Interview with James Mitchell

Interviewed by:
David C. Brock

Recorded February 19, 2019
Santa Barbara, CA

CHM Reference number: X8963.2019

Brock: Yeah. So, yeah, maybe we can talk about your transition from Carnegie Mellon to PARC and how that opportunity came about, and what your thinking was about joining the new lab.

Mitchell: I basically finished my PhD and left in July of 1970. I had missed graduation that year, so my diploma says 1971. But, in fact, I'd been out working for a year by then, and at PARC, you know, there were a number of other grad students I knew, and two of them had gone to work for a start-up in Berkeley that-- there were two Berkeley computer science departments, and the engineering one left to form this company called Berkeley Computer Corporation with Mel Pirtle running it and Butler Lampson kind of a chief technical officer and Chuck Thacker and so on, and anyway the people that I-- the two other grad students that had gone already that I knew, when I was going out, I was starting to go around and interview in early 1970 because I knew I was close to being finished, and it was in those days because there were so few people who had PhDs in computer science and every university was trying to build a department, and industrial research places like Bell Labs were trying to hire people and so on, anywhere you went with a new PhD you got an offer. I went to nine places, got nine offers.

Brock: Wow, what a great situation.

Mitchell: No kidding. It's taught me long ago that if I were a young guy today I'd be looking for the similar green field thing, where not a lot of work has been done, but you can see a lot of potential. Go there, because it will be a lot more fun no matter what you do. Since nothing was done before, whatever you do is better, right?

Brock: Right.

Mitchell: Right. Whereas today if you tried to do a PhD on compilers for example, oh, man, it's been such a deep science and everything, I don't think I could do it, and my first project at the University of Waterloo was a Fortran compiler called WATFOR that put them on the map; four of us undergraduates did it. So anyway, so I was coming out, and I was interviewing at Engelbart's group, I was interviewing at Stanford in the Bay Area, and so I added Berkeley Computer Corporation on it. Did I interview-- I didn't interview at UC Berkeley, partly because I knew they had left, and I didn't think much of the other group that was there, but I thought this one was good, and if they left I should look at them, and others around the country too, but basically when I came and interviewed there, I said, "Oh, man, these are the smartest guys I've found, and they're all smarter than me. That's good because then they'll up my game." Right? And it was people like Butler Lampson and Peter Deutsch and Chuck Thacker, Dick Shoup was there as well. Dick had been a grad student at Carnegie Mellon.

Brock: OK. He was your personal connection to the group?

Mitchell: He was one. There was another guy, oh, boy, I can't remember his name at the moment. Ah, senior moment.

Brock: We'll get back to it if you remember it.

Mitchell: So, you know, I did my final PhD defense in like June, there was a little clean up to do, and by about mid July my wife and I were driving cross-country, and I called the people at Berkeley Computer Corporation, and I had been actually sending them letters, this was pre-email of course, and telling them my schedule, and then, okay, so I'm on my way out, and I called them, where was it, it was Kansas or some place like that, and they said, "Okay, well, we'll expect you so-and-so, and when you get to town, please come see us right away." What's that about? You know. So like sure enough we got into town, my very pregnant wife, two cats and me. She found us a place to stay that would allow the cats, and I immediately went over to Berkeley Computer Corporation, and this would have been like August 1st or something like that, very early August, and what I'm told is, well, we're kind of getting short on money, and in order to make it so-- because we're close to having this machine ready where we could actually start having customers on what was called the BCC-500, which would have been-- supposed to support 500 people, and-- 500, you know, users simultaneously, and I thought, oh, geez, our baby's due in a month and a half, and now I'm finding out maybe I don't even have a job here or something like that, and they took me into the president right away, and he said, "Well, what we've done is basically we've gone on half-time, and we've told the government that we're doing half-time. This allows people to collect unemployment insurance, but, by the way, they're working full-time." That was a real-- that was a dodge. Anyway, and they said, "But you're full-time. We want you here. Don't worry, you know, we're working on the financing stuff, and we really need you here." So, okay, so I thought I'll stay, and so my joke is I was the last hired full-time employee. By January it was clear we weren't getting any more money. It was the Nixon recession, and we were not going to get any more money, and they were going close the doors. Meanwhile--

Brock: I hadn't realized the money was coming from the government.

Mitchell: It wasn't. It was venture capitalists. But it was a recession, and the venture capitalists didn't want to throw more money at it.

Brock: I get it. Bad prevailing economic winds.

Mitchell: But right around that time, that summer the Xerox Palo Alto Research Center had started over in the Industrial Park near Stanford, and one of the people that George Pake, who ran the place, had hired was Bob Taylor who had left ARPA, had been at Utah for only a year or so, and he got him to come and staff up a computer science lab, right? Well, we all knew Taylor, and he knew us. Basically ARPA supported, pretty much at all the major universities, the grad students with, you know, full freight, a living stipend and tuition because ARPA needed research done on computers, and if you don't have researchers the first thing you better do is train them, and that was their model. So Bob found, I think Butler, or Mel may have contacted Bob Taylor because they knew him well, and I'd met him, he came to Carnegie Mellon a couple times, and I'd met him at-- ARPA ran a graduate student conference every summer, I met a lot of these people there. That may be where I first got in touch with Berkeley Computer Corporation because Peter Deutsch was there at the one I went to. So were people from Doug Engelbart's group. Bill Paxton was there. I don't know if Jeff Rulifson was there. Charles Irby was. And Paxton actually still lives here. We're friends. We've known each other since then. Anyway, Taylor came over when he heard we were in financial difficulties and said, "Well, maybe there's a way," he wanted just

to buy the whole group, the whole thing. "Maybe we could buy this, Xerox could just buy this." Because he couldn't imagine any other way to build a team that fast, right? And he knew everybody. "This is like probably the strongest team in the country around the ability to build hardware and software, so Xerox should just buy them." Well, Mel Pirtle put up a-- he really thought the company was worth a lot more than it was, and so it came to naught, and finally on a day in January, on a Friday in January, the guy who ran the venture capital company that had been funding us came and basically said, "I love you guys. I think you're doing great work, but no more money, and we're going to lock the doors." <off topic conversation> And I'm sure there were communications going on that I wasn't aware of.

Brock: Sure. But that's when you got the news.

Mitchell: But Taylor basically said, "Come on over Monday morning to the Palo Alto Research Center, and we'll sign you on." No hiccup whatsoever. So Butler and I we carpooled because we both lived in a similar area, so we carpooled over, and literally left BCC on Friday and on Monday signed up with a substantial increase in salary even. Doesn't sound like much these days. I went from $14,000.00 to $18,000.00, but, hey.

Brock: Hey, that's a significant percentage.

Mitchell: Was what, in those days it was a sig-- Right. No, no. It was significant, and off we went. I think I was number six to sign or something like that. So a couple of people had gone there already ahead of time. Dick Shoup had gone there already. Peter Deutsch had gone there, but that day Butler and I, Thacker and the guy whose name I can't remember, we all basically signed up and away we went after--

Brock: What was the—if I, I'm sorry to-- I'm just so curious to know how Bob Taylor was presenting what the lab was going to do.

Mitchell: He said Xerox-- he spouted exactly what Xerox had said, so Xerox right now they copied marks on pieces of paper, and they said if we're going to survive, we're going to have to become architects of information, not just copy it, but we don't understand it, but we somehow-- the information means something, right? And he was pretty sure that that involved a number of disciplines and computers were a major part of that. So that's—

Brock: Sure. OK. That's the vision.

Mitchell: So he took that as-- and so we all talked about, you know, we're sort of here to invent the office of the future, and it's as good a place as any to push-- we all believed in interactive computing, even though in those days it was timesharing-based, right? And so off we went to start talking and thinking about that, and so that was like mid-January. These [Pendery Papers] are dated June of the same year, so Don Pendery came out I think around March or April or May, something like that, you know, what's this new thing that Xerox is starting in Palo Alto? We had a couple of different visitors, some from the corporate world that made copiers. Everybody was very suspicious of us. Well, we were an outpost. I learned this because I've been in outposts in other parts of my career. In fact, the president of Xerox Data

Systems, XDS, also visited us, and he didn't want-- this is when we were still starting to build MAXC, and he didn't want us to do that, of course, right? But we needed that operating system [TENEX], and he wanted us to build their next best system. So Taylor said, "No, no, that's not our mission" we're supposed to be here figuring out about offices in the future and all the ways Xerox could play in that, not just do another timesharing system, right? But I'm pretty sure we pissed off the president of XDS, and so Pendery came, he was in the strategy office, and he wanted to know, you know, give me some idea of where you think things are going to go and why. I still remember Alan Kay didn't like him that much because he was a very-- he wasn't a researcher, he didn't-- in those days computer researchers always, the really good ones, would take big mental leaps. They could see how things were going, and they go, we're going to be able to do this, we're going to be able to do this, you know, we can't do it now, but we can see how these things are going to go, and he was much more, "Well, tell me what the trends are in offices and everything, and we want to make sure we don't get blindsided." It was more of a defensive kind of thing, whereas Alan's there talking about the Dynabook and, you know, all that kind of stuff, right? Very, very different, and so Alan just thought-- Alan claimed out of that, that that was the point at which he actually said that, you know, the best way to predict the future is to invent it. I don't think that was original with Alan, but he reads a lot, so. It's a good thing to say. I've certainly used it, used it a lot, and that's how all that got going, and it was a good exercise for us to do because it forced us to talk to one another a lot and try to write these things down and so on, and we found we had an amazing amount of agreement about what sorts of things needed to be done, and then the only issue is how.

