Oral History of Bill Joy

Interviewed by:
Alex Bochannek

Recorded: March 30, 2011
Mountain View, California

CHM Reference number: X6073.2011

Alex Bochannek: Welcome to the Computer History Museum. Thank you for doing this interview with us.

Bill Joy: It's a pleasure to be back.

Bochannek: Why don't we start with your name, where you were born, and your family? Tell us a little bit about your family, siblings; the area where you grew up.

Joy: Sure. I'm Bill Joy. I grew up in the suburbs of Detroit, Michigan. I'm the oldest of three children. My father and mother were both schoolteachers. I went to school at the University of Michigan and at Berkeley before starting Sun Microsystems in 1982.

Bochannek: What were you like as a child? Were you very studious? Were you social?

Joy: Well, I didn't study very much, but I got most studious in high school. I think people perceived I studied. I come from a middle class family. My father and mother were both hard workers as were people in the Swedish side of my mother's side of the family. I worked through high school and through college mostly in the restaurants as a busboy, cook, and dishwasher to make money to put myself through school.

Bochannek: Related to that, if I may ask, what were those values that you were raised with? Politics, religion, ethics; did those things play a big role in your upbringing?

Joy: I think the importance of working hard, trying to do your best [did]. My father was a strong believer in relative grading. You get an A if you do your best, not if you do better than the other people. My father, my uncle, my brother, my sister, and all of us in our family are very much project kind of people. We like to do things that take an extended period of time, have an arc. If they have some sort of ethic nature that's even better. My brother likes to say he likes to work as a hobby. It's more creative than passive. I think that's something that I, as well as my siblings, got from my family.

Bochannek: What do your siblings do?

Joy: My brother's run a number of different businesses. He's run restaurants, he's run manufacturing shops, he's built a lot of houses, run ranches. My sister has worked in legal services. She works for a series of companies that have used computers to automate handling of large corpus of legal documents including LexisNexis and other related companies.

Bochannek: When you were in school—elementary school, middle school, high school—which topics did you particularly enjoy? Which ones didn't you enjoy?

Joy: I wanted to be a math major. I enjoyed math, and I enrolled in the engineering school partially because you could take more math classes as an engineer than as a math major. I found computer programming, when I was college, to be even more interesting. I'd say more seductive and addictive than math classes were.

Bochannek: You won a prize, The Michigan Math Prize. Tell us a little bit about that.

Joy: In high school I went to probably what at the time was a relatively ordinary suburban high school. We had really good teachers and everything, but we didn't have the advanced math classes that you'd find in a private school, which I couldn't have afforded and my family couldn't have afforded to send me to. There was somebody affiliated with the school who, afterhours, would come and take a small group of the best math students and coach us. Then we entered this contest. I won the contest; the test that they gave everyone who was participating in this contest.

Bochannek: That's great.

<*crew talk*>

Bochannek: We're talking about your high school and computers. What was your first exposure to computers?

Joy: My first exposure to computer programming was [when] one of the more hippy-oriented students at my high school—a kid with really long hair and a beard—was sitting on the floor of the high school scribbling notes on a line pair listing of a FORTRAN [IBM's Mathematical Formula Translating System] program. I asked him what it was and he said, "FORTRAN." I had no idea what it was. It wasn't until I went to college that I had any further exposure, because we didn't have computers at our high school.

Bochannek: You went to the University of Michigan. Why did you choose University of Michigan?

Joy: I applied to Stanford and MIT [Massachusetts Institute of Technology] and Carnegie and Michigan, and I got into Michigan and Carnegie both with some financial aid. MIT and Stanford didn't work financially for me. I didn't get into MIT. It wouldn't have worked financially but I just tried anyways. Michigan was an in-state school and quite affordable. I could keep my weekend job to help pay my way through school.

Bochannek: What was that weekend job?

Joy: I was short-order cook on the midnight shift at a restaurant where I'd worked for many years. I would drive back from Ann Arbor and stay at my family's house in the suburbs there and work midnight to eight on Saturday. The day cooks weren't the most reliable, so I'd generally get ten or twelve hours before they would show up to work at three bucks an hour. I'd make $75 or $80 on the weekend which really helped pay the way through college.

Bochannek: What was the college experience like at the school? Michigan is a very large school.

Joy: It's a very large school, with an engineering department that was really ascendant. I mean, it's gotten much better since then. They had really good computing facilities. They had a computer operating system which they had written themselves called MTS [Michigan Terminal System]. It had been a hotbed of innovative computing since the 60s both in programming language and operating systems. Taking IBM [International Business Machines, Inc.] hardware and writing new software to run on the machines. There was a professor there who was very into large-scale numerical computing. I had an exposure to both programming languages, numerical computing and operating systems, which were very contemporary topics at the time. Including a rich set of programming languages. They had PL1 [Programming Language One], FORTRAN, Assembler, Snowball, MAD, the Mission Algorithmic Decoder, which was an extensible programming language. There was a really rich tradition of different things. That was really a great place to have an undergraduate education in computer science.

Bochannek: What classes or teachers did you feel had the most impact on you while you were there?

Joy: The professor I worked with, Don Callahan. and the work we did on the early super computers writing sparse matrix codes; that was really, really influential. The job that I had at the computer center, as a consultant, gave me access to large amounts of computer time, because I had an account for that purpose. Normally in a class, you'd get $30 of computer time. They would charge you for everything. A dollar-an-hour to sit at a terminal. So much for virtual memory, so much for real memory. That helped enormously. It was more than the classes. The classes were good, but the things that helped the most were these essentially extracurricular/summer activities and the jobs that I had to pay my way through school. That was unique, and the computing facilities were very advanced at the time. There was even a computer network between Michigan and Wayne State and Michigan State called the Merit Network. This was a long time ago. The ARPANET [Advanced Research Projects Agency Network] didn't exist. There were a few leading universities connected in that way, but it was a very advanced computer network for the time.

Bochannek: You said you did consulting work in the computer center. What was the work you did?

Joy: You'd sit in a cubicle, and people would come and ask you questions [about] why their programs didn't work. Generally, they'd type down punch cards and run through the card reader submitted batch cards to introductory programming assignments. You had to consult on all the different languages that people were writing their programs in.

Bochannek: When did you graduate and what did you intend to do with your degree?

Joy: I graduated in 1975. I applied to grad school. I wanted to go to grad school in computer science. I was actually an electrical engineering major at Michigan, because I don't think they had a computer science department per se at the time. I may be wrong about that. To my recollection they did not. I applied to Carnegie, Berkeley, Stanford, MIT, Caltech. The two best offers were probably to go to Caltech as a research associate, or to go to Berkeley as a teaching associate, just with support. Stanford, MIT didn't really offer support. I would have had huge college loans. It didn't seem that attractive. I decided to go to Berkeley, because of, I think, the reputation of the school with the Nobel laureates.

Bochannek: How did your parents feel about that move?

Joy: They were quite happy with it. I was ready to leave Michigan. I needed to see the world a little more. I had no intention to stay in Michigan for graduate school. Didn't apply to do that.

Bochannek: Berkeley was synonymous with major social changes in the 1960s. How did you experience those as a teenagers in the '60s, and then as you arrived in Berkeley in the mid-'70s? Did those affect you?

Joy: No, I wasn't really that involved in any of that because I was going to high school and working. I wasn't aware even going to Berkeley of the free speech movement and all that. I only learned about that later. I was pretty academically inclined, and not really socially active at all.

Bochannek: Can you compare the cultures, the campus life and off-campus life at University of Michigan and Berkeley?

