Oral History of Bertrand Serlet

Interviewed by:
Hansen Hsu

Recorded September 20, 2023

CHM Reference number: 2023.0158

Hsu: Okay, today is September 20th, 2023. I am Hansen Hsu here with Bertrand Serlet. Nice to have you here.

Serlet: Thanks, Hansen.

Hsu: So we'll start with where and when were you born?

Serlet: I was born at the end of 1960, near Paris, in France. Although I didn't stay there for very long. My parents relocated. My dad got a job in the Alps and so we relocated to a little village in the Alps and that's where I spent the next five or six years.

Hsu: And what were their backgrounds?

Serlet: My mother was a teacher at some point and my dad was an engineer.

Hsu: What kind of engineer?

Serlet: He was a chemist, originally, yes, but he went into management at some point.

Hsu: Oh, okay. Do you have any siblings?

Serlet: Yeah, I have a brother who's a math teacher, a university teacher in France.

Hsu: Great. Any particular religious or political leanings?

Serlet: No, no. In France, most people are nominally Catholic. That was at least the case from the time I was born, but my family was not.

Hsu: What were your favorite subjects in school?

Serlet: I liked school in general, but, elementary school and all that, that's an old memory, so I don't recall. But I like puzzles, I like math, that kind of stuff and of course, high school was all the classical curriculum.

Hsu: Yeah. How did the curriculum in France differ from, say, the curriculum here in the States?

Serlet: It's probably more classical, but I was, I consider it very lucky because when I started in middle school, there was a switch to modern mathematics and that was set theory, and essentially it was done to prepare people to the modern world, right? There was a world of computers and all that. So I think that had a really good impact on my development.

Hsu: What kinds of books or media did you read or media did you consume?

Serlet: Oh, only science fiction books. Just, I devoured science fiction books all my teen years.

Hsu: And what sorts of hobbies did you have?

Serlet: Really, not too much. Now, I must say that one of the most profound things was a gift I got for Christmas when I was 10 years old, where my parents got me a computer, okay? It was a toy, really, developed in collaboration with IBM and it was a kind of a mechanical thing where you had three bars that you could move horizontally. So that was the input, 3-bit input and there was a button you would press, and it would light up some lights and there were four lights, so that was 4-bit output and you would program it by inserting metal tacks in some holes, depending on the position of the bars. So there were three times four times two, so that's 24 holes that you could program. So it was essentially a 3-bit input computer, 4-bit output. Which sounds very limiting, but actually, that's a lot of different programs you can do, that's a few million programs with that setup and so all of Christmas of that year, I just explored how you can use this, okay? And there was a little booklet that came with the toy that showed how you could program a few games and so I programmed those games, and then I invented new games and so it was an act of trying to modelize some real-world thing with a 3-bit input, 4-bit output computer. So that was very formative.

Hsu: Wow, that sounds really fascinating. What a great toy.

Serlet: Yeah, so that was a total accidental. My parents were not into computers or anything. Prior to that was the summer of '69. Of course, I was fascinated with space and working on the moon. But already then, I was intrigued by computers and so after that, I became very enamored of computers and then there was a second totally random thing that happened, which was sheer luck. I was living at that point in Normandy, we had relocated to Normandy, and it was a fairly blue-collar town and farmers, really not the high end of education. But France decided to seed high schools with computers and there were 60 computers seeded throughout France and my high school became one of the recipients of those computers and so from the age of 13 to 17, all my brain power was dedicated to how to program that thing and it was programmed in something like BASIC and that's where I experimented. I learned on the spot all kinds of things. I understood how to do recursion and things like that.

Hsu: So how old were you when you got that initial toy?

Serlet: So the initial toy was for my Christmas, when I was 10 years old, exactly and then the more sophisticated school computer was when I was 13 or so.

Hsu: So did you have any influential teachers or role models, mentors growing up?

Serlet: No, not really. This was not a school with famous teachers, I should say. It was a fine high school, but I was a little bored with all the classical curriculum, but I would go to the computer all the time I could spare.

Hsu: Okay, so we just covered the Christmas computer. So then, okay, you mentioned in high school, so your high school and middle school were combined for that one year [ph?]?

Serlet: Well, so I spent those seven years in the same place, okay? And it's continuous, there's no big difference in France between middle school and high school, so all the seven years were continuity, moving along step by step.

Hsu: So then you went to college starting in '77?

Serlet: Now college in France is a little different. You have this thing that Napoleon created that's called Grande École, which means big school, and you have two or three years of preparation in order to enter those schools. So that's the whole college, somewhat equivalent. So I went to Rouen for those preparative classes and then there's an exam, multiple exams, actually and I landed in this big school, which is called École Nationale Supérieure de l'Enseignement, ENS, École Normale and that was in near Paris.

Hsu: Did you already decide what you were going to study?

Serlet: I was studying math and the vague goal, but I didn't think too much about it, was to teach math at university level. But after essentially a few months in that school, I realized that my colleagues were much better at math than I. There were some people who were really formidable. So after a year, I said, well, maybe I should try physics. So I took a year of quantum mechanics, which I thought was a thing to learn in physics and it was a total disappointment, because I did pass the exam at the end of the year, because I had all this math background. But I didn't understand a thing. So I was very disappointed, because I wanted to learn and understand and I didn't. So I said, okay, math's too hard. Physics is too hard. I'll do what I know how to do, which is computers.

Hsu: So what sorts of computer systems did you use during this time?

Serlet: Oh, so when I was in the prep classes, so the beginning of college, my girlfriend had an HP programmable calculator, that was a big thing in those days and so I totally explored that and really enjoyed that. After that, when I went to the École Normale, it was kind of a setback because I had to use punch cards. It was a big multi-user system on the campus, and you had to punch your cards. I remember I built a compiler at that time, and it fit in a rack of cards and my big fear was not that I had a bug in the compiler, but that I'd trip while carrying my set of cards. That would have been the end of the compiler.

Hsu: I have here, you got a Z80 computer in 1980?

Serlet: Oh yeah. So when I was in college, in parallel with the curriculum, I bought this little computer, a Z80 based. The Z80 was a microprocessor that was fashionable in those days and for some reason, England was a place where a lot of hobby systems were made. So I went actually to England to take hold of my computer. I was not good at soldering and all that stuff, so I had it built for me. Normally, most of the things were coming as a kit, and you had to solder and all that, and that was not me, very clumsy, so I got that Z80, and it was fun. I programmed not in a high-level language, not in assembly, I programmed in hex. So I had the instruction set in my head, and I would program by just typing the hex. But it was still quite an upgrade from my 3-bit input, 4-bit output computer.

Hsu: Were there any important people that influenced you in college?

Serlet: Yes, yes. When I started to switch away from mathematics, I started in parallel to take computer science classes. At that point, I had done computers for over 10 years, but I still had no formal education whatsoever in computer science and so I took a class, something that was equivalent to a master and my professor was Jean Vuillemin, and I just loved his teaching. He was teaching simplicity, elegance, and that really resonated with me and also, his domain that he was teaching in those years was VLSI, so chip and that was, the V stands for very, very large-scale integration, but very, in those days, was a few thousand. Okay, so it was just the beginning of integrations of chips. But he had the foresight to see that the V will get, of course, much bigger V, right? And that's Moore's Law, of course. So he taught me the basics of VLSI, and he, frankly, took me under his wing, and I became, I ultimately did my PhD with him.

Hsu: Before we go there, I wanted to go back and ask, what was the first program you ever wrote?

Serlet: Well, it depends on which machine. When I was 10 years old, so that was just games, like guessing cards. After that, when I got the school computer, which was a much more hefty computer, my first program was chess. Okay, now, okay, I should quantify that. It was just playing by the rules, but it was not doing a deep search or anything like that. But I had essentially no programming experience at that point, okay? So that was a great learning experience, how to modelize the chess game and program the moves and so forth.

Hsu: You went on to graduate school to work with this professor, and so that was, but that was at a different school or the same school?

Serlet: Well, okay, so it's a little complicated, okay? The French system sometimes is a little hard to understand, but so you have this prep school for two years. Then I entered this École Normale, where I was supposed to be a teacher. So when at École Normale, you actually take classes at the university. So I was, my diplomas are from the University of Orsay, which is south of France. But in parallel, I did this computer work with Jean Vuillemin, and he was working at a French research institute called INRIA and so I did, I was a researcher essentially at INRIA, I had a double job, and as time progressed over those years, I spent more and more time at INRIA, because that was where all the excitement was.

Hsu: INRIA physically is located where?

Serlet: It's located in the west of Paris. The University of Orsay is in the south, INRIA is in the west. It's a research center and the computer there was Multics, which is kind of the ancestor of UNIX. So UNIX was a reaction to simplify Multics. Multics was very complicated. But people just started to understand how to do those kind of systems.

Hsu: I'm surprised that they used the Multics system.

Serlet: Yeah, yeah. Yeah, that was in 19, I was there from '80 to '84.

Hsu: And what was your thesis about?

Serlet: My thesis was about doing CAD tools for VLSI and it was how to represent circuitry. Nowadays, everyone uses Verilog. But Verilog didn't exist in those days. So this was kind of… has similarities with Verilog.

Hsu: So HDL types?

Serlet: Yes.

Hsu: So then you graduated with your PhD in 1983?

Serlet: '84.

Hsu: The work that you were doing at INRIA was part of your thesis or were you doing other stuff?

Serlet: Yeah, it was part of the thesis. But there was some fuzziness. So I spent a lot of time at INRIA working with Jean-Marie Hullot, who was a longtime colleague and friend, and who died regrettably a few years ago, and he was developing object-oriented systems for LISP and so we were just, I was helping him on that because it was fascinating and so even though the main topic was VLSI, there was a lot of object-oriented framework being developed and stuff like that.

Hsu: You knew Jean-Marie Hullot already at INRIA?

Serlet: Yeah.

Hsu: He was there. For what years?

Serlet: He was a researcher there from the late '70s till the year '80. I don't know exactly, '85, '86.

Hsu: So there's that connection there. Is there any more about your time at INRIA that we haven't covered?