Brock: Interesting. So the papers were in a response to an initial meeting with Don Pendery?

Mitchell: Pendery came out and spent, I think he spent a week there, and he asked us about this, and he talked to a lot of us individually, and I don't know whether he came with that agenda item to get us to write something or that arose during the week, I don't remember.

Brock: Okay. And was there a subsequent meeting with him about the-- once you had written the papers?

Mitchell: I have no idea. I don't remember.

Brock: Okay.

Mitchell: I don't remember. He did come out some more times. I'm sure we did talk about it, but I don't remember like something really formal. The most formal thing we did, there were two things later in PARC's history that really made a big difference, one was it became pretty clear that the Xerox management and even most of its high-level technical staff from the copier business, they understood that, you know, the core of that hardware, of course, the whole Xerography thing, that's the stuff that really matters. When it came to the digital world and everything and how you should design stuff, they weren't as sharp, and it was just-- it wasn't that they were dumb, it was just they didn't work in that milieu so much, and we wanted to-- in particular we needed to convince the powers that be that the way this stuff was going to work was not that you build one-- a whole bunch of different pieces of hardware, but that you make different software to run on-- you will make hardware, but you will make software to run on

it, and the software, in fact, I can still remember Butler saying “hardware enables software, software makes systems.” So we actually put together a course, and Xerox high-level management came and had a three-day course, I think it was three-day, at PARC. We spent a lot of time trying to figure out how to teach them what the issues were, just the issues, so they had a sense of how this world was going to work, you know. In particular we pushed hard on the whole business of interfaces versus implementations, that what you really want is, there are good interfaces and bad interfaces. A bad interface is what all of the remote controls for garages use, to make it single button. The only thing that they could imagine doing in the early days was the frequency's on, and the frequency says to the garage door opener, "Change state." Which means if it's open, it'll close, and if it's closed, it'll open. Well, that's a bad programming interface because now you can't say to the garage, "Open." You can only say, "Change state." Well, if it's open, it's going to close, and it's been an ongoing problem, right? So that's a bad interface. There were examples from the Xerox world as well, like mating a collator to a copier, what's the right interface between those two things in order that it be reliable and you could change the copier and the collator would still work. You could build to a standard on both sides, and you could independently implement and innovate on each side of the interface, right? And so I would say that was the major message we gave them. The second big huge thing was what Geschke did? I think it was '79 [’78] or something, which was the Boca Raton thing with demos, and that was really good and it validated what we thought. They had like people from Fuji Xerox from Japan, and all those things, and they're setting up at that time all these Altos, and we had kanji on it, you know, we had Chinese characters, we had all these things, and they were really confused. It's like, well, how is this machine different than that machine? They're the same. If we take the disk out of here of this Alto, and we put it in there and run it, watch, see, and that blew them away. But Geschke did a huge amount of work on that, and John Ellenby I remember who later-- who's passed away sadly. Those two things had a lot of impact on Xerox. But let me come back to the outpost problem. There were parts of Xerox that just didn't trust us. We were not part of the core. This happens with mergers and acquisitions as well, and, in fact, while we're off, finally Xerox said enough had been created. I think it was around '77 or '78, and we actually formed the systems implementation group [Systems Development Division (SDD)] and started building a product [the Xerox 8010 or “Star”]. Now there were things, technically the product was wonderful, but what we lacked, we poor computer scientists was a good sense of marketing. It was too damned expensive, and it solved problems people didn't know they had. It had the best distributed email system for upgrading distribution lists and all that stuff. I still don't have something that good. But people didn't, they didn't want to pay for things they didn't know they needed, and if I give Steve Jobs some real credit it was when I first saw the first little Macintosh I went, "Well, this is underwhelming." You know, but it hit a price point, and it did enough of the right things, especially once Chuck Geschke and John Warnock went off and started Adobe and did PostScript to drive laser printer, the Canon laser printer. All of a sudden you could do things you couldn't do before, and I think Jobs had that part right, no doubt about it, and we did not.

Brock: Right sizing it, in a way.

Mitchell: Yeah.

Brock: I know that during your PARC career you would come to work on Mesa and Cedar, but I was really intrigued by-- it seems like you must have very early on been-- it seems like you were the bridge, the first bridge between PARC and Engelbart's lab. Is that earliest--

Mitchell: I was a bridge.

Brock: A bridge.

Mitchell: Yeah.

Brock: Could you talk about that connection?

Mitchell: Well, two things happened. One was, part of it was necessity and part of it was desire. The necessity part was that ARPA saw this group of ex-ARPA researchers all in one place at PARC, and they go, "You know, we'd like to be able to hire you guys as consultants and help us with things that we're doing." Right? "No money, just we want to be able to get your brain cells thinking about some of our things." And we were happy to do that and push their agendas, but they said “but in order to do that, you have to-- And we want you on the ARPANET." Because now email did exist, and that allowed us to do a lot more stuff. "But we need you to have a contract for this in order to get you on the ARPANET."

Brock: Oh! I had never thought of that.

Mitchell: So the question is what would the contract be. So Peter Deutsch and I came up with this idea of there was a new little, you could almost think of it as personal computer, it was called IMLAC PDS-1. It was a graphics terminal, it was very like an Alto in many ways. It had a calligraphic display, right? Not bitmap-- and we could hook a keyboard and a mouse up to it, one of Engelbart's, and the question was could we use NLS, their system, remotely?

Brock: Through this--

Mitchell: Yeah, over-- and we leased, this is funny, in order to get the speed we needed-- we leased a 56 KB line between SRI and PARC. Blindingly fast. <laughter> Peter did a fair amount of the programming, I did some, I wrote up the sort of proposal, but it's reason was that it was the smallest project we could do that would give us an excuse to be on the ARPANET.

Brock: I see, and the project was to see <inaudible> remote access--

Mitchell: How it would work to do NLS on a remote minicomputer, right?

Brock: Interesting, okay.

Mitchell: And we did learn some things doing that, and we got a report out at the end and that was good. Well, as I said, I already knew people over there, and we were starting to work on this programming

language we originally called MPL for modular programming language, and it grew out of the fact that Berkeley had a system programming language, SPL, and when I was at Berkeley Computer Corporation I ended up owning-- pretty quickly they said here's the file system, make it work, and that was all written in this higher-level language, SPL, but it didn't have what we think of as strong typing, and I noticed that an awful lot-- I'd be writing a little operating system routine for a file system, which is something like, you know, give me a piece of the metadata, like date last modified or something like that. That routine would look like about 4, 5 or 6 IFf statements checking that the parameters were not going to violate security. They weren't trying to give me an address in the operating system and trick me into writing in the operating system, little checks like that, right? Because this was supposed to be a security barrier, and then there'd be like a single assignment statement or something like that to give you the thing.

Brock: What you're actually going for.

Mitchell: Right. And I said, well, this is a waste. Can we possibly-- everything was about efficiency in those days, both space and time, because memories were small and computers were slow, and I said, "I can see that if we introduced some sort of strong typing here into this language, we could get rid of a whole lot of these checks." Less code and things run faster, you know, and you can't make mistakes, or there's a certain class of mistakes you can't make anyway, and so as soon as we got to PARC I said, "I want to work on developing a new programming language for modular software and that takes the lessons from SPL and makes it better." That's why it was called MPL. Well, it turned out at the same time for the same reasons my friends at SRI were also considering doing a similar thing because they were seeing the limits of the-- they were just using BCPL, I think. I'm not sure what they were using. They might have even been using assembly code for a lot of it, and we got together and just talked a few times, and out of that grew this idea of, well, why don't we do a joint project. We didn't even have MAXC yet, right? Well, you've got PDP-10s over here and NLS, which we believe in, that'd be a great place to work to do this MPL, and we can do the design together, people from there, Bill Paxton was one of them, and that's how that got started, and we got to the point where it was pretty much all implemented and things. But the people working for Engelbart were getting upset because he was so directive. He was the architect, and you had to do what he said and everything, and at some point in there when we really had-- we had this thing working, and--

Brock: The language.

Mitchell: The language, working on the PDP-10, and we were seeing some of the results, but there was more work to do. So it started looking like a good thing, and I don't know who talked him into this, but suddenly Doug Engelbart said he wanted to have a legal agreement between PARC-- we were doing this on [a] handshake, and no one's, you know, it's all ARPA, don't worry about it, right? But he brought the SRI lawyers in and then the Xerox lawyers got involved, that absolutely-- the day we had a signed agreement was the last day we collaborated. It killed it. It just killed it, right? And it was traditional lawyers who didn't understand this kind of thing, and when that happened, by that time we now did have, we were starting to have Altos and things, and so it was like let's make this work, and it'll be our programming language we use at PARC, and that became, so it got renamed to Mesa at that point, and we started implementing it, making it available on the Altos. It took a few ye--

Brock: It had these same features that--

Mitchell: It was the first language I know of that actually-- you separated interfaces from implementation. There was an interface language, and then the implementation language, and you could define all the interfaces, say, for some system you're building, and this guy would go off to build it. He could use the compiler and manufacture calls via those interfaces, even though there weren't implementations yet. So he could at least get through the compilings and stuff and so on, and you could run a project pretty much-- if someone found a bug, they fixed it, the interface didn't change, everything just kept working, right? It's just that you got rid of bugs that way, or you made it perform better. So it was actually doing amazingly well. I made one huge mistake, and Alan Kay would agree with me, it really should have been object-oriented. So Smalltalk was at the same time going on and being just completely object-oriented and everything's dynamic and figured out at the last second, no strong typing or anything like that ahead of time, and every time I tried to figure out how to implement that really well in a compiled language, I didn't like the efficiency hit, and so I backed off of it. Biggest mistake of my career. If I had made Mesa object-oriented at that time, then a language I worked on much later at Sun, Java, wouldn't have been needed. We would have had Java in like 1976 or '77.