Joy: Well, it's a lot warmer in Berkeley. No, I lived in a co-op in Michigan, and that was nice. A group of maybe 16 people in a big house with a bunch of rooms. When I moved to Berkeley, I had a girlfriend and we got married shortly after arriving in Berkeley. We were living in an apartment. That's obviously a big change. But Berkeley was very intense. I got involved very quickly in the work that was being done with UNIX, and Ken Thompson came to Berkeley. It became pretty much all-consuming very quickly, because that was an enormous project that we were undertaking to improve the system there. It was much more intense than Michigan. The graduate courses were harder, but the computing facilities were more immediate. At Michigan, there was this large central computer that only a few people could really make changes to. At Berkeley, we had a minicomputer that we were basically doing all the work on, all the operating system. The departmental computing facility was ours to care-take, to "husband," if you will. That was quite a lot of work.

Bochannek: You mentioned your girlfriend, later your wife. Did you meet at Berkeley?

Joy: No, she was in the co-op at Michigan and moved out with me.

Bochannek: What is her name?

Joy: Colleen.

Bochannek: Colleen, okay. Now what was your position at Berkeley? You said you had an assistant teaching position?

Joy: Teaching assistant for undergraduate classes, yes.

Bochannek: The work you did at Berkeley pertains to UNIX. Can you explain in your own words what UNIX is and what its relevance was at its time?

Joy: In that time, a lot of computing research was being funded by DARPA [Defense Advanced Research Projects Agency] and the leading universities that they funded were MIT, Stanford and Carnegie. All of those schools had large minicomputers. PDP [Programmed Data Processor]10s that they bought. Even though they all had the same computer, they ran, in fact, separate operating systems. Each of the universities had written their own operating system. Michigan had a large system also. Berkeley had had a large system, but when Reagan came in as Governor, it was an old Seymour Cray CDC [Control Data Corporation] machine, and they basically cut the funding. There was a budget crisis in the state. Berkeley had no large computing system. When I arrived it had only a 16-bit computer, which is a relatively small computer, a PDP-11, as opposed to the PDP-10s which were much larger computers that the other universities had. This little PDP-11 ran UNIX. Ken Thompson was the author of UNIX, but Dennis Ritchie was an alumnus of Berkeley. The Departmental Computing resource essentially was this one PDP-1170, and also it had a fraction of another PDP-11 shared with the Math and Statistics department, which ran three different operating systems depending on the day of the week, and the time of the day.

Bochannek: One of the people who was key at Berkeley in bringing UNIX there was Bob Fabry. What was his interest in UNIX? I understand Ken Thompson was at Berkeley when you arrived there as well.

Joy: Well, what happened was Dick Fateman actually, who was a professor there, had come from MIT and brought a symbolic algebra system called Maxima. Maxima was too big. It was a list program, and it was way too big to run on the PDP-11s. Since Berkeley didn't have a large computer, the department cobbled together the resources to go get a VAX [Virtual Address Extension], which was a new computer architecture. In fact, Carnegie, Stanford and MIT weren't that interested in it, because they already had PDP-10s which they'd done a lot of work on. The VAX was very incompatible. It was more like a PDP-11. We got this machine, and we arranged to get a version of UNIX to run on it, which had been done by some people in Homedale at Bell Labs. The original UNIX had been done at a different Bell Labs facility in Murray Hill. They had done that version of UNIX. Now we had a VAX running UNIX, which was relatively unique. The large research universities weren't running UNIX. They were running PDP-10s and they were running incompatible operating systems. The nice thing about the VAX was [that] it was cheap enough compared to a PDP-10, that smaller universities could afford to buy it. That was essentially a market opportunity within education and research. Now we had an opportunity to take this version of UNIX that Bell Labs had done and improve it and make it more suitable for universities. For example, by putting a Pascal compiler on it; that was a language that people taught computer programming in. In order to run Maxima, or "VAXima," as we called it, we used virtual memory. I got involved with another student over a summer, and we put virtual memory in the UNIX, so we could run larger programs.

Bochannek: What year was that?

Joy: We probably got the VAX in like 1977 or '78. Sometime around then we put virtual memory in the UNIX. I don't remember the years exactly.

Bochannek: Now you worked on the Berkeley software distribution at Cal, and that's related to UNIX. Can you explain what that was?

Joy: Right. Chuck Haley and I had spent a summer, probably in '76 or '77, I don't remember exactly—the department had found a little money to have us try to finish a Pascal interpreter, which Ken Thompson had written. Ken had come back from a sabbatical, and brought with him a hobby project, which was to write a very simple Pascal interpreter. Ken hadn't really finished it to the level that it was that easy for students to use, because the error messages that it gave you when you had a syntax error were just too hard to understand. The runtime errors just weren't clear enough. Chuck and I, over a summer, fixed it up. We also took the UNIX editor, which is a little bit obscure, ED, and another editor called EM, which we'd gotten from England and fixed it up to be EX. We had an editor that had error messages which was easier for the students to use. Then, I decided that this could be useful to other people who had UNIX running on other PDP-11s. I got some nine-track tapes and offered to send people copies of the software for $50. I must have sent out hundreds of those. It created quite a little slush fund, because the tapes cost $10. We had $40, and it's not taxable.We probably had $10,000 now to spend on other stuff: travel, phone bills, all sorts of things. When the VAXs and the 32V, which was the operating system we got from Bell Labs, came along they didn't have virtual memory. We started fixing it up, put virtual memory in, and then ultimately internet code into it. Then [we] took the tapes, which had just been application software for the PDP-11, and extended them to be full bootable VAX operating systems. Now you could get a UNIX operating system with virtual memory and ultimately the internet and all this software from Berkeley, rather than getting it from Bell Labs, or from Digital Equipment. Pretty soon, over a period of years, we're maybe pushing a thousand, maybe even more than that. There were a lot of machines around the world running that operating system, which became the ultimate. It was called 4BSD [Berkeley Software Distribution], which was the fourth version of this. 1BSD and 2BSD were PDP-11 tapes. 3 and 4 then ultimately were VAX tapes containing a lot of internet protocol support. It very quickly became very important to the internet, because TCP/IP [Transmission Control Protocol/ Internet Protocol] were DARPA things and not really supported by things like VMS [Virtual Memory System] and other operating systems.

Bochannek: Who worked on BSD? You said you worked originally on it with Chuck Haley?

Joy: The two principle people who worked on it the most were Kirk McKusick and me. There were many others. Keith Sklower, Mark Horton—many other people worked a lot—Peter Kessler. Other people worked on the various aspects of it. The primary contributors were Kirk and I.

Bochannek: What was the atmosphere like in creating this?

Joy: Well, we had a corner office with a nice view of the campus, and you'd find us there hacking most hours of the day. We'd take the machine down at lunchtime, and install new upgrades, and then reboot it. If you'd come in the morning and complain, but maybe in the afternoon we'd have fixed whatever was bugging you. Eventually after a couple of years, we got a little more space. That was on the fifth floor, and we got a little more space on the fourth floor of Evans and a half-a-dozen VAX11 750s, which were smaller machines. Then the larger timesharing 780 VAX, it was called, that we had upstairs, and that allowed us to do more experimentation. That was part of a DARPA grant which Bob Fabry had gotten for us to help them with a problem. As I mentioned before, the PDP-10s all ran different operating systems, so if DARPA would fund some research, say, at Stanford, and there was some other research at Carnegie, they couldn't share software, because they were written in different programming languages for different operating systems. DARPA wanted to have a common operating system it could use across a lot of different programs at different universities. They wanted it, in fact, to be hardware independent, too. They could imagine carrying the software forward in the future. We had a group that Bob got us a grant for, and I got then a job working for the university as a principle programmer to develop the operating systems, develop UNIX more to meet the technical requirements that DARPA had for its programs.