Serlet: It was very formative. I think Jean Vuillemin is a formidable person, and it was great to learn from him and he was my first mentor. I've been very, very lucky that throughout my path, I've had mentors and Jean was fantastic. He gave me some of the fundamentals of computer science algorithmic-- I discovered algorithms with him. Again, before him, I had zero academic background, it was just learning by doing. He had the knack to simplify problems to the substance of them. Now, I learned many years after that, in fact, sometimes simple problems have regrettably complex solutions. So some of his teaching was that simple problems have simple solutions. Well, that's not always true, regrettably. But nevertheless, he was a great mentor.

Hsu: Then in 1984, you relocated to the US to come to Xerox PARC. Is that right?

Serlet: Yes. Yes, that's right. I had visited Xerox PARC for several weeks the prior year with Jean and there I met Louis Monier, who is a great friend of mine and so I got a taste of PARC and how much advanced compared to the rest of the world, including INRIA, it was, and so I just wanted to go there. I had no attachment to France or in France. So I said, yeah, let's go there. So I visited Louis in 1984 because I was going to a conference somewhere in California and so Louis said, why don't you do your resume? So there was an Alto lying around, so I just wrote my resume on the Alto and interviewed that week and they made me an offer.

Hsu: You started at PARC in September of '84, and you joined the Computer Science Lab?

Serlet: Yes, yes. PARC was an interesting place, to say the least, and there were three labs, there was CSL, where I was. That was a lab that was working in Mesa. Mesa was a very C, C++-like compiled language and everything that was done in the lab was done in Mesa. There was another lab in the other third of the building that was done in-- where everything was done in LISP, Interlisp and there was a third lab, where everything was done in Smalltalk and there was a bit of competition between those labs. There were some talks where all the labs would come and listen to whatever was presented. But by and large, the three labs were independent and competing. It's a little bit crazy to think that in '84, '85, I had access to an email system that had fonts, text with fonts, that had images, attachments, and the other two labs also had access to a system that did that. But it was three different systems and in the entire universe, that was the only place that had such a system, right? And of course, there's Metcalfe's Law, which is the square of the interest of a system is the square of the number of users, right, for a network system and this was really silly, to have three labs duplicating all the effort and the worst thing is that those systems were not compatible. So there were three mail systems with fonts and attachments that were incompatible in the building of Xerox PARC in Palo Alto.

Hsu: You were primarily working in the Mesa Cedar…

Serlet: Yes, totally. Totally. There was no, no, no crossover. That would have been bad.

Hsu: What was your area of focus?

Serlet: So I was focusing on design aid for VLSI. We were building a chip that was a multiprocessor and one of the first multiprocessors in those days with coherent memory and all this, and so I was developing the design aids for that chip.

Hsu: This is already after, I believe, Doug Fairburn is no longer at PARC at this point.

Serlet: Yeah, that name is not familiar.

Hsu: Yeah, okay. Because he was a VLSI guy at PARC before. Who did you work with at PARC?

Serlet: So I worked with Louis, Louis Monier. I loved, I worked with Pradeep Sindhu, Rick Barth. But my really, the person I learned the most from, was Russ Atkinson and Russ was not working on VLSI per se. He was working on the Cedar and Mesa system and I essentially learned some very important things from him that followed me all my life. One of them was the importance of interfaces, programmer's interface, APIs. The APIs hide a lot of the code of the implementations, they abstract it and he had a sense for APIs. He designed beautiful APIs. So I just tried to imitate his style and really get it in my guts, which I did, and it was beautiful the way he was writing code. So that was probably my second mentor and he also had some tricks in his bag. Like one of the tricks was immutable data structures, and in the main Mesa Cedar environment, many objects were immutable, notably all strings and nowadays, every programming language that has occurred over the last 10 years has, or 20 years probably, has immutable strings. So this is commonplace. But this was fairly novel in those days, and it really worked well, and I got the spirit of when to use immutability and when not. The other thing in the Mesa and Cedar environment is there was a lot of hash tables and hash tables is one of the most useful data structures. Most… at the same time basic, but not totally trivial. It has nice properties and again, if you look at modern languages, Python, things like that, they all use hash tables very significantly.

Hsu: What was PARC like at this point in the 1980s?

Serlet: So PARC probably peaked before I arrived there and there were some big names, many of the older generations who had left just prior to me arriving there. So it was a place in the decline. But it was still way ahead of any other place in the world, really. It invented technology in so many domains, VLSI, networking, personal computers, the list goes on and on, and so there was so much to learn there. So I stayed four years, but I really learned a lot there. I left unhappy because I was very frustrated because it was a great place, but for 40 people and doing all this work that we were doing for just the benefit of 40 people was very frustrating. So I wanted to have what I do be used by others.

Hsu: What sort of hardware did you use when you were there?

Serlet: So there was a successor to the Alto called the Dorado. That was essentially all built with discrete chips. It was quite a formidable accomplishment to have built that, and it was very fast for those days, but it was amazing. It was more powerful than the VAX I had been using, but it was my own computer as a personal computer while the VAX was shared with lots of terminals. So it was a great hardware environment there.

Hsu: The Cedar technology, I think PARC was trying to spread it to Sun systems around that time?

Serlet: Well, there was a project that was a follow up of our multiprocessor project. Our multiprocessor was called the Dragon, and it became clear around the time I left that this project was not going to be successful at PARC and so PARC and Sun made a joint agreement and Pradeep notably went to Sun, along with another colleague, Jean-Marc Frailong. So Pradeep and Jean-Marc Frailong went from PARC to Sun and made the Sun project happen.

Hsu: So PARC just recently merged with SRI. So given that maybe could you reflect a little bit on PARC's legacy and its contributions?

Serlet: Oh, it has many, many forms. I think it touched all the fundamental, all the basics of computer science. So in many ways, it is an example of what happens when you get a great group of individuals and you give them a lot of freedom and all that. But it's also an example of total failure of transforming that to a success that benefits many other people. The main benefit of PARC has been visitors like Steve Jobs coming and learning from PARC, having trained all the scientists and then the scientists move elsewhere. So it gave me a sense that research is not a way to do things in computer science, and I think it's different in, for example, physics. In physics, you want academic research, right? In computer science, it's also a team effort and PARC was not so great at team efforts. There was a lot of individual personalities who wanted to publish fundamentally as a main goal. So many, many years later, when I started at Apple, my boss, Avie Tevanian, given my research background, asked me to investigate the Apple research and I was not so excited with what I saw. So, yeah.

Hsu: Speaking of Apple, though, what was your first experience with a Macintosh?

Serlet: So towards the last year at PARC, I had this dissatisfaction of getting people to know what we're doing. So I started an effort to publish a lot of what we had done just to spread the word. I also had this plan to maybe have a startup with designers and so I went to see someone at PARC who was in charge of technology transfer. Startups funded by Xerox to take the technology out and essentially this person totally discouraged me from doing a startup. So it wasn't successful in that way. So I started looking what should I learn about? And so I got a Macintosh, a small Mac, and started programming on the Mac to learn the Mac environment.

Hsu: Was that the original Macintosh?

Serlet: No, that was the Mac SE. So that was, yeah, that was, I think by that time it was '88. But then my longtime friend from INRIA, Jean-Marie Hullot, had rejoined NeXT, working for Steve Jobs and he said to me, you should come in. So I interviewed and by the, towards the end of '88, I decided to move to NeXT.

Hsu: What was that first meeting with Steve Jobs like?

Serlet: Oh, so that was an interview. So Steve interviewed me. I had interviewed with the rest of the team, and that was in October '88 and that was before the announcement of the NeXT computer, the Cube, that was unveiled just a few weeks later. So Steve brought me in front of the Cube and he said, let me show you something and he goes to a terminal, a Unix terminal, which I was shocked, okay, that it wasn't like all [graphical] user interface based and all that stuff and he launched a music player from the terminal and he said, listen to this and he played classical music. I hate classical music. I didn't tell him. But his demo was a flop with me, and then to grab the classical music, and starting with a Unix terminal, which for me was going backwards right, in history, because I had been working on a VAX at INRIA, doing Unix in the early '80s. But I let that slip and I did join NeXT.

Hsu: That's interesting. That's the first sort of Steve Jobs demo flop I've heard. So which team were you joining?

Serlet: I joined the AppKit team, the Application Kit, which was totally aligned with my background, object-oriented programming and I did a lot of things at NeXT. I helped Jean-Marie in many ways. I was kind of Jean-Marie's helper. And very happy doing so, Jean-Marie had a knack for UI. So Jean-Marie was another mentor. He was my boss, my friend, my mentor in object-oriented programming, in UI, in how to present. So I learned a lot from Jean-Marie.

Hsu: So Jean-Marie famously created the Interface Builder. So he was part of the AppKit team?

Serlet: Yeah, but he had a little bit of a special status because he had a direct line with Steve. I think Steve considered Jean-Marie as a personal friend, as well as the Interface Builder guy.

Hsu: Right, okay and was William Parkhurst still...?

Serlet: Yeah, yeah, we overlapped for a while, Bill Parkhurst, yes.

Hsu: Who else was there? Trey Matteson was there?

Serlet: Trey Matteson, yeah. With Trey and Ali Ozer, we did the round two of NeXTSTEP, which was called OpenStep, which notably introduced immutable strings and hash tables, and a few things like that, and that was a major cleanup of the APIs and that essentially, I mean, jumping gears, became Cocoa on the Mac and on the iPhone.

Hsu: So Ali was there when you joined as well?

Serlet: Oh yes, yes. I worked a lot, really enjoyed always working with Ali and Trey. Yes, yes.

Hsu: Who else was in the AppKit team at that time?

Serlet: There was a new kid called Scott Forstall. There were a bunch of others, but it's starting to be quite a while back.

Hsu: How much continuity, in terms of your experience, was there going from PARC to NeXT?

Serlet: Well, the things I learned at PARC, I think, were very applicable. API, hash tables, immutability. But there was no codebase continuity. It was just what I learned, which is one of the things that's a little sad about Xerox PARC, is all those people learning all that stuff, but could not apply it unless they left.

Hsu: You mentioned that the [NeXT] Cube was announced pretty soon after you joined.

Serlet: Yeah. Well, actually, pretty soon after I interviewed.

Hsu: Oh, after you interviewed.

Serlet: I joined formally on the 2nd of January. Although I did show up on the 31st of December, just before midnight, with a bottle of champagne and it was a startup, so there were a few people around, notably Avie, and then we drank some champagne around midnight.

Hsu: So you formally started in ’89 then?