Brock: Wow, that's kind of hard to imagine.

Mitchell: That was a big mistake on my part.

Brock: Was that just about the imagination of the-- as you say, you know, is like time and size, these matters of efficiency, so I guess--

Mitchell: That was my problem. I wasn't thinking out of the box enough.

Brock: Well, that's interesting because I've heard-- Well, that's all very interesting, but also that it was a collaboration on this programming language that was the connect to Engelbart's group, not, you know, because I and other people were imagining that it was some sort of move to commercialize what was going on over in Engelbart's lab. No?

Mitchell: No.

Brock: Not at all? Fascinating.

Mitchell: If there were discussions like that going on in Engelbart's lab, I don't know about them. He did have the whole issue of, you know, he'd done all this great thing and had the great demo and all of that stuff, and I think ARPA was trying to figure out what to fund him about, and that's when he took on this task, which I think was a mistake on his part, to be the Network Information Center.

Brock: You think that was a mistake because it muddied the waters, or?

Mitchell: Because it led you to thinking that there was an information center rather than just a distributed-- like the internet today, there's no center, right? And that's the only way it can work and be something that can expand. But Doug very much had this notion of, you know, he was trying to augment knowledge workers, right? It was the augmentation system, he was trying to augment knowledge workers, but he had this very centralized view of he would be the architect in the center of all of this, and he always hated HTML and the whole web thing, Doug never really liked it, and, you know, because it's just everybody gets to do their own thing, you agree on--

Brock: The interface.

Mitchell: HTTP is an interface, right. Exactly. And then people went off and just implemented their own stuff, and Doug was never comfortable with that. He really didn't like it. He should have taken great pride in what he had-- the direction he had set and that people were going that way. But he actually got a little bit bitter about it. I remember talking to him in the later years. That's too bad, but.

Brock: So what was the--

Mitchell: But that was the connection. It was not commercialization. We all knew about him, and we knew about the demo, and we really liked that. I mean, we adopted the mouse, we hired Bill English and then made a whole series of mice at PARC, we hired Charles Irby and Bill Paxton and Jeff Rulifson. In fact, right up until the time I left Oracle, Jeff was reporting to me. So we knew all about that, so in that sense there was a lot of connection, right? And we knew-- in fact, if you noticed, the Altos had a five-finger keyboard. I had to jump up and down and argue for that really hard because I had gotten good at using Doug Engelbart's NLS system, and I said, "This is really nice. I can edit so fast." When it's just fix the spellings thing or do this or that or the other thing, you never have to move your hands, right? But it never took off on the Alto and just died away.

Brock: So do you think it was these issues of control that led Engelbart to want to get everything legal and bring in the different lawyers about the--

Mitchell: I don't know. If I had to guess I would say it was more likely the lawyers who instigated this.

Brock: Oh, okay. So I noticed that in your contribution to the Pendery papers that you mentioned that you wrote it using NLS. Was that using your project to access NLS remotely, or was that your writing it over there?

Mitchell: No, we hadn't started that yet. We were just in the begin-- No, no. By this time I was already working with them on MPL, I had an SRI badge, I could come and go as I pleased, and if I was going to write something I was going to have a good system to write it on.

Brock: That's what Alan Kay recalled about his piece that he went over to Stanford to the Artificial Intelligence Project right as, you know, just using their system.

Mitchell: Because we didn't have anything. I think we were just buying-- for a lot of experimental stuff, we were buying Data General Novas.

Brock: Right.

Mitchell: Right. Minicomputers. So we could get some stuff done. There was a BCPL compiler on that, I think Peter had-- Peter Deutsch had a lot to do with that as well. But there began to be a diaspora out of Engelbart's group.

Brock: Yeah, did that come in--

Mitchell: And they tended to come to PARC because we had good salaries and there were a lot of smart people there and it was kind of a gravity well, you know.

Brock: What did you think of the effort to do the POLOS, that kind of distributed NLS effort.

Mitchell: Well, from what I remember about POLOS was that this notion of more or less having terminals and then having the computing all centralized in racks, but doing the graphics there, but for every terminal that had its own graphics board and everything. So it wasn't a personal computer, but it made sure you got a lot of bandwidth for that, and it was a hub-and-star system, and basically once you have Altos and just Ethernet just going around, that's a better way to-- because if you want to add one, you add one, and the POLOS thing, if you get to a rack that's full, now you've got to think about a whole new rack, and it was not a crazy thing for them to do, but the way the hardware trends were going was against them. It's almost like a half step from a timesharing system to a personal computer.

Brock: Now had you known Chuck Geschke at Carnegie Mellon?

Mitchell: Yeah, at one point we were sharing an office.

Brock: And then at some point he joined the effort on Mesa.

Mitchell: Yeah, we hired him-- Right. He did. He joined-- probably the strongest engineer on the whole Mesa thing was a guy named Ed Satterthwaite.

Brock: I've seen that name.

Mitchell: I think I credit Ed with this idea of we knew we wanted interfaces separate from implementation, and we weren't quite sure how to do that. He said, "Well, let's just make one of the things an interface language, and it looks just like the other one except that you can't put executable statements in it, and it starts off with a thing that says I'm an interface." And then he really wrote the bulk of the compiler and everything. Chuck helped with runtime and stuff, I did too. I'm trying to remember who implemented it. Implementing things on the Alto, Alan Kay, I still remember, said exactly the right thing one day, he said, "Since the Alto's a microcoded machine, if you want to make the densest code you can, have a machine

that's bytecoded." And what he meant by that was the right way to do this in firmware is, you know, someone says go execute this program, and the firmware goes there and picks up a byte and it jumps via a 256-way table to the routine that emulates that instruction. This is the easiest way to do it, and you can't make smaller than a byte sensibly, and we built it around that, and that bytecode if you look at the way later at the Java bytecodes, they're very similar.

Brock: That was a direct line.

Mitchell: Right. So it makes things smaller by a lot, you know, I mean, a PDP-10 had a 36-bit word and that was what an instruction was. Well, we could get four and a half instructions in the same space, and on the Alto that really mattered a lot. Now that machines are not microcoded anymore it doesn't matter so much, but it turns out if you have a world where you're doing both interpretation and compilation, software emulation and compilation, and all the things that you hear about like with Java and so on where they talk about HotSpot compilation, that's what that is, it's you keep the code in this bytecoded form because it's very compact, but you interpret it in software and then as a side effect of interpreting a particular instruction eventually you're going to, in the interpreter go by a couple machine instructions you've set up to do whatever that was supposed to do. Well, instead of just running by them and executing them, take them and write them over here, and gradually what you'll end up with the first time you go, say, through a loop in a program, you'll end up with compiled code. The next time through, if nothing has changed-- remember programs are data, so they could change, but if no one's changed it, just execute the compiled code.

Brock: Oh, so that solution--

<overlapping conversation>

Mitchell: So that idea, that's actually my PhD thesis.

Brock: Okay.

Mitchell: But Python, all of those things use that technique these days. That's how they make them fast. Still fully interpreted. You can change any line in Python and just keep going, and it knows-- the key part of this is how do you back out when you got compiled code but it's no good anymore?

Brock: Okay, when you have changed--

Mitchell: When you've made a change that affected that compiled code, and we won't go into the details. It's pretty easy actually, but you've got to recognize that problem has to be solved, and once you've done that it worked well. The other thing that it really helps is, when you build a debugger for a system like Java, you only have to ever know about the bytecodes, because if something goes wrong and the person hits a break point or something, they want to go look at something, throw away the compiled code; go back to the bytecode, because the bytecodes are machine-independent, and so you can move the debugger to any old machine, and it works just fine.

Brock: Oh, I see. Right.

Mitchell: Right? But the compiled--

Brock: You take it to the invariant part.

Mitchell: Right, exactly, exactly.

Brock: And just deal with the invariant piece of it.

Mitchell: Yep. So it helps a lot with that.

Brock: Okay, fascinating. Well, how do you see the relationship when you were working on MPL at this early stage? What is the connection to the kinds of things that were happening with Unix and its kind of modularity? I don't think it had--

Mitchell: Unix is not very modular.

Brock: Yeah. Not in that way, right?

Mitchell: But one of the best things about Unix-- the basic structure's lasted pretty well and been re-implemented a number of times-- was honestly the C programming language. Which was just-- the C programming language was very like the SPL I used at Berkeley that made me go and try to do MPL. But C has lasted a long time.