Bochannek: UNIX came out of Bell Labs. Later AT&T commercialized UNIX themselves. Sun had a very strong relationship with AT&T. There are a number of legal issues around UNIX and AT&T for many years. How would you characterize the attitudes of the UNIX community towards AT&T?

Joy: We essentially had a university non-profit only commercial version of UNIX. It was non-commercial in the sense that we only charged a distribution fee for materials. It couldn't be used for commercial purposes, but our goal was university research. That was piggy-backing on a license that AT&T had. When AT&T went into the commercial UNIX business with System 5, I think the technical community would have said that their software was inferior to the Berkeley system, because ours was much more robust, and had virtual memory. It had really been running on thousands of machines for a very long time. That made an antagonistic relationship initially with us with that commercial group, because we, in some instances, were cutting into their market. Also, I think, the commercial group at AT&T was different than the research group. Our operating system was really a spinoff of the research version of UNIX. The commercial system was from a different part of AT&T. Whatever tension there had been between those two groups in setting the future version of UNIX internally, we carried over to our relationship with the commercial side of AT&T.

Bochannek: You mentioned the PDP-11, you mentioned the VAX. The BSD work was done on machines by DEC [Digital Equipment Corporation]. You mentioned the larger PDP-10 machines used in many universities. Even though BSD was quite popular on the VAX machines, DEC's management had mixed feelings about UNIX.

Joy: Right.

Bochannek: Construct the situation and also maybe expand a little bit going forward and what some people refer to as the UNIX wars of the '80s, what those were.

Joy: I was a bit peripheral to those. Oh, I know what you mean. No, that I wasn't peripheral to. What had happened with DEC first of all is that the PDP-11 had a ridiculous number of operating systems. I think it had 25 different operating systems. This was a problem for Digital in terms of support. In this day and age, you see Apple having a single OS for the iPhone, the iPad and the Mac, really. Underneath it's a derivative of BSD, in a way. That's an enormous simplification for them. DEC was fighting; if Microsoft has three operating systems for three devices, DEC had 25. It's just ludicrous. They made a decision with the VAX to have only VMS. Then UNIX comes along and having just killed 24 systems to get to one, they were in no mood to have two, because that would be the slippery slope. They were not receptive to UNIX. There's only one operating system. We had microfiche because there really wasn't good documentation of how the hardware worked, because they didn't have to document it for lots of people to understand the low level workings of the machine. We ordered the microfiche for the assembly language source code for the assembly language parts of VMS, which is most of it. We would have to read the microfiche to figure out how the hardware worked. We actually sold a lot of VAX hardware for DEC. We also wrote people a memo called, “How to Buy a VAX.” Bob Kridle and I wrote that memo. It said, “Buy the cheapest configuration of the VAX. Sell the peripherals you’re forced to buy on the used market. Buy these third-party peripherals, and you’ve got a machine for half the price of the higher end of the VAX line.” I don’t think that made us very popular with Digital. Aside from the fact that we didn’t sell these overpriced peripherals, we sold a lot of the guts of their computers, and got them into a lot of accounts probably they wouldn’t have gotten into.

Bochannek: DEC eventually did offer UNIX on some of their hardware. Throughout the ’80s, [they] found itself on the opposite side of these so-called UNIX wars to Sun.

Joy: I’m not familiar with what happened with Digital, but I know that what really happened is Sun and AT&T decided to try to bridge the gap between the UNIX that came out of Berkeley. That was System 5, which was the version that had [been] done at AT&T. We announced an agreement to do that, probably a little bit handily locking out—or at least they perceived they were locked out—people like DEC. Some of the vendors who perceived themselves—DEC and HP [Hewlett Packard] and others—having been locked out created an organization called OSF, which they called the Open Software Foundation, to come up with a version of UNIX that they perceived wasn’t controlled by AT&T and Sun. Of course, not only UNIX had always been controlled by AT&T. This was a little silly. I think the technical people at OSF weren’t very well-grounded. They said they were going to do a lot of things, which were, in fact, very difficult to do, and I don’t think they achieved. It made a very large disruption in the UNIX market, which certainly hurt UNIX in the ’80s in competing with Microsoft’s operating systems on the servers. It probably hurt UNIX also in terms of just a single standard emerging on the desktops. UNIX has always been a little bit fragmented in that way. That’s been a strength, but also its weakness.

Bochannek: Related to this, the intellectual property situation with UNIX has always been quite complicated. You hold a number of patents, including patents for software. What’s your position on intellectual property law, patent law, with regard to software? Has your position changed over the years?

Joy: I never filed any patent things on UNIX-related stuff. When we did the Java language in the mid-’90s, we filed some patents on certain aspects of it. Much like Digital was chastened with 25 operating systems, we were concerned about Java splintering. We tried to have a programming language where the program would give the same answer, no matter what hardware it was running on. I think, in our minds, more defensive against people trying to splinter our platform. In fact, the platform, the Java platform was available quite inexpensively. I personally don’t have a strong opinion about software patents. I think patents, in general, are a good thing. Sometimes the patents are so trivial—they patent the obvious and the common sense, and gets upheld, and it just impedes progress. I think you need patents, and maybe there’s always going to be stuff in the gray zone that people get unduly enriched for patenting the obvious. If that’s the price we have to pay to encourage invention, then maybe it’s the best system, if not a perfect system.

Bochannek: Now returning to BSD, a user community developed around the Berkeley software distribution. How many sites used BSD?

Joy: Probably a thousand. I don’t know.

Bochannek: Up to a thousand.

Joy: In my time. There may have been many, many more obviously later.

Bochannek: What did they do with this? How did they communicate amongst each other? Was there a community built up around it? What was your role in that community? How were you perceived by that community?

Joy: At the time, Bell Labs had some sort of simple stored forward networking. Then, in fact, Eric Schmidt, who later became CEO [Chief Executive Officer] of Google, had written a networking layer for UNIX that did stored forward messaging at over modems. I think we called it Berknet. It was all dial-up modem stored forward. It wasn't a packet switching network. The network news, you know, the kind of the [collection] of blogs all went over those connections. There emerged a community like that. At Berkeley, we had a list of things we wanted to do to the operating system, and people would send us reports and we'd look at them, but we were pretty busy forward facing. I think obviously, the community out there developed a lot of things locally. They didn't really get integrated in the way things get integrated now because the bandwidth to connect people was relatively limited, and there was not real good way of distributing even megabytes of code. The stuff that went on the Berkeley tape, other than the stuff that was done at Berkeley, was relatively limited.

Bochannek: To get back to the other question, how do you think you were perceived by that community? What was your role? Your identity in that community of BSD users?

Joy: In those days when I was involved, we didn't really have a user meeting, or anything. My personal experience was pretty much one-way. We'd send tapes out, people said they liked it. But it [was] mostly a one-way interaction.

Bochannek: You mentioned earlier DARPA became involved in funding the BSD work. TCP/IP was implemented in BSD. I believe that was the time the Computer Systems Research Group, CSRG was established?

Joy: That's what we called ourselves after a while.

Bochannek: What was the impact with BSD with TCP/IP with the larger community?