Serlet: Yes. January 2nd.

Hsu: January 2nd. One thing that I discussed in some of my previous interviews was you contributed to the creation of protocols in Objective-C.

Serlet: Oh, yeah. Yeah. The concept [of categories] was to extend a class outside the class, because very often you have a class, say a person, and there's a first name and last name of a person, and what you would like is to extend it and say, I want to define full name, and full name is just first name, space, last name. But I want to define that outside, without changing the code of the person, because it's a different subsystem. Maybe I'm building a layer above, or whatever, for whatever reason, and so you want to enhance a class and now in Swift, it's called extension and the other piece of that is, I want to extend this, define this last name, this full name, in terms of first name and last name. But I want to do that, not just for this object, that's the person object, but I want to do that for an employee. I want to do that for a student. I want to do that for every class that abides by a certain protocol, okay, and so that's where we had the protocols and the extensions on protocols.

Hsu: Right and where did this concept come from?

Serlet: I think it came from the need of trying to not rewrite always the same code.

Hsu: Were there similar ideas from other languages that you're drawing from?

Serlet: Oh, there were. There's always been a lot of influences. I had read about Eiffel, because it was created by a French guy, and that had definitely impacted me, but I never programmed in Eiffel. So I had read a lot of the literature. So yeah, it's, when you do something new, it's often someone who did something kind of like that in the past, in computers.

Hsu: So protocols was necessary in order to do Distributed Objects?

Serlet: Ah, yes. So one thing that we had at NeXT was the Mach operating system. It was developed at CMU by Avie and others and Avie had convinced Steve to base NeXTSTEP on the Mach operating system, which we did, and Mach was very elegant. It was all based on message passing and so you would, nowadays, we call that microservices architecture, you would tell the service, okay, please do this. And it was labeled as an object-oriented operating system. That was a little bit of marketing, but more than anything. But we had an object-oriented language, which was Objective-C. So with my colleague

Lee Boynton, we thought, well we can probably automatically map the message of the language to the messages of the operating system, and that became Distributed Objects and that was great. It lent itself to great demos. In the end, I think it was not that useful, but it was great for demos.

Hsu: Talk a little bit more about your collaboration with Blaine Garst and Steve Naroff on Objective-C.

Serlet: Oh, sure, sure. Yes. Steve was the master of Objective-C, of the compiler, of the tools, and so when we did these extensions of protocols, we did that with Steve and with Blaine. Blaine Garst.

Hsu: Blaine Garst. So we covered Interface Builder. Talk about the Workspace Manager.

Serlet: Yes. So the Workspace Manager in NeXTSTEP was the equivalent of the Finder on the Macintosh. So this is a place where you go to access all your files and there was a first version that was done in ’88 and Steve was not totally satisfied with it and Jean-Marie said, well, there's a few ideas on how to do that, how to do a browser and all that, and some of those ideas were coming from the work he did on Interface Builder. So Steve said, let's redo the Workspace Manager and Jean-Marie, you do it. So that's when, with Lee Boynton, the three of us worked for a year on the Workspace Manager and it was a very exciting time. I was living in Palo Alto and I typically was working all night long and I would meet my wife at the cafe at 7:00 a.m., I was on my way to bed. She was on her way to work and we had this rhythm of working at night for about a year. We would usually have breakfast with Jean-Marie and Lee at Denny's around 3:00 a.m. just to get a little boost of energy to keep going, and we did a lot of work. Just the three of us did pretty much the Workspace Manager from scratch in about a year. But it was really hard work. But I loved it. We all loved it.

Hsu: So this new version of Workspace Manager came out in what year?

Serlet: Oh that, I think that was probably by then '90. Yeah, it was '90. Because in '91, Jean-Marie relocated to France and I followed him after a few months and I spent a year in '91 in France and we, at that point, we were still maintaining the Workspace Manager, but it was done. It was shipped as part of the version of NeXTSTEP.

Hsu: So you were working remotely then?

Serlet: Yes, yes, yes. We were working remotely. There was Jean-Marie and there was Bryan Yamamoto who was also working with us from Paris.

Hsu: So the three of you were all in Paris. So I guess, first, why did Jean-Marie relocate and why did the three of you, like all of you, go?

Serlet: Jean-Marie preferred living in France. His wife was there and so, I would say to Steve [Jobs’] chagrin, he [Jean-Marie] relocated to France and I followed him.

Hsu: You were there for how long?

Serlet: I was in France for just a year. Towards the end, Bud Tribble and Leo Hourvitz, the folks managing the software team, thought it was not a good idea to have in France the people doing Interface Builder, doing Workspace Manager, doing Project Builder, and doing Mail, which we had started doing at that point.

Hsu: Oh, so your team was doing all of those?

Serlet: Yes.

Hsu: So you worked on all of those products?

Serlet: Yeah, different times, yeah.

Hsu: Then, did Jean-Marie stay or did just you and Bryan move back?

Serlet: Yeah, Jean-Marie stayed and I moved back.

Hsu: Then, after that, you interfaced with him remotely?

Serlet: Oh yeah, oh yes, yes and when I came back, I was a little, tired is not, I should say, maybe a little bored with programming apps and programming frameworks and all that. So that's when I decided maybe I should try management. I had been avoiding it for-- as much as I could, but so that was a point where I said, okay, I'll try it for a bit.

Hsu: What was the first team that you managed?

Serlet: Well, first I had one employee, which was Blaine Garst. Then I had two, then I had four, then I had eight. I'm just kidding. But it more or less followed that.

Hsu: So you had Blaine, so then what was the team working on?

Serlet: We were working on Foundation and that's the OpenStep non-UI base, all the basic collections and that's underneath Cocoa today still.

Hsu: So basically, is that prior to OpenStep, there was no Foundation framework?

Serlet: Correct. The AppKit had both UI and non-UI and we thought that from a layering perspective, it was important to separate what's non-UI from what's UI. Because for example, you can develop services that are non-UI that can leverage this space.

Hsu: Talk more about the OpenStep initiative, like that whole strategy.

Serlet: Well, so OpenStep was just doing some cleanup of the version one of NeXTSTEP. It was necessary cleanup. There were things that were done, it's called now technical debt, that were done quickly, and we needed to just clean up the code and have a more solid foundation, so to speak. So that was a technical project, really, not based on any business and Bud Tribble had left NeXT at some point and went to Sun, and Bud Tribble said, well, maybe Sun should use OpenStep and so then there was this alliance deal between NeXT and Sun and that was, ultimately, that didn't go anywhere. Sun didn't do much with it. But that's when it happened.

Hsu: So it was only rebranded OpenStep after the Sun deal? Or was it already being rebranded?

Serlet: Yeah, I think it was rebranded along with the Sun deal. I am not sure. Maybe we had used that prior.

Hsu: There's a whole bunch of changes going on, right? So there's Bud leaving, there's the decision to port NeXTSTEP to Intel, and then there's the whole pivot away from hardware.

Serlet: Yeah, yeah. NeXT, I think, had six or seven pivots. NeXT never really worked from a business perspective. It was a great place to do work that was at the leading edge of technology in a number of ways, but we never found this juicy niche. We had a few false starts, okay, where we thought it was working, and then after a few quarters, no, no. So we do another pivot and so the company pivoted many times. It started as the company doing a computer for education. I think, in Steve's mind, it was another Apple, right, kind of done right. But that was a good entry point. But it was way too expensive for education, education cannot afford such a computer. So then it became the interpersonal computing, all based on Mail. We had a great mail system, again, with fonts and attachment. That was probably one of the first commercial systems with fonts and attachment. For me, it was déjà vu, because it was like four or five years after… More than that, it was six or seven years after using such a thing at Xerox. But it was new for the industry and that got some traction, but not really. Then we let go of the hardware, and for a while, we had PCs. We had Windows machines, OpenStep for Windows and it's only towards 1995 with the internet starting to work, right. So the first [web] browser was, of course, developed on NeXTSTEP and we had this database product, DatabaseKit or something like that. That got some traction, because people in enterprises were able to take the SQL database and produce a web interface for that and so we did a version two. I hired a young engineer from Oracle, whose name is Craig Federighi, to really contribute on that project and he quickly became the thought leader on that project. That became EOF, Enterprise Object Framework and then we had WebObjects, which was on top of that, led by Nicolas Popp, and the combination of EOF and WebObjects was really fantastic for the time, because it let people create dynamic websites based on information that's stored in the database and it was very easy to do and we had really great customers, notably Dell. Dell was selling 100% of their computers using WebObjects and so that was towards 1997 and we were very excited. Finally, after all those pivots, we had found one that worked, and we were starting to do the work to go public and then we got acquired by Apple.

Hsu: I want to go back to. One of those pivots was mission-critical custom applications. Can you talk a little bit more about that?

Serlet: That was trying to use our object technology. That, again, was pretty advanced for the day and there was no Java yet at that time and object orientation lets people write less code to have the same functionality and so many people had businesses where they had some business logic, some code, and they could rewrite it using our object technology and that would save time and so we had a few customers that were interested, but not enough to make a real working business out of it.

Hsu: Could you also talk a little bit more about the third-party developer community for NeXT?

Serlet: Oh, yes, yes. It was very active. There were some great apps. There was, in particular, an app to do presentations that both Steve and Jean-Marie really enjoyed and they did great presentations with that app and that will get revived many years later under--[by the] name of Keynote.

Hsu: So I read that Bud Tribble... What was the exact circumstances of him leaving? Do you know?

Serlet: I don't know exactly. I think I was in France.

Hsu: Oh, okay.

Serlet: So yeah. So there was this thing happening in the headquarters. But I was pretty disconnected.

Hsu: So after Bud left, there was some jockeying about who would actually take his place. Did you see any of that happening?

Serlet: No, no. Again, I was in France, and frankly, at that time, I was not very interested in that side of things. I just wanted to build amazing products.

Hsu: Did you have any interaction with Dan'l [Lewin] when you were there?

Serlet: Yeah, yeah. We had a few. Yes, yes. But I was an engineer. So again, I wanted to stay away from management in those days as much as possible and then I flipped the switch. Right.

Hsu: Talk about the culture of NeXT.

