Oral History of Kenneth Kocienda and Richard Williamson, part 2

Interviewed by:
Hansen Hsu
Marc Weber

Recorded November 13, 2017
Mountain View, CA

CHM Reference number: X8367.2018

Hsu: All right, it is November 13th. I am Hansen Hsu with Marc Weber. And we are here with Ken Kocienda and Richard Williamson, again, for part two. So, let's get back to where we were and start where we left off. So, we were talking about sort of working on iPhone 1.0. And one thing that I didn't get to ask is what was sort of the experience like of being on that team and working in secret and the long hours and that general sort of the feeling or the life of working on that project.

Williamson: Do you want to go first?

Kocienda: Sure, I'll go first. It was intense. There was a lot of pressure. It was fun because we had something that, from the very beginning, seemed like it could really be big, important for the company. I don't know [that I] quite thought that it would be important for the world, but important for the world of technology, I think.

Williamson: And important for humanity. You don't have to be modest.

Kocienda: Yeah, it is important for humanity. Well, I think we hoped that it might be. And the people were just terrific. It was very inspiring all the time to be surrounded by people who were so smart and so good at making the software that we were trying to make. And so, it made me feel I had to keep running as fast as I could to keep up with everybody else, but also if I got stuck, that there were people around me that I could turn to who would have insights or an opinion or some notion about what to do, what the solution might be. So, it was-- thinking back on it now and looking at the product, at how it turned out, to me, it makes sense that it turned out that way. It was not a fluke.

Williamson: I agree with pretty much all of that. For me, also, it was a time when I was working with people that loved what they were doing, just like I loved what I was doing. And when you're in that kind of community, it's really not like work. It's really-- it's a passion. And even though the hours were intense, there were many all-nighters, and most weekends we worked, it didn't really matter. It wasn't-- I didn't-- at no point, I think, [did] any of us thought [sic] about, "Well, what about overtime? We're working too hard. And what am I going to get out of this?" It wasn't like that at all. It's that we were creating something, an artifact, that was really going to change the world. And it was one of the best times of my life.

Kocienda: Yeah, there were-- you would get a problem, and I don't know-- for me, I would get a problem, and I couldn't stop thinking about it. So, just like what Richard said, this notion of like weekends or clocking out, it just was not part of it because for me personally, I didn't work as long hours as other people in the office simply because that's my work schedule. That's how-- I have a certain amount of sensory input I can take in during the day. And then I need to go and be an introvert. But that didn't mean I stopped thinking about the problems. And so, and again, like Richard said, to second his comments, that's what everybody was doing. There was this real alignment. There were not people going off in all different directions with all different priorities about or approach to the project. So, that just made it really--

it's like a superconductor, no resistance. There was never any problem with getting that current flowing in the right direction.

Hsu: Was it difficult on relationships or your family that you were working such long hours?

Williamson: For me, yes, but it wasn't until after the fact that I really recognized that. I had three kids during the course of my work on the iPhone and missed a lot of those early years. And you can't get that back. But strangely, it was-- even though I'm dedicated a thousand percent to my family, I didn't feel that struggle. I felt like I had to work on the iPhone at that time. So, yeah. And you know, there's lots of stories about other people working on the iPhone that had significant relationship problems. I was very fortunate. My wife was very understanding. And I have great kids. But yeah, obviously, family life was impacted because we weren't there. Many of us weren't there.

Kocienda: Yeah, I'm lucky in that I'm married to somebody who is extraordinarily understanding. So, there was never really any tension on our relationship over working long hours or, even when I was home, still thinking-- still so obviously thinking about the work. You just get this far away look, and she's talking to me. And I'd say, "Wait, what?" Like I say, she is just really extraordinarily patient and understanding and understanding that I was doing something that I loved, so she didn't give me trouble over it.

Hsu: Did the secrecy of the project impact the way you guys worked? Was it a hindrance? Or was it beneficial?

Williamson: Not really for me because we were with the people that knew about the secrets all the time. So, it wasn't as though we were at a social event and couldn't talk about things because we worked and socialized with all the people that were working on the same project. And I was lucky enough to actually be one of the people that carried around a phone very early on. And so, my kids, my young kids, all actually got to play with the iPhone before we launched it. And honestly, I didn't hide it from my wife. So, Apple will probably have something to say about that now. But I think Scott knew, and it was-- for me, it wasn't an issue at all, the secrecy.