Joy: I think it was synergistic. We took some code that the guys at BBN [Bolt, Beranek and Newman Technologies] had written, which was tuned to run over like a 65-kilobit line. We had Dave Boggs of Xerox had built with/for us some Unibus, which was the plug-in slot for the VAX that it inherited from the PDP-11. These cards. We could run a three megabit Ethernet, an early version of Ethernet. When you put the BBN code on a 3-megabit Ethernet, it still ran it like 65-kilobits. It wasn't tuned. They tuned it enough to work on a slow line, but if the line suddenly is 50 times faster, it saturated the CPU [central processing unit]. We ended up having to rewrite the code in order to straight-line it to so it would run at wire speed. People at the time felt that TCP/IP was defective and would never run fast. In fact, it wasn't the protocol itself that was defective. You could imagine ways to improve the protocol; it wasn't perfect. The main reason it ran slow was the implementations weren't very good. It hadn't been debugged for high speed. Once we had this high-speed version, it became interesting, not only as a wide area network protocol, for the wide area networks of the time, which were like 56-kilobits that you could afford. Also for the local area net internal environment, where you had a ring net, or you had a low-speed Ethernet. When 10-megabit Ethernet came along and people started to get more and more into the internet, and we went through the NCP [Network Control Program] to TCP [Transmission Control Protocol] transition from the old circuit switch network to the new packet switch network for the internet, Berkeley UNIX was really ready for that, and became really one of the principle operating systems on that packet, new packet switched network with a very high performance protocol stack.

Bochannek: The TCP/IP work in BSD happened in the 1981 timeframe, 4.1A TCP/IP.

Joy: Right.

Bochannek: I believe that was the version of BSD.

Joy: Right. We were marching along 4.1, 4.2. We were too ambitious, so we stuck 4.1A, B and C in-between 4.1 and 4.2.

Bochannek: Now this is the timeframe when you met your fellow Sun founders.

Joy: Right.

Bochannek: How did that meeting come to be?

Joy: I'd met Andy Bechtolsheim at some point, and I knew Vaughan Pratt. Vaughan had been a professor at MIT [Massachusetts Institute of Technology], and he was an advisor to Andy at Stanford. Andy the co-founder of Sun. When I had designed the vi, the screen editing, the WYSIWYG ["what you see is what you get"] editing portion of that editor for UNIX, I had adapted some of the ideas that Vaughan had put in an EMAX-based editor that he'd done at MIT called DOC. I knew of Vaughan and had met Vaughan. Scott McNealy and Vinod Khosla and Andy, in starting Sun, realized they didn't have a software person. Andy was a hardware person. Scott was a manufacturing person, and Vinod was the CEO to start the company. They were missing software. At one point they were trying to find someone to be their software founder, and Andy and Vaughan discovered that they knew me and decided to come to see me. They came up and arrived at my office, and they looked very young, as I'm sure I did at the time. People had come to see me about using UNIX on various startup companies and whatever, but I figured these were like the junior guys, and that the senior guys would show up later. I had them wait in the corner of my office until they all arrived. Then I realized that this was—they were all 20-somethings like me.

Bochannek: What was your relationship with the other three guys over the years? You said Andy Bechtolsheim is somebody you had known before. You got to know Scott and Vinod, and of course, Scott McNealy stayed at Sun about as long as you did.

Joy: Longer.

Bochannek: Longer, okay.

Joy: He was the last of us to leave. First of all, I think when they came to Berkeley, they were very impressed, because I walked them into my room where I had six minicomputers. Andy seemed quite interested. I simply walked up to a machine, turned it off and started pulling the boards out. Andy was very impressed that I wasn't afraid to take the machine apart. Of course, he put things together. In the early days, we had clearly separate responsibilities. At some point Scott and Vinod had similar ambitions. Vinod left the company not very long after it was founded and went to Kleiner, Perkins because I don't think the company was—there wasn't enough room for one to be the CEO, and the other to be the President. That wasn't going to work. Through whatever process I wasn't fully privy to, they decided that Scott would stay and Vinod would go. They were very, very close. They were sharing an apartment at the time. Once it got down to them, Andy and Scott and I as the three founders, we worked together for many, many years until Andy finally decided he was interested in computer networking hardware, and left to start a computer networking hardware company.

Bochannek: How did you decide to join those three guys to found Sun?

Joy: It was a funny reason really. The things we were doing at Berkeley were very successful. We had plenty of pocket money. We were sending out hundreds of tapes a year for $300 each, so we had tens of thousands of dollars in an account we could spend on things. I was making a decent salary and trying to finish my thesis. We had budget to hire people, but Berkeley was so constrained, I couldn't hire anybody because I couldn't give them an office, because the campus is so locked in the city. It became untenable at some point. More and more people were using the system. More and more things wanting to be done, and no ability to grow the group of people to address the—it clearly needed to—it didn't fit in the university anymore. It needed to be a commercial activity. When they came along, it just seemed like, "This isn't kind of working where I am now, so why not try something else?"

Bochannek: You made the transition from Michigan to Berkeley. Your parents were very supportive of that. How did they feel about going to this new startup? Did they feel it was a risk you were taking?

Joy: No, I don't think they felt it was a risk, because I don't think they knew what a startup was. They were fine. I had one job. I actually was paid less at Sun than I was at Berkeley. They probably wouldn't have thought much of that. We only paid ourselves like $2,000 a month when we started Sun, which was $24,000 a year, which is pretty low for a VP [Vice President] of a company. That's what we could afford at the time.

Bochannek: You were married at the time, too.

Joy: I was, yes.

Bochannek: What about the other guys? What was their personal life like?

Joy: Vinod was also married. Andy and Scott were both single. I got divorced shortly after we started Sun. I think Colleen, my wife at the time, preferred a different lifestyle than the lifestyle of the Valley. I think it worked out better for her.

Bochannek: There's an article from Bob Havre in the June '82 issue of *Login*. The *Login* magazine where he said, "Interesting Developments. Bill Joy of UCB moving to Sun Microsystems. Bill Joy has decided to become involved with a new startup company, and will be phasing out of the Computer Research Group at Berkeley over the next few months. He will be joining Sun Microsystems, Inc. [SMI], a company whose founders include Andy Bechtolsheim, the designer of the Sun workstation. While SMI may need to develop proprietary software to service specialized areas, Bill expects fixes to the shared base of 4.2BSD programs, which are made at SMI, can be distributed by Berkeley. The current corporation efforts between CSRG and various industrial groups are seen as a model for the relationship."

Joy: Right.

Bochannek: Did that transition go as Bob Fabry said it was going to do?

Joy: I actually don't know. One of the things I did when I went to Sun is rather than being involved in UNIX, I decided to get involved in the hardware side more as an architect. Dave Patterson at Berkeley had the risk idea, and very shortly after I went to Sun I became a chip architect. At Berkeley it really had been mostly Kirk and me. We needed a much larger group. I felt like I didn't want to tower over the UNIX group at Sun. I wanted to hire a more balanced group of people. I decided to not to continue to do so much with UNIX. The relationship between the UNIX group at Sun and the Berkeley group was really outside my purview.

Bochannek: Do you actually know if things got fed back into CSRG?

Joy: I suspect they did, but I don't know the exact working relationship.

Bochannek: What was your relationship with the people that were in CSRG, like Sam Leffler, Mike Karels, or Kirk McKusick?

Joy: I think after I left, they assumed a larger share of responsibility, and they continued to do a lot of really good work on UNIX for many years. In fact, starting at Sun—Sun started in February of '82, and I think between February and July, I worked both at CSRG and at Sun part-time on each side. I tried to phase it out in a responsible way, but they very quickly shouldered the responsibility for the releases, and I didn't have anything more to do with it after some time in the middle of '82.