Serlet: So it was a culture trying to do great things, leading-edge things, and I think this is really important in any company to try to push the envelope, and so the folks that we hired were folks that were interested by that side of things. Now, as always with a startup, you also expect, hope, that one day you'll have a big exit and all that. But for some people, this is a very, very minor thing. They just want to do great work and benefit users. So the people who were mostly interested in the big exit left over the years. At each pivot, we would shed a few of those people and the core of the team that was left was really trying to push the envelope and make things progress and do great products and that was, I think, the strength of the team at NeXT.

Hsu: I heard it was also kind of a very aggressive culture in some ways.

Serlet: No, Steve [Jobs] was sometimes aggressive. This was a younger Steve, compared to the later years at Apple, and so he still had some of this aggressivity. But I think overall, it was pretty tame.

Hsu: You mentioned working with Bryan Yamamoto, Leo Hourvitz. Anyone else you want to talk about?

Serlet: Oh, there are many. Paul Hegarty. Yeah, there were many folks.

Hsu: So Swift now has basically extensions on... Basically, with the equivalent of categories on protocols, but I heard that this was actually proposed at NeXT.

Serlet: Well, yes. Yes, the extensions were done at NeXT. Yes, I hacked the new compiler frontend to add them. Yes.

Hsu: But I heard that adding protocols to categories was something that would have been done then, but due to the Sun relationship.

Serlet: Yeah, I think we stopped doing it at some point because it was complexifying things. Having to change a compiler was not a good thing and we did it some other way. But the concept was in the air.

Hsu: So the OpenStep strategy, it started with Sun, but it also included HP, Intel as other target platforms?

Serlet: Yeah, yeah and we had this product called Distributed Objects. That was based on the work we did with Lee on automatically remoting kind of objects. But it was also Foundation, it was a bunch of other things and we started having new folks on the project, like Toby Patterson and we started porting that to HP and some other machine, I think, over time.

Hsu: How were the platforms decided? For instance, like why HP? Instead of another Unix platform.

Serlet: Yeah, I don't remember how that was decided.

Hsu: We talked about shifting to management. So over time, you mentioned your team got bigger and so did your responsibilities grow as well?

Serlet: Yes, yes and so that was another time where I had two mentors. I had our HR person called Caroline Morse and she really taught me the ropes about management, and management is first and foremost caring about the people and she was a great HR person who cared about the people and she taught me that side of things and to pay attention to signals and all this. So she was really a great influence and frankly, my transition happened very smoothly and that was thanks to her, the people side of things and then at the same time, I learned also the ropes for project management and that was from Charlie Kleissner, who at some point became the head of software at NeXT and Charlie, being Germanic from origin, knows how to do a very strict process and he had some great techniques for how to manage a project very systematically and to make sure that the project just keeps progressing all the way to shipping. So I learned that from Charlie. So that's really the triptych, right? It's a technical side, which I had. People side, which I got from Caroline, and the project side from Charlie.

Hsu: So at the point right before the acquisition, what was your team? What did your team [do]?

Serlet: So I was in charge at that point of WebObjects and EOF. So the company was, the software was, well, the company, since the company was all the software, was really divided in two. There was the OpenStep for Windows product that was continuing. We had some maintenance deals with some customers and so forth and we had the WebObjects and the EOF that was really exploding. Having customers, revenues, and all that. Rick Berman was in charge of the OpenStep for Windows and I was in charge of WebObjects and EOF and when the acquisition happened, that was just around my birthday, just the day of my birthday, I was supposed to go back for Christmas to France to see the family. I had a small baby, wanted to go, but I was so excited about the acquisition that I canceled the trip. I decided to stay around here and prepare and I went to see Avie and said I'm really excited by what's ahead. Apple, bought NeXT for the OS. But I'm WebObjects and EOF, can I move into the operating system space? And so with Rick Berman, we swapped, Avie swapped [our] jobs, and so I was, on January 1st, I was in charge of OpenStep, which became the OS.

Hsu: Avie told us that prior to the acquisition, NeXT was preparing for an IPO, and he showed us the documents and it would have been-- the business would have been focusing primarily on the WebObjects stuff.

Serlet: Yeah, that's my recollection of things. But I kind of knew that WebObjects inside Apple was not really aligned with the mission and what was aligned with the mission is the OS. That's why I wanted to go back to the OS.

Hsu: You had not been privy to any of like the talks or the bake-off that happened?

Serlet: Yeah, yeah, I did. Because, so just a few weeks before the deal actually got concluded, Avie said, well, we have some visitors from Apple. They are interested in learning about NeXTSTEP. Can you unearth NeXTSTEP from the closet where it's been gathering dust for the last few years and gather a few people who know things about NeXTSTEP. So many of the folks working on NeXTSTEP had left over the years. So there were just a few of us. I don't recall who, but we were just like four or five, along with Avie, who knew anything about NeXTSTEP. One of them was Ali Ozer. So we gathered in a room and we showed the Apple visitors, with technical evaluators, what we had in NeXTSTEP and OpenStep and all those things, and they were very pleased with what they saw, obviously.

Hsu: But that wasn't the bake-off that occurred, like with BeOS in the other room?

Serlet: That was part of the technical evaluation. Yeah, that happened in early December, I think.

Hsu: So by that point, the OpenStep part of the company was focusing on the Windows product?

Serlet: Yes, yes.

Hsu: Because OpenStep itself included the Mach-kernel, but OpenStep for Windows was only the AppKit?

Serlet: Essentially, it was Foundation and the AppKit.

Hsu: Foundation and the AppKit.

Serlet: And I think we had some form of EOF as well.

Hsu: OK, also part of EOF, OK. Talk about the transition to Apple. What was that like?

Serlet: Oh, that was exciting times. It was, there were many new things for me. Many things to learn. But on January 2nd, or whenever the first workday is that year of '97, I showed up in Cupertino and said, where's my office? And nobody had an office for me, but they scrambled and gathered one and I never went back to the NeXT building from that point on, except I think once to visit something. But I'm someone who moves forward, not backwards, right? So it was all about, okay, let's get this new OS that was so much in need to replace the old Mac, right? And let's make it work.

Hsu: You mentioned how excited you were when you found out the news about the acquisition. So what was the source of that excitement? Why was it so…?

Serlet: Well, I had been working at NeXT since 1989, right? So it was eight years and so when you work as an engineer on a project, you want to have it see the light of day and I thought there was a chance. I didn't think it was 100%. I thought it was less than 50-50, but there was a chance [that] what we had done at NeXT could be used for Apple. It turned out it got used. [Laughs]

Hsu: Right! How did your job change when you were at Apple?

Serlet: Oh, there were lots of people to get to know. I had a much bigger team. It was a major step up. So I really had to work hard. The first couple of months were eerie because there were so many things. Learning the technology, learning the team that I had, learning all the other folks I needed to work with, learning the processes. It went on and on and on top of all that, it was massive layoffs. So we had training sessions. I think Apple laid off close to 25% of its people in the first quarter. So I had a lot of manager training for layoff training and stuff like that, so it was eerie. It was kind of like it was an infection point in many ways.

Hsu: I want to go back to the thing that you mentioned earlier about the Apple research team. So this is the Advanced Technology Group that you were evaluating?

Serlet: Yes. So Avie told me, since you have some background in research, go evaluate what the Advanced Technology [Group] research that Apple did and so I went project after project. We did a review of what they are doing, and I was really looking for two things. Can the project in this new context of this new OS, that was the main focus of the company, would it help, or would it be a distraction? And the other thing is, are there some folks that are great folks, we want to bring in the new OS? And regrettably, many projects were not relevant. They were either academic or going on a path that was not aligned with the new projects. So pretty much, we kept a few people who were great out of that team, but we let go most of it.

Hsu: So basically, the whole division got axed. Right. I mean sort of along those lines were there a lot of culture clashes with the existing staff?

Serlet: There were, because I think the Apple culture, after Steve [Jobs] left, had decayed, and the decay had accelerated over the years. So there was also several attempts at doing a new operating system. Maxwell, Taligent, there were like three or four and so people were cynical. So I established a rule that cynicism is not okay and we let go of a few people who were not aligned, mostly left by themselves, but out of that, we got a team that was great. Because there were lots and lots of really great engineers who had been working on the wrong thing, and so I think we recovered the culture and the best thing you can do for a culture is have success. We didn't get success right away. But a few years later, we got success, and that really made the culture even better.

Hsu: Talk about sort of the initial beginnings of the OS X strategy. Started out with Rhapsody and Yellow Box.

Serlet: Yeah, yeah. So the initial idea is to take the NeXTSTEP, make it look like the Mac and that was Rhapsody, and there was one major issue there, which is we needed to get the applications, notably from Microsoft and from Adobe, and so those applications needed to be written for Cocoa, what became Cocoa, Yellow Box, and the developers said, no, no, we don't want to do that. So that was awkward, okay and when Steve came back around June that year, one of the first things he told me is, this is not going to work. This Rhapsody strategy, you need to find a bridge for the applications and so the summer was spent finding a bridge and by the end of the year, we had the bridge, and that became Carbon and that--developers liked the Carbon strategy and that made the Mac OS X operating system possible.

Hsu: And Scott Forstall played a key role in that?

Serlet: Totally, totally. Oh, yes, yes. Scott was the lead for Carbon and also, Scott has a knack for UI. So also the lead for UI. So along the same time, just towards the end of '97, Steve started saying I think we--that was probably more '98. You have a good strategy. But you need a new UI. You can't have this new strategy, this new OS, have the same old tired UIs that the Mac has had and so, do something innovative for UI, and Scott and others, notably Bas Ording, who is the main UI guy, essentially developed Aqua and the Aqua UI was very successful. It was very fresh, very new, and that carried Mac OS X out to be productized.

Hsu: So Avie was the Executive Vice President of Software and you were in charge of OS X Platform Technology, how were your roles sort of delineated in the sense? What was your part of responsibility? What was your working relationship with Avie, and with Scott under you?

Serlet: Yeah, so Avie was another mentor, of course and Avie is also a friend and Avie is brilliant and really helped me steer the project. Left and right, right? And pretty much at every step. So that was fantastic, to have Avie help, and also Avie was interfacing with Steve. So there were some hard decisions to make. Unix was still controversial around that time for the first few years. After Mac OS X was out, Unix became dull. But before it was out, there was still a lot of people saying, oh, no, no, we have our better networking or whatever. While in fact, Unix had much better networking. So Avie was protecting me from interference and pushing the big things. I was working great with Scott. We were partners on many, many occasions, notably the developers conference and we worked well together.