Brock: In conception, you don't see--

Mitchell: And Unix was a real operating system. MS-DOS was not a <laughs> real operating system, right? Unix was a real operating system that understood about crossing between user mode and kernel mode. You could build a timesharing system in Unix, right? That's kind of why all those things had this kernel mode and user mode and stuff was users couldn't compromise the system because they'd screw up for everybody.

Brock: Well, a couple questions back on the Pendery Papers. What do you feel that their importance was?

Mitchell: I would say their most important thing was it got all of us who were working in PARC thinking kind of in the same direction. That's really what I think it did. I meant to read John Urbach's thing again. I started reading it, and I realized what he was basically doing was being the good incremental scientist of "Here's how things are. Here's how I think things are likely to go." But he wouldn't have talked about networks. <laughs> And Bob Taylor did understand these things even though he himself was a psychologist, right?

Brock: Yeah.

Mitchell: And that's how we got Bob Metcalfe to come, and that led to the Ethernet. That was one of the more brilliant hires frankly.

Brock: Yeah, yeah.

Mitchell: And another thing that stood the test of time by being extremely simple.

Brock: Right?

Mitchell: And there were fights about what was the right architecture for a network in those days. Cambridge had a thing called a fixed slot ring, Cambridge ring, University of Cambridge. IBM was pushing a token ring. Someone else was pushing a token bus. And Metcalfe pointed out what happens in a token ring, if for some electrical reason you lose the token, how do you get it back? And the answer is you essentially revert to being an infant Ethernet, and some one of the machines on the local net says "Things have gone on too long. I think the token might be lost," and it starts doing something, and you just sort it out by who's got the highest address or something like that, so you end up with only one token, not two. But while you're doing that to generate the token, you're essentially doing an Ethernet. There's no token, and so you're doing exponential back-off, and all you're doing is sending a short message, which is the token. And he said "So you got to implement an Ethernet anyway, so why bother with a token?" You got to convince yourself that having the token-- and IBM believed this-- that having the token allowed you to have more reliable communication, because what happens if you saturate an Ethernet? Is it maybe going to do one of these things like a highway where you get too many people on it and in fact the flow drops way, way off? And a lot of work went into that to show, by a guy named John Shoch, who's been a venture capitalist for a while--

Brock: I know John Shoch.

Mitchell: Right. He did some of the early work that showed, no, no, you can get up to about 97.5 percent, and it kind of rolls off, but it never collapses to a lower bandwidth.

Brock: Were the Pendery Papers widely circulated within PARC, or was it amongst the authors mainly?

Mitchell: Within the Computer Science Lab everybody got them, right? It's a good question. I don't know whether it got read in the physics groups. I don't know.

Brock: And Taylor must surely have had to have signed off on it if you were sending it back east.

Mitchell: Well, the process I remember was I wrote a first draft, and I distributed it around to just anybody that wanted it, and I got a lot of input, and I revised it a number of times. Bob may have been involved in that. I don't remember. But, no, once it was done, it was done. He didn't--

Brock: Reading it, it seems to just comport with his vision, yeah. And why do you think it seems especially meaningful to Alan Kay, the Pendery Papers?

Mitchell: Well, it was the first time he wrote down all the stuff about the Dynabook book I think.

Brock: That's what I was wondering. I have to follow up with him.

Mitchell: In his PhD thesis he had written about something different.

Brock: Flex machine.

Mitchell: It was almost a pipe organ, <laughs> which he does play, right?

Brock: Right.

Mitchell: But somehow during this time his thinking had advanced, and it was his chance to put that down and talk about what you needed in terms of display and all of that kind of stuff, and it should weigh less than this, which is pretty funny these days. It should weigh less than five pounds, and anybody who had a five-pound laptop would think it was just enormously heavy these days. <laughs>

Brock: And in your piece, I really noticed the strong, well to me, I heard strong echoes of the sort of language of Engelbart's lab, augmentation, most centrally.

Mitchell: Absolutely. Yeah, we were heavily influenced by that.

Brock: Could you share your thoughts about him and his approach and how that had influenced your thinking?

Mitchell: Well, as I said, I seriously considered going and working there-- if I hadn't gone to Berkeley Computer Corporation, I would've gone to SRI, because I just loved what those guys were doing. As soon as I saw the-- now I've forgotten was it-- I think it was the Fall Joint Computer Conference in '67 ['68] in San Francisco where they did the demo.

Brock: Yeah.

Mitchell: And that just blew me away.

Brock: You were there; you saw it.

Mitchell: No, I wasn't there. I saw a video afterwards.

Brock: Of the film or whatever that they made.

Mitchell: Right.

Brock: Yeah.

Mitchell: It was clearly the way things are going to go. It's the right direction. Whether we all have mice and do this and that, I don't know, but--

Brock: The general--

Mitchell: Having the whole thing there, and it wasn't quite What You See Is What You Get on the screen, but it was very much different than the kind of-- we were all using line-- we might be using displays, but we were using them like they were teletypes, right?

Brock: Yeah, right. Right.

Mitchell: And that heavily-- That's why we did the Bravo document preparation system that became Microsoft Word.

Brock: Right.

Mitchell: All influenced by that, but we thought we could now make it industrial strength and just move the ball quite a ways along. And having the Alto and the Ethernet and the first laser printers, that's all you needed to get everybody going "Okay, what are all the things we can do with this?"

Brock: And before I had the chance to turn on the recorders, we were talking a bit about the decision to make the high-resolution bitmap display, which I think you said was Chuck Thacker's unique contribution—

Mitchell: As far as I know, I remembered-- <phone rings> oh, geez, who's trying to get in touch with me?

Brock: Want me to turn this off?

Mitchell: No. I can call back.

Brock: You sure?

Mitchell: Yep.

Brock: Okay. Bitmap display.

Mitchell: I'm trying to remember the exact conversation, but I know I learned it exactly, and I learned it right from Chuck. It could have been-- We had these weekly meetings every Wednesday that we called Beat the Dealer, and it was for half-baked ideas, and Chuck's was describing their-- while the Alto was

being designed and built, he would talk about it in that forum, and I believe that's the first place that I heard <clears throat> about a bitmap display. And there were people who understood graphics in the room, like Bob Sproull and Dick Shoup and so on. And basically he said "I don't know how to implement the right set of primitives, so I'm going to make it as simple as possible. Just be a part of memory, then you're going to read out of that memory, and that'll be turned into pixels on the display." And everybody thought "Well, that's really simple, but, yeah, we could do anything we need to that way." <laughs> And we had already been experimenting with the laser printer I think, and I think one of those POLOS systems was the one thing that got modified to be the EARS system.

Brock: Right.

Mitchell: So that had to deal with fonts and how you stored fonts in memory and everything, and of course if it's a bitmap display, literally it was just black and white, so it was ones and zeros only, right?

Brock: Right.

Mitchell: And so that meant a font was ones and zeros, and you BitBLTed it over here to paint an A or something like that. And I think that was in a Dealer meeting where he first talked about that. Let's see, what else did he-- a little side thing.

Brock: Yeah.

Mitchell: While I was at Carnegie Mellon, my undergraduate degree was in math and theoretical physics, and I'd had a lot of math. But they wanted me to take some more algebra courses, and I took one, and I went "Oh, geez, I've already had this." Then they wanted me to take another algebra course, and I went to Alan Perlis, my adviser, and I said "Look at my transcript. I've had six algebra courses over my career, and I don't think I need another algebra course." But there's a School of Fine Arts at Carnegie Mellon. I said "I think I should go learn about calligraphy, because we both know that computers are going to be involved with creating documents and things, and we should understand fonts.

Brock: Really? Wow.

Mitchell: I just really wanted to do calligraphy. <clears throat>

Brock: Well, good story.

Mitchell: But he accepted the argument, and so off I went to calligraphy class, which was just great fun. I still have some of the stuff that I did, and I can still do pretty good calligraphy. But one of the things in there is the so-called problem of letter spacing. How do you space letters so that words hang together well and everything? We call those proportionally spaced fonts as opposed to fixed pitch fonts these days. And the guy who taught the course was this great guy. He'd been the chief graphics illustrator at Look magazine for a long time, and so he was an artist, right? And he tried to describe to us how you would do this, and after I listened to him for a while and I started playing with it, I said "You know what?

You can do this almost algorithmically." So I figured it out, and I started doing it algorithmically, and he came around and was looking at us, and he said "Jim has got it." And he's showing the class. "Jim knows how to do letter spacing." And he said "How much practice did it take?" Well, I said "Not much practice. I just had to work out the algorithm," and he goes "What does that <laughs> mean?" And I told him the algorithm, and he did not want to believe that's what it was.

Brock: Wow.

Mitchell: But I demonstrated it to him, and he went "Well, I think you might be right <laughs>." And so when we did EARS, we can now do proportionally spaced fonts. How do we do that? I said "I know. I got the algorithm." And the algorithm's pretty simple by the way. Let me show you. Think I got a pen in here somewhere.

Brock: I do right here.

Mitchell: Here we go. So the deal sort of is your eye wants to see kind of equal spaces between things, so let's do a word like "hello." I'm not spacing it properly. Or a word like "doom." If you're letter spacing these things, when you have a curved thing like this next to another curved thing, you get big spaces in here. You want them almost touching, right?

Brock: Okay.

Mitchell: Then there are open things like--

Brock: Because of the distance between the top of the circle--

Mitchell: Right. Your eye kind of judges the space, right?