Bochannek: Did you recruit CSRG people to join you at Sun?

Joy: Not to my knowledge.

Bochannek: You also worked on the network file system at Sun. You did some VIP [virtual internet protocol] work at Berkeley. Sun's tagline for years has been "The Network is the Computer."

Joy: Right.

Bochannek: I believe that's a John Gauge.

Joy: Yes, John Gauge came up with—that's his co-line he came up with.

Bochannek: What does it mean?

Joy: It's cloud computing in a sense, right? It's the web. It says that most of what you care about is not on your machine. That used to be the case in the minicomputer. You had a terminal that didn't have anything on it, and the information was sitting in a shared place. When the personal computer came along, most of the things of interest were on your personal computer. A spreadsheet, a document. It was a glorified adding machine/typewriter with storage, with reasonable carbon copies. When the network came along, what was clear was that there was going to be a recentralization of information. In other words, the information that you didn't create is more interesting than the stuff that you did. I mean, in aggregate. The information moved back, in a sense, to the center with ultimately the web. Now we're seeing that cycle maybe change again, where people are getting apps on their smart phone, and some local information there. The strong trend, I think, is still towards the stuff that's not stored or maintained or run locally on your machine. It's more interesting. If you're disconnected from the net right now, without the network, you don't feel like you have a computer anymore. You feel like there's something wrong with it if it isn't connected to the net. People laughed at it, because somebody said, "No, the network is the network. And the computer is the computer." They just didn't understand how integral to the computing experience the network was. Essential, and almost now foundational, to the experience that people have in using machines.

Jon Plutte [Media Director]: Can I ask you to say, "Sun's tagline was 'The Network is the Computer,'" so we can use that as an intro to this whole—?

Joy: Sure. Sun's tagline was, "The Network is the Computer."

Bochannek: In explaining "The Network is the Computer," you tied a tagline with the current trend to cloud computing. When you approach designing a system, do you evaluate what came before? Do you look at how current developments relate to the work you're doing? How the past has maybe shaped it?

Joy: I've tried several times to reinvent the systems, not successfully. Reinventing the systems around a new concept is rare. I mean, a successful reinvention. When you start up your connected computer, say, it starts generally with thinking the most important thing is local file system. A truly networked computer would look to the network first, and would treat whatever it has locally as just something it saved to speed things up. It wouldn't really depend on what was there. Nobody's really reinvented the machine in that way. In other words, the network protocols and the things you can get from the network are more important than files. Most systems today are still file-centric. What's happened over time is we've created concepts. The systems have so many concepts, all these concepts about files and locks and this and that and the other. You could also maybe reinvent the system around everything as a database. It seems to me that in this age, the most important thing would be to reinvent the system in a way that the most important thing was the network. Or if you want to take it to a higher level, the most important thing is the browser. That's what the Google guys are trying to do with their Chrome OS. Underneath it's still Linux, or something, and it's still got a file system. It's not really cleanly saying, "I believe in the network, the network comes first. And if ideally the network comes "only," that's the only thing I'm going to have, so my system can be simple. So that it's behavior is reliable, because it only has a small number of network-based concepts." That reinvention hasn't happened fully. Maybe it never will but that's the kind of thing I always dreamed of pulling off.

Bochannek: Let me ask you a little bit about work style. You mentioned earlier when you were programming at Berkeley you were hacking at your terminals at all hours of the day. What's your process when you program? There are different views of what programming as a human activity actually is.

Joy: I haven't done a lot of programming since the late '90s. I've been doing other things, writing and more recently investing. I always found that programming is probably equally hard to writing in the sense that you got to come up with some storyline or what the program is going to do. You need a narrative arc, you need a structure for your program, and you also need it to work in the small, basically line editing and spelling mistakes. When you're sitting at the terminal, you tend to be very focused on the small and so if you're not careful you don't end up with anything that's very good. I'd say that there's a difference between writing and typing. It's very easy to sit and just type, and it's very easy when you're a programmer to just sit there and just hack away. Just change things, change things, change things. That generally leads to a real mess. What I used to do is I would make a listing. I'd print out the program. I'd take it away from the computer screen, and I'd write like editor notes to myself about what I wanted to accomplish. Then I essentially hand it to myself and now I'm operating in a different way. Now I have to go do what I told myself to do, but I would distinguish the architectural design from the carpentry, if you will. If you try to do both at the same time, it never worked for me. Now I know people do things like extreme programming and tag team this and that and that's just not something I have any real familiarity with. It always seemed to me that if you didn't distinguish between the strategic and the tactical, between the architecture and the carpentry, I don't know how you can get a really good and stable piece of software as a result. I also think my view is partially colored by the fact that I've generally worked on the mission critical parts of the system. The operating system never should crash. The editor you're using should never crash. You don't want to lose your work, so you have to be thinking about what could go wrong all the time. It gives you a particularly perverse view of the world because you're always thinking about planning for extreme events. What if I ran out of this? What if I ran out of this? What if this gets corrupted or that's the wrong number? What's the appropriate thing to do to keep the fire from spreading, to try to leave the information around so if there's a root cause for this that you can find it? When you're an operating system and many things are going on at the same time, there may not be much evidence left after something bad has occurred. The bad cause and the bad effect may be widely separated. You become very defensive in your style with a much more cleanly thought out and cleanly partitioned way of structuring the system so that it's diagnosable and repairable. This led to my strong desire for things like object-oriented programming and safe programming languages like we try to do with Java because of just the amount of time it takes to track down and really make the systems robust.

Bochannek: You mentioned you moved into hardware.

Joy: Right.

Bochannek: How did you become involved in the SPARC [scalable processor architecture] processor design and what was the role that SPARC played at Sun?

Joy: It became clear to us that the 68,000 architecture that Motorola was implementing was not going to take advantage of the progress of semi-conductor technology. I had been at Berkeley but decided not to be heavily involved in what Dave Patterson had done with RISC [reduced instructrion set computing] because the CAD [computer aided design] tools at the time were very primitive. It was very, very tedious compared to even the primitive debuggers and compilers we had for the software allowed you to be much more productive. By the time we realized that the 68,000 wasn't going where we wanted the Motorola processor line, we went down and talked to Motorola. They weren't willing to do with more aggressive performance. We decided if we could implement a RISC architecture that we could build higher performance computers and so we decided to undertake that effort.

Bochannek: Was that driven by Sun's customers at all? Was there a desire by Sun's customers for that new level of performance? Did you interact with the customers at all?

Joy: In particular, a lot of our customers were scientific computing customers and they wanted better floating point performance than the 68,000's floating point performance, [which] was pretty minimal. We wanted to design a machine that had better floating point.

Bochannek: During the 1980s, Sun's growth was tremendous. A lot of things changed. Sun became a global company. What were the biggest changes that you observed that you experienced during that time?

Joy: If you live in an expanding universe, no matter where you are it's expanding. Wherever you went in the company, the company was expanding away from you. My experience was [that I] started being involved in many things and then I decided what I could responsibly give away. You don't necessarily give away the things that you're best at or you're worst at. You give away the things that you can find people to take. There is a process of being left with the things that are not so easy to do and that you can't find much support for. It's like what I said about my father and relative grading, you're constantly challenged to do what you can do because the things that many people can do you just give away. I ended up trying to refine a microprocessor pipeline, or improve our floating point preferments, or improve the interrupt level of the operating system or whatever level task you're working on. What I was working on was a little bit thorny because it wasn't something I could uniquely contribute to.