Hsu: Speaking of the controversies, when I first joined Apple, I was in the OS 9 division for the first six months, and there was a lot of griping coming from that side about why are they getting rid of… there was all these arguments about like the CFM format versus Mach-O executable format, or like not having resource forks in the Unix file system, or all these sorts of things that like the old Mac stuff supposedly did better. There was a lot of like that stuff going on and then, of course, when I joined the AppKit team, it was like completely switched over. So it was really interesting to see the different cultures in those two groups, like within such a short period.

Serlet: Yeah, and Avie didn't steer away from taking decisions that were not popular. He always took the decisions that he thought was the right ones, and pushed it very forcefully with everyone, and I was on the enforcing those decisions, so that was a great setup for me, because I didn't have to deal with the controversies much, to a certain extent. The other thing that Avie did that was phenomenal is that he kept Mac OS 9 going, okay, Mac OS 8, and then 9 going and I think that was by talking frankly to the team. This team is paying the bills, okay, while the other team is developing the next-gen OS. Please continue to pay the bill and work hard on 8 and 9 and that message, which is straightforward, I think worked well. There were some tensions, of course.

Hsu: Along similar lines, eventually, for quite a while, Carbon and Cocoa frameworks are both first-class frameworks in OS X. But eventually, by the time Snow Leopard rolls around, a decision is made that Carbon is no longer going forward into 64-bit. Talk about when is the right time to maybe sunset a legacy frame--, or make a framework into a legacy? When is the right time to deprecate? What is the balance between innovation versus compatibility?

Serlet: Right, right. It's super important for innovation to shed the baggage of history and software always accumulates a lot of baggage, right? And so you have to make actually a big effort to shed it. It's easier to keep having the old stuff going. But it also has a cost. You can do less innovation. So you have to make the hard decisions of letting go of the tail that you have. And you can't be too aggressive, okay, because you lose customers, you lose developers. So you have to dose it and I think this is something that Apple has done very consistently and very well for many, many years and still does very well today. To keep moving that window of recent stuff forward and shedding the legacy. So while Carbon was essential in '98 and stayed essential for a few years, there was a point at which we could finally start to give hints that, okay, it's going to get deprecated at some point.

Hsu: Why was that important time the right time?

Serlet: I think it's a balance. You talk to the top developers. You give them a heads up before you send the public message. You check the temperature and, for example, Adobe is a very important developer for Apple and they started using Cocoa for Lightroom, for example, at some point, and so we knew that there was no more any objection against the Cocoa toolbox and we can start shedding Carbon.

Hsu: At some point, Avie is no longer the sort of your direct manager.

Serlet: Oh, yeah. That was a shock to me. I was called to Steve's office and Steve said, Avie wants to do something else. You need to take over his job. And I instantly I thought, oh, I'm going to no longer have all this nice protection that I had from Steve. So one of my first reactions was, "are you sure you don't want Scott [Forstall] in that position?" [Laughs] And Steve said, no, no, no, no, I want you. So I told Steve, okay. Steve asked, do you think you can do it? And I said, sure, sure. Because it was not about the ability to do it. It was more of the consequences. So that's how I took Avie's old job.

Hsu: Talk about working directly with Steve now.

Serlet: Well a few days later, I got the first phone call in my new position. That was, I forgot what, some great idea of Steve or that and I realized that there was nothing I could do to prepare for those phone calls. I just had to be me, and so that's what I decided very quickly. There was no one else [who] I could say, "Avie, please help me." So it just was me, and that worked out.

Hsu: I guess that other than not having Avie as a buffer, how else did your job change?

Serlet: Not very much, because by that time, Mac OS 9 was on its final kind of motions. So I think that was around 2003. So Mac OS X was in its third release, I think. So Mac OS X was starting to be established. The company at that time was still losing as many customers as gaining new customers. So it was a wash. We were still hovering around 3% market share. But we could see the path, right? Because we were shedding the old customers and getting some new customers. So that was a good trade.

Hsu: Let's talk a little bit about programming languages. So in my interview with Steve Naroff, he talked about how you had different ideas about what Apple's programming language focus should be on, that shifted over time.

Serlet: Yeah, I thought that Objective-C was a handicap and Objective-C was--had good sides, technically. But nobody knew Objective-C outside of the NeXT community. So in '97, the only folks knowing Objective-C were the NeXT community folks. NeXT and then the developers of NeXT, right? A very small group and this was somewhat confirmed when Adobe said in '97, we don't want to be part of your Rhapsody effort. So I wanted to escape Objective-C, so I pushed towards Java. Java was growing very, very, very fast in those days. I even met James Gosling to try to see if Java could do a few things that would make the Mac better. James Gosling was absolutely not interested, and so we tried Java, but the problem was that Java was huge. It had a huge footprint. We had the same calculators, you know the little calculators that you find on a Mac done in Objective-C and in Java and Java was 10 times larger in resources and so ultimately, Java was not viable. So then moving forward to the iPhone, the iPhone 1.0 was implemented in Objective-C, but was not open to developers and the iPhone 2.0 was open to developers and there was a bit of a debate with Steve and Scott about, should we open it at the level of web apps, or should we open it at the level of Objective-C, Cocoa? And I was arguing that Objective-C would be a handicap, that very few developers will start writing iPhone applications in Objective-C. I was totally wrong, of course. Steve overrode my objection and said, no, no, no, no, it will work, and he was right and so we went out with Objective-C and at that point, Objective-C was in wide adoption by all Mac OS [X], but more importantly, all iPhone developers. So by the late 2010. But by that time, Objective-C was kind of old, had not changed much. The only thing of significance was the thing that Blaine Garst had done, which is the autorelease, automatic autorelease, so that you can get rid of allocation and free minutia and so we had Chris Lattner, who had done LLVM that we had hired, and he was toying with a new language and I strongly encouraged him, because I thought Objective-C was too old, and we can do much better with modern technologies and then I left. But Chris kept working at it, and that became Swift.

Hsu: So Swift started initially when you were-- under your watch.

Serlet: Yes, yes, yes and it was a pet project of Chris.

Hsu: Yeah, but you encouraged him.

Serlet: Vague encouragement. Because I thought Objective-C was-- the square brackets was just a little passé.

Hsu: It hadn't changed much, but there were small changes, right? And there was an attempt to change Objective-C early on to a more Java-like syntax.

Serlet: Yes, that didn’t <inaudible>.

Hsu: Yeah, right. But then there was also later on Objective-C 2.0 with garbage collection and properties.

Serlet: Yeah, yeah. There were some improvements, but in the end, I don't think they made a big difference. Yeah, garbage collection was a pet thing of mine, because I had been raised with a garbage collector. Even when I was programming in the late '70s, I had a garbage collector. So I thought cleaning up by hand when you have a computer is kind of a waste. But the problem with garbage collection is the performance of it and it has a huge performance cost and so the breakthrough was actually this automatic auto-release that brings most of the benefit of garbage collection, but without the cost, and in essence, that's smart pointers in C++, and that's how Swift all works.

Hsu: Right. I remember when automatic reference counting was announced, and it was huge.

Serlet: Yes, and it made the garbage collector obsolete instantly.

Hsu: Talk about, there was a period focused on C and C-based frameworks as well.

Serlet: Yes, that is around ’98 and the reason is that we had decided to do Carbon. That was the old toolbox written in C and C++ and we had Cocoa, that was in Objective-C and I thought it was important that we start sharing those two stacks by zipping them at the bottom. So I pushed and we developed this framework, all in C, that's called CoreFoundation, that took all the Foundation knowledge, but instead of doing it in Objective-C, doing it in C, and that was underneath both Carbon and Cocoa and that zipping of the stack enabled us to make the user experience very smooth, because it enabled us to zip the UI as well, so that, for example, we had shared menus for the applications, so most users would not know that they had a Carbon or a Cocoa app and that was the goal.

Hsu: How do you think Swift will shape Apple's technology future?

Serlet: Oh, it's a great language. It's a modern language. It has all the right fundamentals. Notably, it knows how to do types correctly, and that's a fundamental thing. So I think it's a major asset for Apple moving forward. Now, it's still very much limited to Apple. I think the rest of the world is moving towards Python, of course, for AI, and Rust, as a low level, kind of fast, compiled language and then Chris Lattner is now doing something called Mojo that has a lot of headroom.

Hsu: Apple now has SwiftUI as a Swift native framework. At what point do you see AppKit going away and being replaced by SwiftUI? How long will that take?

Serlet: So you have to realize that I haven't been at Apple for a dozen years now. So this is all my view from the outside. But I can't wait. I think SwiftUI is remarkable. It's elegant. It's compact, and so I hope that all the old Objective-C, Cocoa stuff will be recasted in native Swift framework as soon as possible.

Hsu: So you were in charge from 10.0 Cheetah all the way up through, was it 10.7 Lion?

Serlet: Yes.

Hsu: And how much changed versus how much stayed the same over that period?

Serlet: I think a big difference from the early days to the later days is success. At first, it was a little dicey. The first version of Mac OS X was really to get the pump started. It was unusable. It was too slow and we knew it, it wasn't like we needed feedback. We knew it was too slow. So I think we priced it, which was a pretty high price, because we didn't want people to really use it. We wanted just a few of the key developers. We wanted to have them. So what we did after 10.0, that was in March 2001, I think the release.

Hsu: I still have the T-shirt.

Serlet: Oh, yes. We very quickly tried to fix the number one problem, which was performance, and so we actually did a release in six months, just six months, which was amazing. But we focused only on performance. We said no features, just performance. So 10.1 was usable by developers because it was now performing reasonably well. It was still very feature poor compared to, for example, what Windows had, at the time, and so 10.2 added a number of features, and 10.3 added more features and somewhere on 10.2 to 10.3, we got to parity with Windows, notably. But of course, we had a lot more legroom, so we could move faster afterwards.

Hsu: And then Tiger [10.4] shipped, which was the last version with the Classic Blue Box compatibility.

Serlet: Yes, yes, and that was, again, this moving window of obsoleting old technologies, to keep a little of pressure, a bit to dose the pressure, so that people move off the old stuff to the new stuff, both end users and developers.

Hsu: It's almost like you're dialing up the temperature slowly.