Brock: Yeah, yeah.

Mitchell: If it's an open thing, if it's a C and an O, say, well, it's the same thing. They have this <inaudible> If it's something like an L, there's a little more space. And the trick is, once you realize it has only to do with the shape of the curve on that side for the letter, then all you do to make the font is, for something like an O, because it's curved on both sides, there might only be one pixel worth of spacing, whereas with the L there might be two on each side, so you just include it with the font. Now you just take them and BitBLT them; you get automatic proportionally spaced fonts-- we never patented it. I can't imagine what that would be worth. We didn't patent those things in those days. Now, that gives you a quick-- if you looked at what Don Knuth did with Tex and everything, that is actually more sophisticated than this, and where this might be I would call it a 90 percent solution, his is the 100 percent solution.

Brock: Okay, which is a much more complex algorithm.

Mitchell: But it's much, oh, much more, because this doesn't do kerning and things like that.

Brock: Okay, yeah, yeah, yeah.

Mitchell: Right, right. There was a time I knew a lot about fonts. <laughs>

Brock: Everybody seemed to. Well, when you're developing the capability to handle-- I think it was in your Pendery Paper essay, too. It said something about saying that people are going to be spending a lot more time in front of these screens, and it has to be--

Mitchell: Oh, yeah, it has to be high resolution and--

Brock: There also has to be a delight or some word like that, and also talking about sort of book-grade typography.

Mitchell: Yes, right.

Brock: It's interesting how solving that problem exercised computing so much.

Mitchell: Yeah, it did.

Brock: So that really pushed--

Mitchell: And it was one of the hardest things at PARC. Both Alan and I and others, we were always-- the physicists would ask us-- who were doing things that "What would be good for the future?" And I said something like a liquid crystal display, because there were small ones then, that had this kind of resolution and was this low power and everything. And they would say "Well, why would you want that?" and we never got them to work on it.

Brock: Yeah, interesting.

Mitchell: It was hard to change their mindset about-- it was kind of funny in PARC that the Computer Science Lab, maybe because Taylor, when he went to budget meetings, he was pretty tough with his peers. And he was the nicest, the best boss I've ever had, so his downward managing was superb. He got the best out of groups of people and everything. But with his peers and up, he was pretty bad. <laughs> And we would give him a hard time about this, too, on occasion <laughs>. It was like "Come on, Bob, you can--" <laughs> because he taught me one thing, which is "Don't talk about efficiency. Don't talk about a lot of things. Just talk about whether you're being effective or not," and we would turn that argument on him and just say "When you get mad at those guys and then they just get their backs up and they don't want to give you any more budget money, how effective are you being then?" <laughs> Very sad to lose him. We were friends right up until he took the legal way out. But, yeah, I admired him a lot. That's it. <laughs> That's a little aside on fonts.

Brock: Well, let's see. Just one last question about the Pendery stuff. Did you ever get sort of a reply from Pendery about the set of works or no, no reply?

Mitchell: Not that I ever remember.

Brock: But nothing prevented-- he certainly didn't impede things from proceeding--

Mitchell: No, as I say, I think the best thing to come out of this was all of us saying "Yeah, yeah, that's what we're doing," and kind of heading off in that general direction and seeing what would come of it and then Taylor pushing the Alto and getting Metcalfe for the-- and doing the local area network. That and then the laser printer, those were the things that all of a sudden, phoom, away you go.

Brock: Right.

Mitchell: One little anecdote-- I don't know if you know this. Who do you think invented the whole idea of multiple windows on the screen?

Brock: Well, this is a subject of some controversy I suppose. I've heard-- I don't know the answer to that I guess is what I would say.

Mitchell: But I'd like to hear what you've heard.

Brock: Well, one thing that I have read and have asked people about but not gotten good information about was evidently there was a provision for windowing on the Stanford AI—the SAIL system or one of them that they called pieces of glass that were--

Mitchell: Like virtual displays or something.

Brock: Text windows if you will, and I'm not sure if they could overlap or not, but they were essentially transparent, so if one were to pass, you would see through--

Mitchell: Oh, see through it. Yeah, okay.

Brock: And Les Earnest makes--

Mitchell: Do you know the date on that?

Brock: Well, it pre-dated PARC.

Mitchell: Interesting.

Brock: Him talking about it is all that I've seen about it, and I haven't gotten good information about somebody who used it or who implemented it. So that's one of the contenders I have for the--

Mitchell: Interesting. That one I hadn't heard.

Brock: Yeah, I don't know much about--

Mitchell: So Doug Engelbart claimed to have done windows, and they could have multiple horizontal areas on the screen, so you could have two documents open like this, and you could copy and paste from one to the other.

Brock: Oh.

Mitchell: And so he said, well, that's like windows. Well, half, right? But the AI thing is interesting. I should track that down. At PARC, they were building the Smalltalk system, which was very elegant in many ways. Very small amount of stuff in there, and you bootstrap-- the rest of the stuff is written in Smalltalk itself. Well, one Monday morning when we came in, there was a demo available of these multiple windows on the screen, just like what we're used to, right? You could grab them and move them around and make one come on top and all of that kind of stuff.

Brock: Yeah, yeah.

Mitchell: And that had been a little weekend project of a woman whose name was Diana Merry.

Brock: I've heard Diana Merry's name. Alan Kay was talking about Diana Merry.

Mitchell: Well, she was the one who did it over the weekend. I don't know whether she got the idea from someone like Sproull from the AI Lab or how, but I know she did it in Smalltalk, and many people when Smalltalk-- that was one of the things that they mentioned. And Diana Merry, by the way, was transgender.

Brock: Oh, I didn't realize that.

Mitchell: I didn't know that at the time either.

Brock: A trans woman.

Mitchell: Yeah, she'd been a man; now she was a woman.

Brock: Okay.

Mitchell: One of the other people that has done really great work in the VLSI area--

Brock: Yeah, Lynn Conway.

Mitchell: Is Lynn Conway, right. I still interact with her via email, and I think it's interesting that those two men becoming women just did pretty amazing things. And even Lynn has written this recent article about how, if you're a woman, you don't get the recognition for it, right?

Brock: Yeah.

Mitchell: And Diana certainly didn't get the recognition. I'm glad Alan mentioned her, because may--

Brock: Yeah. I had made a note to myself to track her down, because I think she left computers, as I recall, and gone into some other field.

Mitchell: Wouldn't surprise me.

Brock: It's interesting. And was that after Dan Ingalls made his--

Mitchell: So that was Smalltalk.

Brock: Was that before or after BitBLT?

Mitchell: Well, BitBLT is early on. It used BitBLT.

Brock: Okay.

Mitchell: But that whole idea that you could easily abstract and have not just one window but many windows-- they're just objects after all, right?

Brock: Yeah.

Mitchell: And they just had an X-Y position and a size on the screen, and that's all there was to being a window. And I credit Diana with that, because they didn't exist on a Friday, and Monday there was a demo. <laughs>

Brock: Okay. Well, I will track that down.

Mitchell: And she had been in all weekend doing it. Yeah, if you could track her down, I think it'd be good. She deserves recognition for doing that.

Brock: I will. Thank you for mentioning that.

Mitchell: What a day. God, it's gorgeous out here.

Brock: Absolutely.

Mitchell: It's too early for the grey whales to be coming by.

Brock: I was wondering about--

Mitchell: Early April, late March they'll be coming by. In the summer here now-- I don't know whether it's global warming or what, but-- the way the channel is, it's very deep out there, and we get this great upwelling of plankton and everything. And so in the summer here, there are whale watching cruises, and there are blue whales here.

Brock: Wow.

Mitchell: There are finbacks here, which are the next biggest to blue whales. There are humpbacks here. There are minkes. There's orcas around, especially when the grey whales are going through, because they're after the calves, and of course loads of different kinds of dolphins. In fact, one day I was somewhere out there on a cruise, and we were actually heading back, and all of a sudden there were all these dolphins around the boat. Great fun.

Brock: Yeah, sure.

Mitchell: And the captain knew just what speed to run so that they got a nice bow wave, and we could stand up there and watch these dolphins just having the greatest time, including mother and calf riding the bow wave, and they're having just a great time. And they were all around, hundreds and hundreds of them, right?

Brock: Wow.

Mitchell: All of a sudden it was like one of them gave a signal. They all went to double speed. Literally it was like, phoom, off they went, and they head off towards the horizon, and they formed a line that was I would guess close to half a mile long.

Brock: A single line.

Mitchell: I asked a friend of mine who understands about dolphins and things, and he said "Oh, yeah, it's called a chorus line." And what's happened is that one of the dolphins has spotted a whole bunch of herring or sardines or something, and by forming in a solid line like this, it looks infinite to the fish, and they can swim faster than the fish, but the trick is to get the fish all together into a bait ball. And so they go out like this, and they chase them, and the fish start running and running and running, and then gradually what they do is they start--

Brock: Flank them.

Mitchell: Flank them in like this until they're contained, and then they've got them in a bait ball, and then they eat every last one.

Brock: So--

<overlapping conversation>

Mitchell: But to see them go double speed and just <makes whooshing noise> off towards the horizon was just quite amazing.

Brock: That must've been spectacular.

Mitchell: It was. Good place to whale watch.

Brock: Yeah. Well, I don't know what your time constraints are.