Bochannek: What about challenges that weren't technical? What organizational challenges—the employee base increased a lot too. Were you involved in hiring? What was your leadership role?

Joy: Scott did a great job of leading the company and helping the senior technical people spend most their time on technical stuff. As a founder, you're always privileged to get a level of respect from other people in the company. If you treat them with grace and respect as well, you enjoy a very privileged life in that way. The organizational changes never really bothered me. Although, you end up going to meetings and if you knew more than one-third of the people in the room at any meeting, that was unlikely because you ended up going to so many different meetings. Or the meetings in the building you never heard of and you may never be in again. You get that sort of strange feeling with a company that's growing that quickly from four to thirty-thousand or whatever it is; it is a huge change.

Bochannek: Related to this growth I'd like to show you this picture that was taken in the '80s and I'm sure you've seen it before. It's the four of you with new cars.

Joy: Right.

Bochannek: And car phones.

Joy: With very early car phones, yes.

Bochannek: Right. What's the story behind this picture? It's clearly a posed photograph.

Joy: I think it was for—is it for *Fortune* magazine?

Bochannek: It's possible.

Joy: Yes, it was a photo shoot for an article in *Fortune*. The one thing I remember is one of us didn't have a red car so they stood out—

Bochannek: Yes, I think it's Andy having the silver Porsche.

Joy: Yes silver.

Bochannek: Right there.

Joy: Andy was very much for standards. He probably standardized on that color. I think that represented a little bit of the time. Most of those are convertibles and if you remember when they introduced the rollover standards there were very, very few cars that were available in the US that were convertible. he fact that they're convertibles probably says more than that they're sports cars. Those were almost the only cars available with rag tops at that time.

Bochannek: I notice you have a Ferrari Mondial Cabriolet.

Joy: Yes, it's four-seater. It was one of the four-seater convertible cars available in the United States.

Bochannek: How did the success of Sun change your life?

Joy: It changed my life financially, but I think what we were trying to do at Berkeley became impossible in the environment because we couldn't hire the people to meet the opportunity. Sun became an important force in UNIX and the kinds of things we were trying to do in supporting scientific and technical computing continued to be possible. For me, that was a story that I was trying to write at Berkeley and it allowed that. It allowed that to happen. I was very proud. Sun was always a very strong technical leader for the industry. I don't think our marketing and go-to-market strategies were nearly as strong so we'd invent things. Often they'd be years ahead of their time. The tagline "The network is the computer"; it took people years to figure it out and now Cloud computing is just blossoming now. This is 15 years later so we were really a pioneer. I'm really proud of the platform it gave me to do pioneering work.

Bochannek: By the late '80s you still worked on SPARC. You were still involved with that. You moved to Aspen at that point.

Joy: I worked on SPARC until the mid-'80s.

Bochannek: Okay.

Joy: To the mid-'90s, right? Then when I moved to Aspen in '89 and '90, I still worked on SPARC but by '92 or '93 we were working on my attempt to come up with a platform with less concepts. We were trying to build a really small system, cleaner than UNIX had become for a networked age. Eventually, we switched to putting our resources behind Java and also trying to put Java in phones. The Java what was called ME [Micro Edition] platform was one of the platforms we were pushing in 1995 which ultimately became these Smartphones with applications. That was our vision long before the Android and iPhone emerged. That was our vision in the mid-'90s; a dozen years ahead of the time but back when the phones had 8-bit processors just going to 16. Our model was it should be a 32-bit processor and run portable applications among the devices as we see now with Android and Java.

Bochannek: Related to Java, in the 1999 ComputerWorld Honors interview you said about problems in programming, "In the case of the better program we are writing the software in a low-level language. Nineteen ninety-five was when we announced Java and it's really still not completed what we wanted it to be so I would say maybe by 2010 it will be done."

Joy: Right.

Bochannek: What do you think Java would be at the time it was created and where is it today?

Joy: What I was saying is that the vision for Java was to create this portable platform for running applications on many devices. That's not only the language, but also the libraries and that whole platform, that whole platform ecosystem has emerged. I would say that with the emergence of Android, that's what we were envisioning. That took about 15 years for a version of Java plus its application environment to find the market traction that it in some sense deserved or that we imagined. It didn't work the first time with Java ME. Obviously, it never really grabbed people. The hardware wasn't ready, probably. The devices weren't capable enough.

Bochannek: The origins of Java were in more consumer electronics devices. There was the original intent I understand, yet it had to go through this phase of being available on the Internet to now become more widely used in again smaller consumer electronics devices.

Joy: Right, it came out of a project where they built a hand-held device with a touch screen. It looked like an oversized iPhone, right? It had a touch screen, it had the equivalent of Bluetooth and Wi-Fi and an animated user interface, much like the gaming user interface you'd see today on an Android or an iPhone. It ran an interface as a remote control for like your television so that you had a three-dimensional shopping kind of second life kind of experience. It was all prototyped. The thing was quite large because it was power-hungry and needed large batteries, but the amount of memory that was in that device if you plotted it against Moore's law meant it would be practical to do in small hand-held devices within a bounded amount of time. When we put Java out on the Web, the Web came along and Java became a language for plug-ins for browsers. Probably ultimately more done by Flash, what Java could have done was for market reasons done were by Flash than by Java. Then Java was also put on the business computers. We had Java EE [Enterprise Edition] for programming relational databases. It got a lot of traction there which ultimately led Sun being acquired by Oracle. It was only years later when the capabilities of the handsets got better the amount of networking band with the availability of the handsets meant that the network is the computer for phones or for pocket computers that Java really came into its own on a platform like Android.

Bochannek: Java is also being taught widely as an introductory programming language in many colleges. Is this something that you envisioned to happen?

Joy: Yes, I think—

<*crew talk*>

Joy: Java is widely used for programming, Introductory Programming.

Bochannek: As an introductory programming language in many colleges now.

Joy: It's probably a little too complicated a language. I prefer to see a simpler language being used for people to start. Java is really designed for precision engineering. It has a very strong type system. It has a lot of checking. For that reason, it's almost too complicated. It's like if you were just learning to build something I don't give you some high-end 3D CAD system with all of its stuff. I give you a hammer and some pieces of wood and let you try hitting a hammer and a piece of wood to see how the nail grips first. You've got to get a sense for your tools. Like we had in Michigan, I would love to see people have a potpourri of different languages and to learn things and to learn some dimensionality in what they do. It's a better language to teach people than C, but it's still not the language I would think would be best for learning.

Bochannek: You talked earlier about how you take the long view in a lot of projects that something- it's a value you were raised with too. UNIX is 40 years old now. WSE [Web Services Enhancements] project started almost 35 years ago, SPARC is about 25 years old. What gives technology longevity?

Joy: Networks tend to outlast the systems that are attached to them, so of all the things we know that the protocols, the TCP/IP protocols and related protocols will probably last the longest. UNIX has proven to last a long time because the community of users that's had access to the source code has been able to improve it and a lot of things the system does are very hard. The code really has to work. If the virtual system memory has bugs in it, it's simulating a world and the world doesn't have any stable physics and everything just falls apart. That code really has to work the same thing for the networking code and other things. It tends to take a really long time for these things to mature because software is so hard to write. Look at next step or whatever object oriented programming is for Apple; that environment that they have came out of the work that was done when Steve left Apple and formed NeXT. It's really fully mature in the marketplace and people are writing applications now. That's very, very old technology at this point. I wish there were newer things that were coming along that were simpler and had some properties that would be better for the kind of challenges that we face but rate of adoption of them seems quite slow.