Serlet: It was, it was and it was very conscious, and we did a number of things to help move developers, notably, forward. For example, applications had all kinds of reliance on weird features in the toolboxes, in the old Macintosh Toolbox. So in Carbon, we had to preserve some sort of source compatibility for those applications. So we had little hacks to preserve the old behavior with Carbon. But we predicated those hacks to a certain version of the app, so that when the app gets revved, they have to clean up the code. It was no secret. We told the app developers, this is the last version that will benefit from this little workaround built in. Now you need to clean up your code. So we would not only move our window forward, but we would really push developers to clean up so that we can have simpler code and more applicable, more regular and there was constant discussion with Adobe and Microsoft at every release to help them move forward.

Hsu: I also remember the frameworks, Carbon and Cocoa, being more or less feature parity, but there would always be one thing that maybe Carbon did that Cocoa didn't do, or one thing that Cocoa did that Carbon didn't do. How difficult was it to keep both frameworks on par?

Serlet: Yeah, we wanted to be on par, at least in the early days, but there were always the schedules, the people availability, the preferences. But the parity was a general mindset.

Hsu: Talk about the Intel transition.

Serlet: Yes. Yeah, that was-- it took a few years to get there. We had several kind of points in time where we thought the PowerPC was not moving fast enough from a technology standpoint, and we should move to Intel. One thing that Avie pushed, and I enforced it, is that we always compiled our code for Intel as well as PowerPC, even though we had no Intel machine and so we had some checkers in place in the build system to make sure every single project can be built for Intel and this is years before we did the transition. So at some point, it became clear that we were going to transition. But that was depending on the deal and Paul Otellini, who was the CEO of Intel, and Steve inked that deal some February. I forgot the exact year, must be 2005, and so it was a very straightforward deal that covered both the hardware side of things, of getting computers and getting help from Intel and we just scrambled to make it happen for the developers conference that was in May, so three months later and one thing that we wanted to do for the conference is unveil it, announce it, which was very material, because at that time the Mac was still a big chunk of Apple's revenue, and so that transition had to be done very smoothly. We wanted also developers to have machines that they can play. But we didn't want the Mac OS to run on any PC. So there was a delicate balance. So we decided to build machines in secrecy and in fact, we enlisted Simon Patience's team, which was the CoreOS team, to actually build the machine that we were going to give for developers at the conference. So for a while, the kernel team was actually building, assembling machines in a secret lab in preparation for WWDC and it was the kernel team, because they were the first one to help make it work. So they all were disclosed. So we had initially in February, when the deal was signed, there were like a dozen people in the know and it grew to several thousand people, all in secrecy. Nobody spilled the beans before the conference and I believe that the conference started at some time, like 10:00 a.m., and at 9:00 a.m., all the people attending the booths were told an hour before [laughs] they were supposed to help developers transition. So it was fun. The secrecy was actually a lot of fun, and it was a lot of hard work to make everything work very smoothly. We had, with Scott [Forstall]. Scott was, as always, my partner in crime for the presentation at WWDC. So we had a whole presentation about how to change the engine while keeping the car cruising and we said at that time that it would take about a year to transition and nobody believed it. All the industry thought that it would take much longer. We were hoping it would take less. But we were not sure, because we had a dependency on Intel for some of the new chips coming up. So we were not sure, and still we're not sure, for several months. So I said, well, we're going to pretend we need to ship ASAP and we don't have a schedule, but we just have a punch list of what's left to do and we're going to shrink the punch list and that's what we did. People were upset because they wanted to know the schedule and I said, sorry, I can't tell you the schedule, I don't know the schedule, but it's ASAP, and so we were able to ship in January with the new Intel Macs and so that was six months, not a year and then the rest of the transition was done in less than a year, the rest of the transition of the machines. So it was a very smooth transition. But that summer, I had two worries. One was that now that we were on PCs, people could compare the speed of the Mac on what was essentially an Apple machine, but a PC, so same processor, with Windows NT and so I was very worried by that because I had used Windows NT from time to time. It was pretty fast. So I said, let's do benchmarking of lots of things that analysts could benchmark or end users could benchmark. For example, you open 100 Windows, how long does it take, right? That kind of thing and we wrote several hundred tests that we could run both on NT and on the Mac and the results came in, and for 95% of them, the Mac was slower. So we had a big communications meeting. I motivated the team by saying, hey, not so good, and by the end of summer, we had flipped that 95% and so we never had any issue on performance after that and so that was good. Because once you have a very clear target, which is those benchmarks, you optimize for those benchmarks. But then, as you optimize for those benchmarks, a lot of other things become also faster, because you optimize all the underpinnings and so by the end, we were in really good shape. So that was my worry number one. My worry number two was that, so you have now the following year, you have some Macs that are Intel and some that are PowerPC and people will find some subtle differences between them, because we've still evolved the Intel code base, and so there would be some fewer bugs and more bugs, different bugs, right? And that people would play the game of find the difference. So, which I think would be very damaging for moving forward. I wanted the transition to be viewed as invisible. So what we did for the rest of that year is we did software updates that had all the improvements that we made to the Intel side. But that also were all those improvements that were not for PowerPC, but we also included that in the PowerPC update and so we had a single code base. We forced the code base to be the same through software update. So by the time January came in with the new Intel machines, they were exactly the same software as the PowerPC. So no one ever found some bug difference between the two. So I think that made the whole transition very smooth and very invisible from an end user perspective and also for developers.

Hsu: Given that NeXTSTEP had already run on Intel you were starting from a baseline already.

Serlet: Yes.

Hsu: It wasn't like you were starting from scratch.

Serlet: Oh, yes. No, we had many of the frameworks were ready and we had a little tiger team, a hidden team, keep kind of making frameworks work, repair the endian issue and stuff like that, pretty much all along, especially once we committed to doing it.

Hsu: Were you worried about, I mean, ultimately people did find ways to create Hackintoshes, right, to run OS X on regular Intel PCs. But was that something you were really concerned about?

Serlet: Yeah, we had some concern by that because we wanted to keep control of where we are running, to do this deep integration between hardware and software. So we put some protections to tie together the particular Mac hardware with the software. All of that was defeated very quickly. But it still offered a certain barrier. So in the end, it wasn't a big problem. We knew from the statistics on software updates how many of those PCs were not-- that existed out there, that were not Macs, and it was a few percent. It was not significant. So after a few years, we stopped worrying about that.

Hsu: Talk about the 64-bit transition, and how that compares to Microsoft's strategy for their 64-bit transition.

Serlet: Yeah, I can't talk to the Microsoft strategy. I forgot what it was at that time. For us, it was part of this moving window, we wanted to move forward and we wanted to move forward as soon as possible. But for 64-bit, we had to move developers first before moving, of course, end users, so we needed apps that were ready. A key element of all this was the multiple architecture binaries. So an application binary could have different architectures inside it, and when you launch your binary, the computer you're in picks the right one and that was great for developers, great for deployment. You didn't have to think. You can have a single binary that has all those things. So we pushed to switch to 64-bit as fast as possible, move that window as aggressively as possible, so that we can have computers that are 64-bit, because we knew that would be a boon for end users in terms of speed.

Hsu: But you carefully staged that transition, right?

Serlet: Oh, yes, yes. All those transitions were carefully orchestrated, debated, thought through, and executed.

Hsu: So like kernel first, and then one year, and then frameworks the next year.

Serlet: Yeah, yeah.

Hsu: And sort of the culmination of this is Snow Leopard, which was marketed as a no-user feature release, but technologically was a pretty big inflection point.

Serlet: Yes, but we had gotten some feedback, both from ourselves and from outside, that things were starting to be a little buggy here and there, and so I forgot how it came up, but we talked about focusing on quality and Steve supported this notion of having a release focused on quality. So at some point, the talk with Phil Schiller became no feature, and I thought that was bold, but Steve and Phil were in agreement with it. So I said, let's do it, no features. Now, you still want to have some technology innovation, so we put a number of technology innovations, notably the Grand Central Dispatch. That was pretty deep, and that had the end user repercussion in terms of performance and responsiveness and we had to do Exchange support, too. So that was a feature, and I think at WWDC, we unveiled the release and said, no feature except Exchange support.

Hsu: So that Grand Central Dispatch came out and Objective-C blocks in C, and Objective-C went along with that. That was also the last version to support Carbon, basically. The end of the 64-bit transition. Yeah, so many different milestones. It's also the last release on physical disc. Interesting enough.

Serlet: Yeah, the iPod, of course, became very successful in the early 2000s, and quickly, the iPod switched to SSD instead of hard disk and anyone at Apple could see how well SSD worked. There were some worries that people had initially about the wear cycle between the iPod and the hard disk, and that it would stop working after a year, right? But no, none of that materialized. But the big win is, of course, the speed, the speed of an access to information, and so it became very clear with the iPod's success with SSDs that we wanted to switch our laptops, at least, to SSD ASAP. Everyone was in sync with that, so we made it happen, essentially.

Hsu: What I meant was Snow Leopard was the last version to ship on a CD, on a DVD, rather than over the internet.

Serlet: Yeah, yeah. Well, there were also this moving window, right, for the hardware technologies, right, and the distribution. By 2000-, what was it, 2010? 2009, maybe?

Hsu: Something around that, yeah.

Serlet: Yeah, everyone had the internet, most of our customers.

Hsu: During this whole period, the OS X leadership, yourself included, but also having Ali [Ozer] there the entire time, still there, remarkably stable. What were some of the benefits and drawbacks of that sort of stable leadership?

Serlet: Well, I'd start with the drawback, is that after a while, you always take the same thought paths, and I was afraid that there was no longer enough innovation and that's why I brought in Craig Federighi, who I knew from my past at NeXT, was full of ideas, and to help bring new ideas. So that's the drawback. On the other hand, you work really well with people you've known for a long time, and that you trust them, and it's a pleasure, so it was really comfortable working with Simon Patience on CoreOS, with Peter Graffagnino for graphics, Scott [Forstall], just a great team.

Hsu: I want to talk a little bit more about Scott. So when he was working on the iPhone, he was still managing the Platform Experience team in OS X, is that correct?