Mitchell: I'm going to skip dessert by the way.

Brock: Yeah, sure. Do you have few more minutes?

Mitchell: Sure.

Brock: What's your time situation like? It's about two-thirty now.

Mitchell: I'm fine.

Brock: Okay. Well, I would be curious to hear more about just how you relate your experience with, as we talked a little bit about, the kind of Mesa-to-Java connection and about that project. But unless I'm mistaken, you did a lot of very interesting work in between there, one of which was your involvement with the whole ARM development, Acorn and ARM.

Mitchell: Yeah, the longest I've stayed at any one place-- I guess ultimately the longest place was Sun.

Brock: Yeah.

Mitchell: But before that, it was PARC.

Brock: Yeah.

Mitchell: And I remember telling Bob Taylor when I first went there "I'll probably stay five years, but then I'll get comfortable, and then I won't be as good, and so I'll move on." He said "We're going to make it so good here you'll stay ten years," and he was more right than I was.

Brock: Yeah.

Mitchell: So when the diaspora of PARC happened, which was a result of putting a new guy in charge of PARC who ended up being kind of an idiot in terms of management-- he was a very traditional manager. So as a result of the way Taylor behaved both with peers and upward, because he really believed in this stuff and he was passionate about it, he couldn't understand why people wouldn't see that they should be investing in it and going fast. And it was one of the ways Xerox fumbled the future basically frankly, by

not seeing that they had to go faster. And so he left, right? And the day he left, when he announced he was leaving, Chuck [Thacker] immediately announced-- I think it was in a Dealer meeting-- Chuck immediately said "I'm out of here." And <clears throat> I don't know when he had been talking to Digital Equipment Corporation. I think it was after that; that would've been the appropriate thing to do. And then Digital certainly understood him and knew about Bob, and Bob basically said "We could create a really good lab here in Palo Alto, and my hunch is I can get a lot of the people from PARC, because they're pretty unhappy."

Brock: Right.

Mitchell: And that worked, and it included me. Let's see. I actually went and <laughs> interviewed, which was kind of funny to interview with all these people I knew. Mostly we sat around and talked. Mostly we talked about "What are we going to do here that was different than what we did at PARC, and why should we be here?" Taylor kept saying "Great thing here is it's a computer company. They understand computers <laughs> literally." I have to back up just quickly.

Brock: Yeah.

Mitchell: In 1980, '81, one of the things Taylor set up was I was able to go on a sabbatical to Cambridge University for most of the academic year, and so I ended up having lots of acquaintances in Cambridge. And when the diaspora was going on and I said, okay, I'm going to-- they had given me an offer letter. I had actually said "Yes, I will come," and it was going to be on this Monday. All of a sudden, I think it was Monday or Tuesday of that week, this guy that I knew from Cambridge, Hermann Hauser, showed up. He had been in Japan. Another guy who is currently Secretary of the Royal Society, Andy Hopper, called him and said "Jim Mitchell's leaving PARC. We might be able to get him for Acorn." And all of a sudden Hermann changed his ticket, and he shows up in the Palo Alto--

Brock: Oh, wow, he was in Japan, and instead of going home, he goes--

Mitchell: Yeah, instead of going back to the UK-- his wife is from New Zealand, and they'd been there over Christmas-- he came to the Bay Area, and he's saying to me "I'd like you to start a research center for me, and we're doing this RISC chip called the Acorn RISC Machine." And I said "Really?" and I really grilled him about that. Jesus, seems like they really are, and I had hired-- it was '80, '81; this is now January of '84-- and I had hired in that interim from Cambridge at least three people, and we brought a couple more over who went to places like Hewlett Packard and so on. In fact, at least two of the final oral exams for PhD students from Cambridge were in my office at PARC. <laughs> Pretty funny. So he really pushed this, and he was willing to give me a lot of money. And we would create a research center in Palo Alto, and we would do the software, and the guys in Cambridge would do the hardware, and we'd make one hell of a great system.

Brock: Okay, so it was the software--

Mitchell: Right, he wanted the software. And I ended up saying yes and having to go in to Bob Taylor on a Friday and said "Bob, I know I signed the <laughs> letter saying I'm coming, and I was going to come next Monday, but I've got an opportunity I can't turn down." And we talked about it for a few minutes, and he said "Well, I'm sad about it, and if it changes, come talk to us. But I understand. Have a good time." And so off I went to Acorn. I was there about six months and had established this research center in Palo Alto, Palo Alto Square actually, and I forget what happened, but we got rid of a CEO, and the founders actually got a real CEO, who also happened to be Canadian but had worked for Massey Ferguson, the tractor company.

Brock: Hm.

Mitchell: He and I met a number of times, and he somehow decided I was a good guy, and at some point he said "I'm reorganizing the company, and I want one head of all of research and development. I don't want you there in California and another group here and nobody except me above you that understands what's going on."

Brock: <overlapping conversation> whatever, yeah.

Mitchell: Right. "So I want you to be head of R&D." And I had to think about it a bit. I took the job, because it meant every month I commuted to Cambridge. <laughs>

Brock: Whew. Wow, that's a long trek.

Mitchell: Ten to fourteen days. I was really good at handling jetlag. So ten to fourteen days, and I did that basically for four years during which, among other things, we developed the ARM chipset, and we did do a really nice, amazing operating system for ARM. Then there was a Christmas in 1983 when everything went to hell in a hand basket. That was the year that Commodore plowed all their Commodore PETs under--

Brock: Oh, yeah, yeah, yeah.

Mitchell: Right? It was the Christmas that didn't happen. That market saturated just like this: bam. <laughs> It just stopped, and the same thing happened in the UK, and so all of a sudden our revenues were going south and we were shopping ourselves around. Olivetti ended up buying Acorn.

Brock: Oh, I didn't realize that.

Mitchell: And owned it for a couple of years. It was actually '83 when this happened, and I guess Christmas of '82 was the one that didn't happen.

Brock: Okay.

Mitchell: No, that couldn't be, because I went there in '84.

Brock: So was it ‘84?

Mitchell: ‘85. It was the Christmas of ‘85.

Brock: Okay.

Mitchell: And we were actually getting the ARM working with a very small team, only 15 people to do a chip.

Brock: Wow.

Mitchell: With not very strong computing. We had one VAX 11/750, and when we wanted to do real simulations, we had to go to Munich where there was a VAX 11/780. <laughs> Honest, that’s how it got done. But it meant it had to be small and simple, a theme that keeps coming up with things like bitmaps and how you do fonts and stuff.

Brock: Yeah, yeah, yeah.

Mitchell: I could see the writing on the wall, because Olivetti did not know what to do with us, a research group. They were doing PCs at the time, right?

Brock: Okay.

Mitchell: And PCs both with DOS on them but other ones where they were running UNIX, because they were basically the guts of all the ATMs <laughs> in Europe.

Brock: Right.

Mitchell: That was sort of their business. Then they screwed up. With the transition to the 386, they missed the market and lost a whole lot of market share, and all of a sudden they don’t have money either, and I could see what was going to happen.

Brock: So, yeah, it was not good.

Mitchell: Ultimately-- So I ended up leaving. That’s when I went to Sun.

Brock: Okay.

Mitchell: But the group in Cambridge, when they finally were going to close the doors on Acorn, the group went to Olivetti and made a deal that I think involved one dollar or something to say “Can we have the intellectual property for the Acorn RISC Machine?”

Brock: Oh, my god.

Mitchell: And Olivetti couldn't think what they were going to do with this weird thing, and they said okay. And those guys said "We're going to go see if we can make a licensing business," and I remember saying "Boy, I hope you're successful. I wish you the best. That's a tough thing to go do." Boy, did they succeed.

Brock: Yeah.

Mitchell: I was so proud of those guys.

Brock: It's remarkable.

Mitchell: It's just amazing how well that has done.

<break>

Brock: The Java story.

Mitchell: Okay, the Java story. So I'm sort of-- I was--

Brock: Yeah. Well you get to Sun.

Mitchell: I was-- yeah. I was at Sun originally to do-- I was doing operating systems in that part of my career, and we were going to redo Unix in an object-oriented fashion. That was what-- it was part of the deal with AT&T to unify Berkeley Unix, and AT&T Unix that Bill Joy and others had set up. And Bill was going to be involved. Eric Schmidt was running the software engineering folks at that time, and I knew Eric well 'cause we'd hired him at PARC, and he'd been an intern there, right? In fact, he did the configuration language for Cedar/Mesa.

Brock: Oh, I didn't realize that.

Mitchell: And so Eric was running software engineering and I just called Eric when it was clear that Olivetti was not going to go anywhere and I-- and I'd actually talked to headhunters and they were trying to get me into a graphics group at Sun and talked to them. It was not a good fit. So I called Eric and I said, "So I hear Bill is doing this thing about Phase 3, redoing Unix in an object-oriented fashion." He said, "Yeah." I said, "Bill will do a really good job of starting it, but you and I both know he'll have a hard time finishing it." <laughs> And I said, "I'm pretty good at finishing." And Eric immediately said, "I want you over here. Interview." And I interviewed with him, which was easy, and Bill, which was also easy, and then with Wayne Rosing.

Brock: Whose name I have heard, yes.