Bochannek: Why did you decide to leave Sun?

Joy: I left Sun because I'd done everything I wanted to do at Sun. I'd moved to Colorado to stay in a smaller environment and have a small research group, but eventually the group I had and the company drifted apart. When the downturn hit in 2000, I thought it was more severe and would have done a larger layoff than Scott wanted to do. I felt like a lot of the projects were underfunded because we'd had too many and decided to try to do something different. I'd become more interested in energy efficiency and sustainability issues and so decided to try to find a way to do something on those issues instead.

Bochannek: You spent two decades at Sun and Sun as a company does not exist anymore. How has this affected you?

Joy: I never viewed myself as Sun. There was not conflation of the two. The technology industry generally has companies rise and fall all the time in every era. If you look at the companies in the minicomputer era, only HP survived really. Data General and Digital were sure very large companies [and] are gone. Essentially, none of the workstation companies survived independently. HP, again, was a workstation company and absorbed Apollo. Even in the PC [personal computer] era, Dell remains in the US, but pretty much all the companies are gone. That's not an atypical result. I think the UNIX work that we did survives, the Java work survives and there's still a lot of great people at Oracle working on the technologies that Sun did. I think that all that work had a very positive impact and not a bad economic outcome. A long way down from the peak, but it created a lot of jobs and a lot of wealth and helps solve a lot of problems. I feel really good about the whole trajectory of Sun.

Bochannek: In 2000, you wrote an article for *Wired* called "Why the future doesn't need us". What inspired you to write this article?

Joy: At the time there was a lot of utopianism around and I felt that there were some unrecognized threats from technology. That there were the equivalent of extreme events that could happen and that we weren't managing the advanced technologies very well. This is something had been written about a lot in the '50s but the conversation had gone cold. In the process of doing some work with the Global Poverty Initiative of the Aspen Institute, I came up with a story arc that I thought was a story that needed to be told again in a new time. I took it to the editor of *Wired* magazine who agreed to help me to write the story because it needed to be told in a certain way and I needed help to tell the story. She saw in the way of a treatment for the story that she thought could make a positive impact so we worked on it together and it gathered quite a bit of attention.

Bochannek: In the decade since, do you feel that the fears expressed in this article did materialize?

Joy: I think that we still have a challenge in this century of managing the ecosystem and to sustainability and to manage some of these advanced technologies. We have certainly seen in the situation with Japan with these reactors. It's not so easy to get control of these things once they get a little bit out of control. There's been nothing, no great planetary disaster from that. I mean certainly disaster for the Japanese people at the moment. I hope they'll get it under control, but you can see how hard it is. It's been three weeks now and the situation still isn't stable. If you imagine the bioengineered virus or some of these other things I talked about in my *Wired* article you can—and something that can replicate and be a global menace, it could be quite—dealing with something like that could be almost beyond our imagination. It looks like a science fiction movie or maybe it's too ridiculous to be a science fiction movie or too depressing. We've got a situation where we take billions of dollars over decades and create really, really powerful technologies and those technologies need responsible management. Maybe in a case of nuclear waste over thousands of years, we create the stuff quickly and there are hundreds of thousands of years you have to manage it. These are not time scales for responsible management that we've proven ourselves very good at. We should proceed with caution, but at the moment it doesn't seem like we are. I think dealing with these issues is something we still have to do.

M1: Could you say for us, "In the year 2000 I wrote an article for *Wired* magazine called 'Why the future doesn't need us'"?

Joy: In the year 2000, I wrote an article for *Wired* magazine entitled "Why the future doesn't need us".

Bochannek: In the time since you wrote this article, are there specific inflection points in the technological change that you've observed that guide your assessment of a more positive or a more negative view of the one you express then?

Joy: I think the issues I mentioned there still bear down on us. The climate change issue for me has almost become even more pressing. There, in some of these nonlinear feedbacks in the climate system—the melting of the tundra releasing methane, or the melting of the ice caps changing the albedo—we're on a short fuse for avoiding catastrophic climate change. That's a clear and present danger which I think I can work on the solution side more readily than I can on the policy side of trying to stop these, for most people, abstract threats. What I was talking about in the *Wired* article was a small number of individuals doing terrorism. Shortly after the article, we had September 11th. You saw a small number of people using technology—in that case airplanes—to do something. That put people on that story. Thousands of people started writing about all that. Now we see the nuclear situation in Japan which should cause people to reflect on whether we are really managing all the risks of a life cycle of this technology or other alternatives. Black Swan events do happen. Thousand-year floods do happen. They don't happen every thousand years, they happen often, actually. Some thousand-year event happens every few years. We need to manage things in a way that accounts for that and manage our collective behavior in a way that doesn't tip the climate over so all these things need a lot of attention. I've chosen to work on things that I can work on the solution side of, because working on the policy side is something I'm not very good at and is also depressing.

Bochannek: You mentioned earlier that now you're on the investment side of things which is related to finding solutions. You cofounded HighBAR in '99 and 1999 it became a partner at KPCB [Kliener Perkins Caufield Byers] in 2005.

Joy: Right.

Bochannek: What were the motivations behind that? You're involved in green energy now. How does this relate to your technical experience that you want to apply your knowledge in this area?

Joy: At the end of Sun, I was the CTO [Chief Technology Officer] and looking at sciences other than just semiconductors, physics for possible computing substrates. You can imagine molecular computing or biologically-based computing or quantum computing. I became friends with Amory Lovins of the Rocky Mount Institute and learned more about his view of renewable energy and heading towards a more sustainable planet. I'd always wanted to work with John Doerr. John had been, fairly early in his venture career, the KP Partner on the board at Sun. After leaving Sun and working on some personal projects trying to implement things that were more sustainable, John offered me the opportunity to work with him. It was a lifelong dream to work together and that's what we're doing now.

Bochannek: You're married again, you have children.

Joy: Right.

Bochannek: What advice do you give your children about the technological change they're seeing in their lifetimes?

Joy: They talk to me about it even more than I do. I think they're very project focused also. My son is creating a seminar series for teens to talk about Great Ideas. He took a Great Ideas session at the Aspen Institute and got motivated to try to get a diverse group of teens together. He's going to try to run two seminars this summer which is a very ambitious all-consuming project when you're also trying to get ready to pass your ACTs and get good grades in high school. My daughter has taken a passionate look at, and she's very interested in, theater. Rather than high-tech, she's interested in live theatrical performance. They both use technology in a balanced way in that they're also very much interested in dialogue and storytelling and conversation as a way of getting people to work together to either create great performances or discuss issues and take actions in the world to solve problems. They're not nerds with respect to technology and I'm very appreciative of the breadth they've found in their lives even at their young ages.

Bochannek: You're being awarded the 2011 CHM Fellow Award for the work you did on BSD and cofounding Sun. Reflecting on your career up to this point, what would you consider to be the major turning points?

Joy: Sending out the Berkeley UNIX tapes to people and thereby creating the very large research community for people to do computer research. Not at Carnegie, Stanford, and MIT but at all the other schools that didn't then have access, that could really do computer research and for the first time; that was really critical. Sun's creation of the notion of workstations which allowed people to have really powerful computers in the '80s to do engineering and scientific computing at the desktop; that was a really big deal. At the time, everybody was focused on computer gaming and Atari and all this stuff. They had great graphics for games and we had no graphics for science and engineering. That was a really important thing. Then I really am proud of the way that Sun helped with the development of the Net and the Web in the context of Java and the Web and all the things that went on the network things in the late '90s. Hopefully the best work I ever will have done, will be the work I'm doing on renewable energy right now. Largely investing in companies whose ideas are hopefully at least as revolutionary as anything I've been involved with before. Most of them are still stealth. They've R&D [research and development] to be done before they can fully announce what they're doing but I'm excited about those green ventures. All those things are for me very important.