Serlet: Yeah, so you probably have heard the story of the iPhone, right? But everyone probably has their own slightly different version. But for me, it started with the prototype of the iPad. The iPad came first, what is now called the iPad. It was a prototype. It was a multi-touch screen where you can do pinch and zoom and so it was tethered, it wasn't working off battery, it was a prototype, so there's still a wire attached to this. But the main thing was just a screen, just like the original iPad, and multi-touch, and Bas Ording got hold of this prototype and developed a photo app that had pinch and zoom and moves the photos and in one of the weekly UI meetings, Bas presented to Steve the demo of the photo app. Steve played with it for a few minutes and he said, you guys, this is amazing. Everyone will want to deal with the photos that way. But it's going to be a niche product if you just have that. So what I want you guys to do is to stop playing with the photo app. I want you to do a keyboard on glass and this is one of the brilliant moments of Steve, this vision, right? That the key to evolve what was going to be the iPad to a real product was the keyboard. Which was kind of, you don't expect that from Steve, right, to focus, so Steve you think is more UI-oriented, but he pointed us to the keyboard and so Scott and his team, for about a year, tried a hundred different keyboards and there was user testing on how to type on glass, I mean, remember in those days you had the Nokia buttons, right, or BlackBerry buttons. You didn't have glass, right? And every week we would review the latest keyboard and the scores and all that and finally, there was this one moment in this UI meeting where there was a keyboard and looking like the original iPhone keyboard, and it worked. It really worked. With the predictions, there was minimal prediction in those days, but some prediction. It actually just worked. So Steve, and that was another moment of genius, says, okay, this is working, okay? Now we can have this form factor, the iPad form factor, with this keyboard. Now what I want you to do is to shrink it, to see how small you can get a keyboard that still works well, okay, but that's smaller and that actually took no time, because that was just a shrink, right? So that took a week, and the following week there was a small keyboard that is exactly the size of the original iPhone keyboard, and Steve said, okay, guys, let's start the phone project now and so he said, Bertrand, I want you to take Scott and to give Scott your top 100 guys who work on Mac OS X. I want Scott and these 100 people to form the iPhone team, which I did, and then about a year later, we shipped the iPhone.

Hsu: Wow. So this all took, so this is only a year between that point and the actual shipping?

Serlet: Yeah, so I think the actual shipping happened, I think, in 2007, in June, and it was announced in January. We had prototypes in December, on December 19. That's my birthday. So that's how I remember. So 2006, and so I think this, Steve said, let's get Scott and 100 top people to work on the iPhone was a year prior to that, roughly.

Hsu: The hardware side and the software side, those stories are like--

Serlet: Oh they're intertwined. Yes.

Hsu: --so there's like separate, but like intertwined and like, it's so confusing to get all these timelines straight.

Serlet: Yes, yes. So the hardware was progressing on that. There was also a question of which operating system would run on the iPhone and there was a skunk project to run Linux on the iPhone. So when that happened, I started another skunk project to run Mac OS X and I think it was in June of probably 2006, we had it running on the-- or maybe 2005. We had Mac OS X shrunk, okay, just removing subsystems that were not needed, run on an iPhone prototype, early prototype.

Hsu: What do you think tipped the balance in favor of OS X?

Serlet: Oh, Avie was, of course, pushing very much for OS X. Steve didn't like the lack of control with Linux and wanted Mac OS X too, so it was a no brainer, really. It would have been very dramatic for the company to have two OSes. You think of the company the way it is now, where it has essentially the same OS on all those platforms. There's tremendous benefits from that, and we couldn't foresee all those platforms, but we could see that the iPhone was going to be huge and that we don't want to have two OSes. So it was kind of a no brainer decision, but what it took is to prove that it can work.

Hsu: At what point did Scott sort of no longer work under you?

Serlet: Oh, I think that was around that time. That was when we started the iPhone project, but we had interdependencies like this, and we always had worked well together. We had worked for so long together that everything was much smoother between Scott and I than with Scott and anyone else.

Hsu: We won't get into that.

Serlet: Yeah. Scott he's very driven and sometimes people were upset by how driven he was. But I like the being driven part. Scott made really the iPhone succeed, all aspects of the iPhone. He was every day, every morning. He was looking at what happens in the iPhone ecosystem and what can we learn from that. That's how Siri came up. He discovered that app in the ecosystem and said to Steve, we should acquire that.

Hsu: You mentioned earlier that decision to finally open up the iPhone SDK, because the word had been web apps only. How did that decision sort of come about?

Serlet: Oh, that was actually very straightforward. For the iPhone 1.0, we had a dozen apps, basic apps and there was no time to develop, to open up, to open an SDK. So for 1.0, everyone was in sync. Scott, Steve, myself, that this is the only choice. Now, Steve kind of liked the fact that all the apps were Apple apps. So in a different history path, maybe Apple should have kept it closed. So Steve was actually arguing for a while that maybe we should keep it closed. But there was a lot of pressure to open up. Now, a few years prior to that, we had on the Mac, a feature called dashboard widgets. That was very, very small, lightweight apps and this is now several years prior. It's 2004, something like that, and this is a time where it's difficult still to get Mac apps. We have to argue with Adobe and Microsoft to get Mac apps and the developer community is reasonable, but it's not expanding as much as we can. We came out with dashboard widgets and within a couple of months, we get tens of thousands of dashboard widgets and so we knew that there was pent-up demand for lightweight applications on the platform. So with that in mind, it was natural to think, when we were now one year into the iPhone, or six months into the iPhone, that we'd be successful if we open it up with an SDK. So we argued how to open up, web app versus Cocoa. We did Cocoa, and we opened up and a year after, we started getting apps and I thought we'd get thousands of apps based on this dashboard widget experience, within a few months. I was totally wrong. We got 100,000 apps. There was a tremendous amount of pent-up demand. So that was the beginning of the iPhone SDK and of course, now there's millions of apps.

Hsu: How did the debut of iPhone and iPad, and basically iOS, change the nature of your job managing OS X?

Serlet: So I was responsible for some components that were on the iPhone, along with responsible for Mac OS [X]. So for example, the whole kernel was delivered to the iPhone and that is the thing. At that point in time, it was more important to get a successful iPhone release than a Mac OS [X] release. Mac OS [X] could be late by a few months and so I diverted consciously the resources towards the iPhone. So that also made it very frictionless with the iPhone team, because we were really helping.

Hsu: So it wasn't like the teams were competing. It's like the whole company was...

Serlet: Yeah. Now, I think some engineers probably were a little upset that their project is not the one that has the limelight, but I think people are adults and you treat people like adults and it works. People understand the context. I think everyone understood that the iPhone was going to be very successful. I did understand that on my birthday, December 19th, because on that day, there were four prototypes available and so I got one prototype for an hour. So I locked my office, lowered the blinds so that nobody could see and started playing and that was the first time we had the software and the hardware together. Those four prototypes, Steve had one, Scott has one, and there were two others flying around to various people and so I played with Mail. I played with Safari and I didn't call because the phone was not working yet. This was December 19th, we were announcing in January, but we still didn't have phone working. But everything else was working, the address book and all that, and I knew it would be very successful. I knew everyone would want one. I could not predict that people would spend several hours back on their phone. So, but yes.

Hsu: The world has changed forever. So you spoke about Craig Federighi. So you mentioned you brought him in. This is even before you decided to leave?

Serlet: I brought him before, as I was starting to think about leaving, yeah. I had been for so long at Apple and NeXT, I just wanted to see the rest of the world, explore a few things. But I just wanted continuity.

Hsu: So when you brought him in initially, what did he work on?

Serlet: So he had to be brought up to speed, right, on things. So we split the responsibility. I told him I'll be in charge of things that are more the software updates side of things, the continued evolution, and you'll be in charge of the next OS and so Lion was his first release.

Hsu: That's definitely a big shift in the cycle. We'll move to sort of what made you decide that this was finally time for you to leave Apple?

Serlet: Well, there's several factors, but one of them is too much comfort. I like to push myself and so it was getting too comfortable. I worked well with all the folks in my staff. It was very smooth, and I felt I needed to renew myself and wanted to go back into the startup land. NeXT was a startup, but kind of not startup.

Hsu: Startup with the benefit of deep pockets. You had already been involved with Grokable since 2005?

Serlet: Oh, yes, yes, yes. That's AI, yes. Some brilliant researcher approached me at WWDC one year with something that was doing essentially tracking your face, and so I kept working with him and I work with him still on this technology and it's a little bit of a research thing, but it got me involved with AI 15 years ago before the winter ended.

Hsu: Talk about starting Upthere.

Serlet: So I saw an opportunity to do something in the cloud around storage and I had some technical ideas based on immutability. I didn't have much business ideas, which ultimately is a problem for a startup. So we created neat technology for storage, but you really need to have business ideas for a startup. So ultimately, the company was acquired by WD [Western Digital], but really has not seen the light of day. So that's too bad, but there's always lessons that you learn in the path.

Hsu: What was it about the idea that you thought was viable?

Serlet: Well, the idea is very simple. It's that storage keeps growing at an exponential rate, actually, and you have a lot more storage that is in the past than you have in the present and so a lot of it is immutable. So the idea was to take advantage of immutability to store storage efficiently and this is the immutable theme, again, coming back from my early days at Xerox, and I think it worked well from a technology standpoint. But with big companies like Google, Microsoft, Apple, Amazon, offering storage, it didn't make business sense.

Hsu: You hired a decent number of former Apple people.

Serlet: A few, yeah.

Hsu: So it was kind of some continuity there.

Serlet: Some, but not at first. For the first year or two, there was no one from Apple and it was actually conscious, because I wanted to be outside of the Apple world. That was probably a mistake, actually, because I think one element of success that's essential in a startup is the initial folks, the founding team in large and getting people that you know and trust and have worked with is probably a key element of success.

Hsu: What other lessons did you learn from that experience?

Serlet: Well, another thing that we did at Upthere was we hired a lot of folks out of school. Because we liked, my co-founder and I, we liked the energy, and we thought that was a great way to do things. Well, not so much. When you get a bunch of people out of school, some work out and others don't. Independent of the technical abilities. Some have a new boyfriend or girlfriend and suddenly they disappear from work for the next month or so and you can't really run things that way. So there was another lesson learned here, is that you want a balanced team and I think initially for the initial group, you probably want to have quite a few senior people to go in the right direction.

Hsu: So Upthere was acquired in 2014, is that correct?

Serlet: Yeah, I think so. No, 15 maybe. Not sure.

Hsu: So then you decided to start another company in 2015?