Mitchell: Right. Wayne lives here. Wayne ended up being the first VP of Engineering for Google when they had only 15 employees. And he actually-- he takes credit-- Eric was running Novell at that time. I

don't know why he ever did that. But he visited Google one day, and when he was leaving Wayne went over to Larry and Sergey and said, "Get him. You need him. He'd be a great CEO. And he's not in a good place." And so the rest is history now, right? And Eric did a good job for them. Now I forgot where I was.

Brock: So you're coming in to work on the object-oriented Unix.

Mitchell: Right. Right. That ended up really not going anywhere. The other thing that's hard to do is-- there was an operating system group that really understood Unix, right. The Kernel Group, they're called. They owned Unix for Sun, and-- and we were going to do a new version of Unix that would be different in many ways: being object-oriented, probably written in a different language, etc., like C++ instead of C. The antibodies just came out. The Not-Invented-Here kind of syndrome. And that wasn't going anywhere. Meanwhile, in the early stages of that, there was a group of people that included James Gosling, Eddie Frank-- I've forgotten the other guy's name. And they were getting upset with what Sun had been doing in graphics and printing and everything. They had been working in that area. And Sun kept approaching it and then backing off and approaching it and backing off, and they were just fed up with that. And they're gonna leave. And literally-- I think it was Wayne and-- Wayne and Eric Schmidt were kind of-- Wayne ran hardware; Eric ran software. Together they were the VPs of Engineering. And they said, "We gotta keep-- these guys are too good to lose." They asked them what they wanted to do, and they wanted to go away and learn how to build what we would today think of as things like smart mobile devices and make them work in a connected world. With all the security, so you could have commerce, and all of that stuff, right. And so they went-- we actually got them a place off-campus. A building up on Sand Hill Road amongst all the VCs and everything. And they went to work to figure out how to do this. And they did this really smart thing, which is-- they knew on one side what kind of stuff they needed for graphics and the various parts of what they'd need for such a system, but they wanted a good language for doing it in, and James wanted to do the language. So he started doing what became Java. He called it "Oak" 'cause there was an oak tree out his window. And I went-- I was still in Sun Research, not running it; I was doing this operating system thing, but I went there because of my past experience with programming language design and I knew James, and I participated in the design meetings early on. Well, what they would do is they would say, "We need some sort of language primitive like this." James would come in in the evening, meet with them, they'd say, "Here's what-- we found these bugs in the language today" and they were just doing it as an interpreted thing, which is just a fast way to implement something like that. And we need something that does this," and they would talk about that and about, "Well, what language features should we have?" James would work all night and by the next morning, they'd come in, probably late in the morning and there'd be a new version of the Oak programming language interpreter there and they could start using these features and the little Read Me thing. And so they would then try to use it during the day and that--

Brock: That cycle, wow.

Mitchell: So the language co-evolved with the application. Which is I think-- and James knew all about Mesa. A few things from Mesa got in there. The exception handling. We had done one in Mesa that was too complicated and I argued for something simpler and we got that into-- into Java. So there are some

things like that. Well, while that was going on, we had-- there was an ill-fated distributed objects program at Sun.

Brock: Oh, I've heard of this.

Mitchell: Where they were trying to be standards in the world to do things that-- distributed meaning across the network, right? And there was a whole object-oriented-- there was a standards body trying to standardize things in this area. And, once again, we had to find an interface language, because that's what you need to describe the objects and so on. And a way of implementing that in an operating system way so that you didn't need to know whether the object you were talking to was local or remote. That was kind of the key experiment. Could we do this and then build a system where you really didn't need to know where things were? And we, at some point, got pushed out of the labs to try and get that into product, and-- as part of this distributed objects group. Because the Sun OS folk said-- effectively built antibodies against us. So off we went to do that. It wasn't going well. And at one-- but I had this great team I'd built and-- and one day Bill Joy came in and said, "You know, this is just not going-- this is great technology. It's not going anywhere. Over here we're doing this-- this thing." And by that time the language had gotten the name Java. First Person had changed into Second Person. That was the name of the group. And all of a sudden, with some marketing, if you will, by Mark Andreessen, who said, "Oh, this is really cool stuff. Now we can make web pages really active and everything." It had gotten the name Java and had just taken off. And Bill came and said, "We need your guys. We got a lot of work to do, and you've got this wonderful team that really knows how to work together and understands objects. Come on in to JavaSoft." And so we did. And I ended up-- so James and I kind of-- he was the Chief Technical Officer and I was named Vice President of Architecture. What we were really doing was splitting the job for being the Chief Technical Officer. And he was a rock star by then. He was getting called all the time out to talks and trips and things like that, so he needed someone who was paying attention, you know, down and in, as well. And so that was kind of my job.

Brooks: I get it.

Mitchell: And that's how I got involved with Java.

Brooks: Okay, wow.

Mitchell: Kibbitzing at the beginning and then seeing what a great thing it was and Bill Joy talking us into going over there and then I did that. I did that for, let's see, from '95 to about '98, and then they formed this little offshoot that was to do what we would call the Internet of Things today. The consumer electronics.

Brooks: Which was-- kind of goes back to the whole roots of the concept.

Mitchell: Right. Right. But still in JavaSoft. And I ended up being asked to go and be CTO over there. I did that for a little while. And then-- then I forget what happened but-- oh, Greg Papadopoulos, who was CTO for the company, came to me and said, "I'm running the Labs and this is just a lot of work. Would

you come and run the Labs?" And James was feeling kind of like he was not central anymore to all the Java stuff. I mean, he was definitely the rock star and stayed that way. But we had hundreds of people working on things by then and some really strong people, and he basically said, "I want to go think about something new." So I basically said, "Well, I'm going to go run the Labs. Come with me." <laughter> So he did. He came back into the Labs and did some stuff there for a while. When Oracle bought Sun he said, "I'm not gonna go work for Ellison." And he left and he went to a-- he went to Google for a short period, then he went to a start-up, Liquid Robotics, and now he's at Amazon. Still one of the-- maybe the best programmer I've ever known, and a very nice guy. I really love him. We used to joke 'cause he's James, I'm James; his wife's name is Judy and my wife's name is Judy. We're both Canadian. We both went to Carnegie-Mellon. It's like-- we both work in programming languages. Like twins separated at birth or something like that, you know.

Brock: And so you took over the helm of the Labs--

Mitchell: In late '99.

Brock: Was that--

Mitchell: Greg Papadopoulos.

Brock: Had Bert Sutherland been doing it?

Mitchell: Bert had left.

Brock: He had left.

Mitchell: He had left, yeah. He left PARC and-- the way the Labs actually started was: Bert had left PARC and he and his brother Ivan, who was really the inventor of both object-oriented programming and computer graph-- interactive computer graphics-- and Bob Sproull formed Sutherland, Sproull and Associates and-- and they had this consulting company on the side. This was after Ivan and Dave Evans at Utah had started Evans & Sutherland to do really big graphics systems that were used for the first electronic simulators for airplanes, right. And so they're off doing a consulting thing on the side. Sproull had been at-- a professor at Carnegie-Mellon and was kind of based in Pittsburgh. So we actually bought them a little-- as Ivan would point out, it was actually done as a merger of equals.

Brock: Oh, yeah. <laughs> He likes that line.

Mitchell: I know. He does. He does. He always says that he should have "employee number one." And he should be one, and Bert should be two, and Sproull should be three, and Scott McNealy should be four. Yeah. <laughter> Another good friend. Ivan's just great.

Brock: He is so funny.

Mitchell: Very-- yeah. <coughs> So what had happened was, Bert was running-- Wayne ran the Labs at first, but it was clear he needed to get out of that. Then Bert was running the Labs and-- and Bert's a really good lab manager. And he was managing when we then pulled the operating system out of the thing and went to the Distributed Objects Everywhere. "DOE" as it was called. And so we were out there when-- then went to JavaSoft and we were away. But when Papadopoulos took over as CTO, because we bought Thinking Machines-- we bought most of their people.

Brock: Oh, and that's where Greg had been?

Mitchell: That's where Greg had been. MIT, Thinking Machines. And then Sun. He and Bert just didn't get along, I guess. I don't know any details whatsoever but-- but at some point in there Bert decided to leave. That's all I know. And Papadopoulos was running-- he had-- we had-- why do I think we had three labs at the time? I thought we only had two. We had the one in California and one in Burlington, Massachusetts. And he had put a manager in each of them. And Bob was running the one in Massachusetts, in fact. So Papadopoulos asked me to come in and basically run the whole thing with Sproull reporting to me and another guy reporting to me. And I did that for about four years. And then decided, this management stuff, you know, I don't get to have any fun anymore. In fact, that was the-- first rounds of layoffs were going on at Sun. That was the least fun thing I've ever done.

Brock: Yeah, I can't imagine. I have not had to do anything like that.

