. I would really like to see something utilize the transistors better on the chip for the user interface, and if it's 3D we can do it for free essentially, so we're at that point in technology that that's possible.

Smith: I think Charles actually identified, if anybody's interested in pursing it, I'm not in that business myself, but a direction to go in trying to push the interface forward: look at the games. Look at video games.

Garner: I want to see a prototype, because we're doing silicon, that we need. I'm just shocked that there's no university research or no research at all in this area, and I really would like to see a prototype, and, I'll give you the hardware if you give me the software prototype.

Question: Well what about all the VR [virtual reality] stuff that's been ... ? [Comment: It's a good start.]

Question: Could you talk a little bit about the invention of the Mesa programming language and why you decided to choose Mesa as the language for Pilot and Copilot and the Star?

Liddle: Sure. The question is, 'how was Mesa invented, and why did we decide to use it?' The other language alternatives that we had at that time fell into two classes: they were either very superior for quickly building prototypes, or even building large elaborate systems, but had the well-known performance difficulties that you would expect with these two very powerful systems we had here, namely InterLISP and Smalltalk. It was not practical obviously to do an implementation in those, although Smith stubbornly continued to prototype in them for as long as he possibly could. The other set of alternatives that were available at that time weren't sufficiently type-safe or didn't have the right set of abstractions and so on for building industrial strength software, of which you hoped to sell hundreds of thousands of copies. So -- that was primarily BCPL, C didn't exist at that time, that splendid language embodying the grade given to its designer. [Laughter] It wasn't even here in that moment, so it was BCPL or other very weakly protected machine-oriented languages. So, Mesa was actually quite a nice design. It had a few excesses in it that took us a little while to get out, but it gave us tremendous reliability in constructing this elaborate piece of software. And by God, you made a change to things and it started running again the way it was supposed to, and so on. This was a curious window, in the history of programming languages, and the idea of building a big software system that was nevertheless going to be replicated and sold very cheaply, so you couldn't ship a systems engineer with every copy, was really quite new. And that was the reason for the Mesa choice. Peter?

Answer: I'd like to just add a little footnote to that. There was a working group at PARC that met a number of times. It was a very high-powered group, looking at the requirements for a programming language for the work to be done at PARC itself. While that group was not directly coupled to the evolution of Mesa and Mesa's follow-ons -- in fact, Mesa had already been developed to a fair point by that time. There were ideas coming directly from PARC as to what a modern system programming language needed, that I think influenced the choice of Mesa very heavily. For example, the fact that Mesa has a very rich type system and is generally very type-safe. This was not a popular point of view for hard-core system programming in the late 1970s and early 1980s. I think it's fair to say that the Mesa artifact, the ideas in it, originated pretty much at PARC. And that it was the apt tool at hand when the Star was being programmed.

Question: I wanted to respond to the thing that Charles mentioned, about 3D interfaces being a direction to explore. It's amusing to me that the rage in video games right now all are three multiplayer network games. I recall many years ago Maze War being a very early 3D ...

Liddle: In fact, the best thing about Maze War, the thing I appreciated about Maze War, was that otherwise I never would have gotten people to be willing to test all that complicated internetworking code. [Laughter] No, I'm perfectly serious; it was a wretched testing problem but people playing Maze War absolutely tortured every line of that code, and I was tremendously grateful for that idea. Dick?

Answer: In response to the question about why Mesa, Dave Curbow was kind enough to take a paper that I had written back in '85 and scanned in with [inaudible] capture that I put on his web site, which doesn't really go all the way back in the history of this stuff Peter was talking about, but it has bibliography that does, and it's a fairly clear description of Tajo, the state of Tajo circa 1985, so ... I don't know if we could make the URL for that available. [Comment: We'll publish that.]

I can give another URL. [Reply: Go ahead.] If you just go to -- well, I will have to put it on tonight, but if you go to www.sweetshoppe.com, spelled s w e e t s h o p p e, that happens to be the web site that my wife gave me when she changed her domain name, I'll put a copy of the paper there.

Question: Not a question, just PARC trivia. I believe it is true that the Maze War -- all the programmers played it all the time, and of course we were all hackers, and we were hacking the sources, and the result of that, the authors got upset with everyone making cheats, and therefore they stored the sources in

encrypted form, on the source tree, and I believe out of all those software innovations that happened at PARC, the Maze War sources were the only sources that were stored in encrypted form. [Laughter]

Question: Did the Star ship with any hidden Easter eggs, do you know?

Liddle: The question is, did Star ship with any hidden Easter eggs that I know of. Let me assure you, not any that I knew of. [Laughter] But I wouldn’t bet that there weren’t. What else? Okay, thank you very much. [Applause]

END OF INTERVIEW