Serlet: Yes. I reconnected with Pradeep Sindhu, from my days at Xerox PARC. Pradeep, who had enjoyed success funding Juniper Networks, and who had some ideas for how to use silicon to create a lot more efficient data centers and that was connecting with some of my foray in storage at Upthere. So we incorporated, we started discussing in 2014, we incorporated in 2015, and we didn't make the same errors that I made at Upthere, so we started by recruiting key senior folks that are still with Fungible for most of them to this date. I mean, Fungible has been acquired, but they are still in the same team.

Hsu: So the focus is data processing, hardware, this is for data centers?

Serlet: Yes. So if you try to visualize all the computations done in data centers, it's difficult. First, because there's a lot of variety, right? You have a lot of vanilla stuff, and you have scientific stuff, and lots of business, AI now. But you look at what are the computations, really, that are made, and there's a whole class that's very general computation, like your Excel spreadsheet. There's a whole class that's math-oriented. That's AI, graphics, it involves linear algebra, it involves math. But there's another, let's say third, okay, which is just moving bits around, and maybe transforming the bits along the way a little bit, but mostly organizing moving bits. So all of networking falls in that category, all of storage falls in that category. You say, I want to read this file, and there's a lot of activity along the way, all the way to the hard disk or the SSD where it's stored. All this activity is very vanilla. It's processing flows of information and so we developed the DPU to very efficiently process flows of information. It's about an order of magnitude more efficient than your CPU at doing that, and each flow is very different. So it's not like a GPU. In a GPU, everything is the same, right? The computation is the same across the entire array. No, with the DPU, everything progresses differently, and so we have a radically new architecture for that, and that's what we did at Fungible.

Hsu: Is there any AI or machine learning involved?

Serlet: Not really, no, no. We do have a few things that smell a little bit like AI, but nothing like the latter-day AI, which is LLMs.

Hsu: That more or less brings us up to the present with the acquisition of Fungible by Microsoft, which we won't discuss. I don't think we have time to get into the major themes, but talk about what in technology are you most excited about now?

Serlet: Well, today, it's about AI. It's about LLMs. Now, I've been doing, on the side, a little bit of AI for the last 15 years and I was always thinking in the future when they will get AIs that's general, right? But it was always in the future and in the last year, it's in the present now, and it's a discontinuity. I think it is more significant than the iPhone was, more significant than the internet was, because now we have machines that are kind of like us. They can think, they can do logic. When you have an LLM, it has a snapshot of all the web knowledge, okay? So that's knowledge. But it has also abstracted patterns and it has all this collection of patterns and patterns of patterns. That is, I think, a fundamental piece of intelligence. So I think that's fascinating. The field is moving very fast and I can't wait for the next two or three years.

Hsu: What do you think about some of the current problems with AI and LLMs in particular? And how do you think we should approach solving these problems?

Serlet: Well, so you have a problem, which is hallucinations, right? So yes, generative AI tends to say things that are not the truth. I don't think that's a big problem. I think this will get better and better significantly. Already, you see the delta between GPT-3.5 and GPT-4 and it's far fewer hallucinations. So I think that's a technical problem that can be solved. But we've always dealt with misinformation, okay? Every human society has some level of misinformation. So I don't think this is a big issue. Another problem is ethics and the big US companies have guardrails in place, and so I think this is addressed by the big companies that have big LLMs and I don't think the government should do much about this. It should let the commercial space deal with that. Because the big company wants to avoid controversy, right? So the big companies, the big US company, will do all the right stuff to avoid controversy, so I think we don't need to regulate that. Then you have the next level of worry, which is it's going to take away some jobs, okay? That's real. I think a significant fraction of jobs are going to disappear in the next few years due to the impact of AI. Now, it's going to be progressive. It's going to be first the West, then it will reach the rest of the world. But we'll adapt, and in fact, I think it will let many people do higher quality work, okay? By pushing up, okay? Using AI as a tool. So I'm an optimistic. I think actually this will be beneficial, but will provoke some change. So there will be the studios that have people on strike, and there will be the studios that use AI as a tool. I think the latter ones are probably going to do movies that you watch rather than the previous ones, right? So I think this is all good and bad. So moving up the risk, we have now bad actors, okay? There can be a Russian LLM, there can be a Wagner LLM, right? And they won't have the same guardrails, right? That's bad, that can be very dangerous, because there's a lot of power with computers that is really small. But I think we'll be able to deal with that. It's going to be cat and mouse, right? We'll have to ratchet up, and that's why we cannot stop doing what we're doing. We need to keep accelerating to make sure we keep ahead of those threats. Now the last piece of the ladder, okay, of the risk ladder is sentiency, okay? I believe it's just a small number of years, two or three, before we have AIs that are sentient. Because I don't believe that there is any particular reason why biological constructions are more sentient than artificial ones. So we'll have AIs that are sentient. I'm hoping that with the right training, especially the training that we do in the West, the AIs will be benevolent. That's not a given, but I think it's a natural, logical conclusion and so we want to make sure that AIs continue to be benevolent, and so I think we need to register all those sentient AIs, and make sure they form a democracy that keeps the malevolent AI out. So I think that's one of the biggest risks. But I'm an optimistic, and so I think we'll push things in the right direction.

Hsu: Interesting. But how do we know that the AIs will be benevolent?

Serlet: Well, yes. It's like the Turing test, right?

Hsu: I mean, they're already causing problems now. Even without being sentient, right?

Serlet: We need to develop tests for sentiency and for benevolency. Yes, I think that's what we should regulate at the government's level, not the rest. The rest, let's let the free market do its good things.

Hsu: The other issue is, how do we decide what benevolent means? Because different people have different values, and what's benevolent to one person may not be benevolent to another person.

Serlet: Yes. That will bring some new very high-level concerns, and ethics will need to get developed. But I think that's pushing the human race up.

Hsu: So maybe this is a preview for what we can talk about next time. Looking back through your career, up till now, what sorts of things have been constant over your career?

Serlet: So I think, first and foremost, it's people. I've been always blessed by being surrounded by great people. I've had a number of great mentors. Steve, of course, was another mentor and I think you want to always be surrounded by great people, people smarter than you, people you can learn from. So I think that's definitely a theme and I've never shied away from going towards the people who are the best, and I encourage every young engineer to gravitate towards people who are the best. So I think that's a theme, okay? It's people. On the technology front, I fell in love with VLSI, the simplicity of a transistor. A transistor is formed conceptually by two wires, and if this one on top, the gate, has one, then the current flows, if it has zero, the current doesn't flow and from that, you build all the computers that we know today that have billions of transistors. If you look at the big AI system, they have millions of billions of transistors all together. So this is a story of composing big things with little things. You look at AI, it's based on the matrix multiply. Multiply two numbers and add them, okay? That's the basics and from that, with a couple trillion parameters, you actually make something that's intelligent. So this is a story of composition. Now, we still have a ways to go with composition. For example, for AI, we need to compose these LLMs with more classical algorithms so that, for example, they can calculate, multiply two numbers. They're not very good at that right now. So we need to understand how to do this composition, but that's been a theme on how to compose and APIs are very essential in the composition of software. So that's been another theme, which is APIs. I think you cannot do software, I should exclude AI for a second, unless you understand and master APIs. An API is a way to encapsulate knowledge. Something that has thousands of lines of code becomes two functions and now other people can use those two functions to build higher level constructions. So that has been, I would say, a theme. Immutability has been a theme. I think it's still underrated. Some languages, programming languages. I've been fluent in a number of programming languages, probably two dozen, and I enjoy programming, still.

Hsu: I want to go back to the API thing because it brought to mind something. I remember API review at Apple and how careful that was, right? And I think that's something that had come from NeXT, right? Making sure that APIs are very well crafted, very well designed, and that culture of making sure that the APIs are good. Can you talk a little more about enforcing that discipline?

Serlet: So that's what I really learned from Russ at PARC, applied at NeXT. I was surprised that there was no API review when I came in and the NeXTSTEP API could have been better if they had had a review and that was one thing that we did with OpenStep, we started doing reviews, and that's where it came from and it's very delicate because you want the owner of the API to own it. You want owners to own, right? You don't want to disempower owners, but you want to have feedback too, so the API process that we replicated at Apple was very simple. You, when you have a new API, that's public API, that customers will, developers will use. You have to publish it to a certain forum and you have to address all the feedback, one way or another, you can say, oh yeah, I don't care about that. But you have to address the feedback somehow and it's time-bound. It's a week or some limit like that and after that, that's it. You can go ahead. Your API is approved, and so this process did not disempower the people because they were in charge, and we had the forum be a distribution list of all the folks that are API savvy and pretty much the trick there is that anyone who has to be on the list is on the list, and that self-determination really works, because people who are not interested in this will not ask to be on the list, right? Very simply. It's subject to abuse. I think during my tenure at Apple over 10 years, right, 15 years, there were two or three cases of abuse. In those cases, that is the feedback was not addressed or the feedback was this is really a bad API and they went ahead, and you deal one-on-one with the abuse. Very minor. Two or three out of thousands of API reviews. So I'm a big proponent of very light-touch processes, not heavy processes and leaving people ownership.

Hsu: Yeah, because the consequences of a bad API, once it's out, is that it's much harder to get rid of it.

Serlet: Oh, yes. It's very difficult. You can change the code, the implementation, but changing the API, now you need to evolve all the apps that have depended on it. It's very difficult. It takes years, literally. We've planned obsolescence to put the new in place. You tell people, I'm going to start deprecating, then you deprecate it several releases. It's typically two, three releases.

Hsu: What would you tell a young person starting in the industry today? What would your advice be?

Serlet: Well, on the domain, I would say AI or silicon, but there's still things to do in silicon. But everyone has their own preferences, so there's still lots of work to do in algorithm, in operating system, in frameworks, in languages. So I'd say all of it is okay. But always strive to do your best. If at some point you find yourself that you're not doing your best, well, you should move.

Hsu: So what one word of advice would you give to a young person?

Serlet: Okay. So I prepared that and this is my word.

Hsu: All right.

Serlet: Challenge.

Hsu: So explain why this word.

Serlet: You have to challenge yourself always, I think, to grow, to do new things, to learn new things and you have to challenge others too. Not in a rude way, but you have to push people around you, I think that's how we progress as humans.

Hsu: All right. Thank you very much.

END OF THE INTERVIEW