Mitchell: Just awful. 'Cause you're not letting bad people go. You're letting very good people go. Right? And the rule is: you're not letting people go. You're deciding you're not going to do this project anymore, and so if the person's on that project their choices are: if you can find another project to join and they've got an open slot, then you can go there. If not, there's going to come a day when we're going to give you a package and it's terminated. So I went through all that. I did it all. Got past it. And then some people went back to now DARPA and they ended up getting funding for-- DARPA was running a supercomputer project to develop petascale supercomputers. And I think originally there were five companies. Phase Zero was write a proposal. They pay you to write a proposal. So there ended up being Sun, Silicon Graphics (SGI), HP, Cray, IBM. Those were the five. And the guys doing it on our side, the PI and everything, I would go to the meetings and sit in just to see what was going on, and I thought, "This is not going well. This is not." And I ended up talking to Ivan about it. I said, "You know, Ivan, we got this ARPA contract but we're not gonna even make Phase One with the way things are going. It's just not going right." And we had some technical ideas we really wanted to try on the hardware side, as well, and so-- I don't know whether Ivan talked me into this or how it happened, but basically Ivan and I started going to all the meetings and gradually tried to get it on track. And at some point I became the PI. I kicked the PI out who was there, who knew a lot about supercomputers but did not know how to get something going like this. They just wanted some short-term money. And the nicest compliment I got was from people I knew at SGI who said, "We were sure, after Phase Zero, Sun would just be gone. We weren't worried about you, at all. And then you and Ivan showed up." And all of a sudden there's a real contest going on and we ended up winning and HP and SGI were dropped, and it was us and-- for Phase Two it was Cray and IBM and we were the wild card. Those were the ones the government's been buying supercomputers from. And so off we went to do that, and at the end of Phase Two they had to downselect to two

companies from three and we lost. Best thing that ever happened. <laughter> I hated it because I hated losing, but literally within a month or two after that I said, “I think we dodged a bullet.” The main reason was the guy who was running DARPA at the time had a manufacturing background, and he didn’t just want you to do all the research and build, say, a small-scale system that demonstrated all the hardware and everything. We thought if we built a one-eighth scale system, an eighth of a petaflop--

Brock: Yeah, that’s pretty big issue.

Mitchell: That-- a, that was gonna be big. We could afford to do it with the funding. And we didn’t see any scale-up problems from there. That is, you’d already hit all the scaling problems.

Brock: You would replicate that eight times.

Mitchell: And he wouldn’t buy that. We had to build a full-scale model. And we were kind of willing to go off and do it, because they were going to give a fair amount of money, but then, as I say, we lost and so we didn’t do that. And that was one of the best things that ever happened. <laughter> I was in the supercomputer design business for about four years. One of my-- then what did I do after that?

Brock: That must have been around the time of the acquisition.

Mitchell: Not quite. It was getting close ’cause it was 2000 and-- it went from 2003 to 2007. 2007 to 2009, Greg had me getting involved with a number of things and then we were pushing-- in particular, there was one-- a couple of technologies that we had developed that we thought we should continue to push because they would be really useful in Sun’s products. And we were doing that and then a group inside of Sun found out about a DARPA call for a proposal for doing silicon photonics. That is doing--

Brock: Oh, right.

Mitchell: Light. Right? And there-- we happened to have a really, really good group in San Diego who knew about optics. They were good physicists but, in particular, they were really good optics folks. And so we ended up-- I managed that group. I was not the PI. It was someone else-- I didn’t know about optics. Not this-- well, I know about classical optics. Quantum optics and things--

Brock: Different game.

Mitchell: Different game, yeah. Different game. But it was a great group and-- and that research did very well. We set world records for speed, for lowest power per bit, all kinds of-- multiple wavelengths down the same fiber or wave guide. That was one of the reasons you could-- one of the dirty little secrets is that, in the server farms that are out there, the limiting factor, the bottleneck, is the processor to the memories.

Brock: Yeah, right.

Mitchell: Always has been. And-- and everybody uses what's called Serialization/Deserialization. It's called SerDes.

Brock: I don't know what it is.

Mitchell: Instead of having a very wide path between the pro-- using a lot of pins, what you end up doing is you just have a clock and two.

Brock: Okay, and you're flicking back and forth?

Mitchell: And you do this, and so you take the parallel stuff and you convert it to serial, but you run at real high speeds. Well, it turns out, at around 25 gigabits a second down these kind of wires, you start running into physics problems like skin effects, and so the amount of power you have to put in to keep-- to go faster is going up and up and up and-- and like 30 percent of the power of the chip is going into running the SerDes.

Brock: Okay, wow.

Mitchell: This is a problem, right? And so the idea was: could we use optics in a-- basically in place of that. And that has happened. The first use of optics was between chassis, where-- literally in some supercomputer centers the cables between chassis weighed so much it was breaking the floor. The copper. So they came out with these things called active cables. So they look just like the copper cables in terms of the-- the connectors at the end, but inside the connector was something that did electric to an optical, then it was a fiber, which doesn't weigh much, to the other end, and then back to electrical, so you could just do that and replace--

Brock: Right. Plug in-- plug and go replacement.

Mitchell: Interfaces and implementations, right? So that was the first thing that happened. That solved the particular weighty problem. The second thing that happened was people started doing it up the backplane between the servers in a rack. And then the next was to start going from the backplane onto what's called mid-board, getting closer to the processor, if you will. And I think ultimately the processor will do it and come straight off in optical, because-- here's the difference. As I said, we were running out of oomph at around 25 gigabits a second with the electrical stuff. Our first attempt with optical, we were running at 40 gigabits a second at eight colors.

Brock: Wow, so eight times--

Mitchell: Yeah. Eight times forty.

Brock: Geez, that's incredible.

Mitchell: Exactly. That's a huge difference, right? And-- and whereas-- and the energy per bit was literally a picojoule. A trillionth of a joule.

Brock: You were getting down to the limit of--

Mitchell: I think we could get down to about a half a picojoule. But-- so the power went down. The speed went way up. You don't have to worry about noise. One of the great things about light, I like to say to people, is, unlike when you're doing the SerDes stuff electrically, the problem you get is this Gaussian spread, right? You start off with a pulse that looks like this. By the time it's gone all the way it's getting spread out like this. Light doesn't do that.

Brock: True.

Mitchell: Go right across the room or go one inch it's sort of the same and it's about the same amount of power. So it was a winner all around. And so by 2013 that research had done so well that I said it's time for products. And the way to do products is you gotta move-- you gotta have the team go. You can't throw it over the wall.

Brock: Yeah, the team has to do it, right. Yeah. Move with that.

Mitchell: And in this case, unlike the, you know, the Not-Invented-Here thing with the operating system-- Larry Ellison was behind this. And he allocated money and-- and a place for the team to go in product land, and-- and said, "Three years. Go and make a product based on this." And so the whole team moved over there. And this was late 2013, at which point I looked around and said, "I have a secretary and nothing else to do." <laughter> Good time to retire.

Brock: Yeah, right.

Mitchell: It was perfect, right?

Brock: So they must be closing in on commercializing.

Mitchell: They did a first product and then Oracle got out of the SPARC business. And without the SPARC business you really don't need this. And so that team is off and they've done a startup.

Brock: Oh, to take it out there. Who knows?

Mitchell: Oracle's not giving them that IP, so I don't know what they're doing exactly. I know a little bit from about-- I keep in touch with them about every six months, but-- when they do a friends and family round, maybe I'll get in. I'll get involved.

Brock: It's such a fascinating range of things that you've had an opportunity to work with. It's quite, quite incredible.

Mitchell: As I said, I get bored easily, and so I kept switching careers [because], you know-- operating systems and this-- processors and stuff like that. And this isn't about computing, but in retirement I was really worried I'd get bored. If I'm not around smart people and I'm not learning new things I get bored, and I thought, "This is not going to be good." One of the reasons we came here is because having this university here-- I really--

Brock: Yeah, smart people around.

Mitchell: You know, my undergraduate degree's in math and physics, and the Kavli Institute for Theoretical Physics is a great place to go and-- for lectures and stuff. But a friend of mine got me involved in, so now I am on the Board of Directors for a foundation with an endowment that gives money away for basic research in biology. For life sciences. So I'm learning biology as fast as I can. And running-- and I run the Science Advisory Board for that, to review, you know, applica-- grant applications and stuff.

Brock: There's so much going on in there and there's so much of computing in it, too, now, that, I mean, you've got to-- you might have a leg up by understanding that dimension.

Mitchell: Yup. It's gotten very interdisciplinary. Machine learning and AI are involved in a whole lot of the-- of the stuff, yeah. If I were a young guy, I think that's the field I'd probably go into.

Brock: It seems-- it seems-- there's going to be a lot of problems to tackle in that area, technically--

Mitchell: And ethically.

Brock: Yeah, that's exactly what I was thinking.

Mitchell: This idiot in China who actually took two girls to term.

Brock: That, I think-- I was just talking with somebody about that. I cannot wait for the book to come out about that guy and what that whole story is, because interest-- you know, all of the people involved have just kind of disappeared off the map after he--

Mitchell: He was put under house arrest.

Brock: Right?

Mitchell: Yeah.

Brock: I mean, where are the-- where are the families? Where are the children? You know. And having that-- the full accounting of that--

Mitchell: I hope they are protected. I hope someone's following them in China, because the concern is that with CRISPR you can get off-target editing, as well. And if there was off-target editing, who knows? When they're 20, they may find suddenly they're getting some very bad disease, right?

Brock: Absolutely.

Mitchell: That's why he wasn't supposed to do it.

Brock: Absolutely. Yeah. Well, fascinating. Well, let me just switch this off.

Mitchell: Okay. And I get to have smart people with me there. Our first hire on the Science Advisory Board is Jennifer Doudna, the co-inventor of CRISPR.

Brock: She's so impressive.

END OF THE INTERVIEW