Bochannek: How has your thinking about the computing field changed over the decades? You went through a number of technological transitions but how do you relate to the field of computing in general?

Joy: When I started in computing, it was really the 16-bit era and it took artists to really to program those machines. When the 32-bit computers like the VAX came along, the playpen got very large and it became possible to be rather sloppy. Then the Web came along and there were some constraints and quality really mattered again. Now we've got another wave of mobile applications and so we've seen waves of software. My biggest frustration with the computing industry was that we were building things in C and FORTRAN which were languages from the '50s which didn't promote the construction of really reliable software. I was frustrated with the unreliability of the systems so I worked really, really hard to make UNIX reliable and to make Java language we put in really reliable code. I think we were only partially successful. A lot of the code in the world is still written in C and very prone to bugs and viruses. I don't think, as a discipline, computer science has really owned up to the necessity of building safe software in the sense that here in California with earthquakes you want safe buildings. We don't have the equivalent of seismic codes for software. I'm disappointed that that's taken so long and hope it will happen someday but it's a long time coming.

Bochannek: What were some of the difficult choices you had to make in your career? What were things you regret, things you'd like to maybe do differently?

Joy: Java was more of an opportunity then for Sun. I wish we had found a way to make it be more used than it was. It's nice that it was used as an adjunct to relational databases and that 15 years later it became an important phone platform, but I think the language could have had much more of a positive impact on the software community in terms of people writing really good programs. It needed the stuff that we were trying to do in Aspen research lab in the '90s. Right before Java came along, we almost needed to do a new operating system. A new open source kind of operating system. Not Linux which is just another version of UNIX but Jinux. Some Java-based thing that was clean and simple and available to the universities as a new platform with a cleaner and smaller number of concepts. I wish that would have happened. I know a number of universities have tried to do that but it never really took hold in the presence of these huge waves of things going on like Linux, like iOS, like Android. All these things were nice, but they weren't the kind of clean simple, transformative platform that I was looking for. Maybe that will never come to be. I don't know.

Bochannek: What inspires you to keep looking at new areas? What do you enjoy?

Joy: What I do now is look for transformative technologies for sustainability and people with scientific advances that seem nonsense when you first look at them because they are so amazing. We're trying to find things like radically better energy storage, batteries if you will. Large, radical new ways of taking wind energy or tidal energy or geothermal energy or solar energy so that those things can be cheaper than the incumbents; cheaper than coal, cheaper than the grid, so that they can go viral essentially. They can be so inexpensive that they'll be widely adopted and change our environmental footprint. They're out there. You have to look really hard to find them. We look at thousands of them a year and find a few. It's really, really exciting when you find the teams that have ideas that are real and producible to practice to then go and try to build a company out of it. Building companies is hard. We're trying to do it for 50 or 60 of these companies right now. At KP, we've invested in that many sustainability companies and that's a lot of heavy lifting. Hopefully, some of those will be the Netscapes, Amazons, and Googles of the sustainability of the green revolution. Some of the leading energy and resource companies of this century will be new. That's our dream and the technology will drive it. It's the technological advantage and tactical advantage, I think, that will give you the breathing room to grow a really large and profitable business.

Bochannek: What advice can you give young people?

Joy: Oh, there's a lot of opportunity out there. If you go, there's a lot of opportunity in computing. There's also a lot of opportunity in the applied physical sciences. At KP, we have digital practice which does the computing stuff. We have a green energy practice where we focus largely on chemistry and physics. We have a life science practice. I think getting a good education and getting a technical education in any of those fields is incredibly important. We have another boom going on in computer technology right now. The green revolution is really important for the next couple decades and as you can see with healthcare costs and the way technology is being applied to medicine, the digitization of the life sciences is going to make enormous opportunity to have a career in the life sciences; a long and interesting career. If you really want a really long career, go into brain science because probably the ultimate biological frontier is understanding the brain. It's a marvelous time. Computing skills apply to all these areas and it will be a great time to be starting a career.

Bochannek: In closing, can you summarize your life, your career up to this point for us briefly?

Joy: I've been privileged to be in the right place at the right time a couple of times. I was very lucky to be at Michigan when we had a great time-sharing system to learn my craft, to cross paths with Ken Thompson, and have the opportunity to move UNIX forward. To be at Sun, to help start Sun at the beginning of the microprocessor era when we could really give people powerful personal work stations for the first time, and to be around when the Internet really went viral with the Web in the mid-'90s. I hope this work I'm doing now on renewable energy and sustainable use of resources will be as impactful as some of these other things have been. Those other things were nice, but perhaps this is even more urgent and I'm hopeful about the work I'm doing now and glad to have the opportunity to do it.

Bochannek: Is there anything else you'd like to add?

Joy: No, I think that was a pretty exhaustive interview.

Bochannek: Great, thank you. I really appreciate your time.

Joy: My pleasure.

Bochannek: Congratulations again for being a 2011 award recipient for the CHM Fellows Award.

Joy: Thank you very much.

Bochannek: Thank you.

M1: Could you say the word "Computer Systems Research Group" and just give a two-line description of what that was?

Joy: The Computer Systems Research Group was the group that we formed under Bob Fabry when we had money from DARPA to help them create an operating system that would allow them portable research applications. It ultimately became the PSD [Programmer's Supplementary Documents] UNIX that ran on thousands of axis all over the world, creating a platform for sharing research and scientific software among a wide range of educational and research institutions.

Plutte: One more thing; in our museum we have what we call the “Exit Theater” and you gave a wonderful description of what advice you would give to a young person today. I'm wondering if you could address that to the camera and do a little shorter version of that as if you were actually talking to that person.

Joy: Okay. Say the question I'm answering.

Plutte: What advice would you give to a young person starting out today?

Joy: I would give advice—one sec. If I was a young person starting out today, I can imagine getting an incredible education in basic science and computer technology and then having the opportunity to apply it not only to the computer sciences but also the life sciences and to renewable energy and materials. All these areas are going to see enormous numbers of new ventures created in new possibilities in the next few decades and really will transform the world. I'm excited about all those areas. [I would] just encourage everyone to get a real good grounding in computer science and the other physical and life sciences.

Bochannek: Alright.

Joy: Can I do one more thing? Actually, my wife has two kids and I have two kids and I talked about it as though I only have two kids. I should say something about, if you don't mind.

Bochannek: Yes, absolutely, please.

Joy: About my family a little bit.

Bochannek: You mentioned you have children. Tell us a little bit more about your children.

Joy: My wife and I have four children and they're marvelously diverse. Our oldest is an artist and very passionate about third world development and going to school in Boulder, Colorado. Our second oldest daughter is interested in psychology and criminal justice and she's also a student in Colorado. Then we have a high school junior who is very interested in biology and physics, biophysics and also in public policy and conversation about policy issues. Our youngest daughter has just been admitted as a sophomore to a performing arts boarding school, Interlochen in Michigan, and is interested in theater and wants to do acting. They're the joys of our lives and constantly astonish us by the various things they're pursuing.

Bochannek: What are their names and your wife's name?

Joy: My wife is Shannon. The oldest son who is studying art is Dylan, Logan is studying psychology, Hayden wants to study biophysics, and Maddie is going to theater school.

Bochannek: Great, thank you.

Joy: Thank you.

END OF INTERVIEW