Kocienda: No, it wasn't. As he [ph?] said, we were all very focused inwardly on making this product as good as it could be. And we were self-sufficient in a way. We had enough people with enough different skills to get that done. Now, there was one time, in particular, while I was working on the keyboard, where I did want to get help from some outside experts at Apple, certainly not outside the company. And I'll address that in a moment. But some folks who had long experience. Obviously, Apple shipped operating systems with text entry systems, particularly for Asian languages, Chinese, Japanese, Korean, some pretty sophisticated input methods and also text processing. And so-- particularly for like junk mail filtering, latent semantic mapping was one pretty sophisticated text processing algorithm. And so, when I was developing autocorrection for the keyboard, I wanted to talk to these people. But I couldn't. Eventually, I talked to Henri Lamiraux, Richard's manager, and then to Scott who was Henri's manager and obviously running the whole software organization for the iPhone. I got the permission to talk to them.

But I couldn't tell them what for. And that actually turned out to be all right because they worked at Apple. They knew that it was something something, some secret. And so, we-- I asked them questions. And they talked about their work. And it was helpful to orient me in this new problem space. It would have been different if it was somehow possible to really enlist their help. But that would have changed the dynamic of the organization. It was a small team. And we were all given the responsibilities to figure out what to do in our own areas. And we did, obviously. So, and when it comes to like looking at other cellphones or thinking about what other companies were doing at the time, or like for me, playing with a Blackberry. I never did. I never played with a Blackberry keyboard to see what their software did while I was making the keyboard software for the iPhone or any other phone, any other Nokia. I mean I think there were a couple phones around or whatever, but I didn't want to look at them. We were making something new. And so, that's what the focus was on.

Weber: But why not then look at-- what would have been the harm in looking at other ones? How did you think about it?

Kocienda: Because we were thinking blue sky.

Williamson: Yeah, in general, that was true across the board that we didn't really look at competitive product. We were creating something new. And we had a lot of really great ideas. So, there was never competitive analysis. And maybe elsewhere in the organization, marketing perhaps had done some of that research. But within engineering, we didn't do any of that.

Kocienda: No.

Hsu: Did you feel like it would have tainted or narrowed, restricted your creativity to do that?

Williamson: Well, "tainted" is a loaded word. So, I think in terms of "tainted," from the legal perspective, I don't think anybody was thinking about that at all, on the engineering organization. And yeah, I think we had the sense of not wanting to be constrained by what had been done before, to try and do something-- if we could do this from scratch, what would it look like? What would be the best way to do it? And just going with a sheet of glass and no buttons, that was really new.

Weber: Did that come from within-- the blue sky impetus came from within the team, or from above, or both?

Williamson: Both.

Kocienda: Both, yeah.

Williamson: Steve [Jobs] really wanted to create something new obviously. And the design team, they had a lot of great ideas. The engineering team had a lot of great ideas. And we just didn't feel the need to do the competitive kind of analysis. We weren't like, "What are we going to do here? We have to look at what did Blackberry do." It wasn't ever like that.

Kocienda: The problem was never, gosh, boy, I wish we had some ideas here. That was never the problem. There were too many ideas. And so, it was really a matter of picking the best ideas from this huge list that we already had, rather than, oh my gosh, we better go out and see what other people were doing so that we can figure out what to do with this thing. That was-- it was not like that at all. We had too many ideas. So, it was really just a matter of, again, focusing inwardly and editing our list and picking the best and then refining and honing and also keeping track of the things that maybe we crossed out earlier because maybe those things-- some of those things would come back later.

Hsu: Ken you mentioned that you had consulted with the Asian input people for auto-complete and things like that. Was that pretty significant that they had-- they had already had this experience doing essentially auto-completion of full phrases or sentences in Chinese or Japanese. And now, you're doing this for English. Was that-- how significant was that?

Kocienda: It wasn't at all.

Williamson: Right, and it's kind of a timeline issue, too. The initial launch, we didn't have a lot of input from those teams. But post-launch, they took ownership of that.

Kocienda: Yeah, eventually, those same people who worked on Kotoeri, the Japanese input method, wound up being the ones who took over Japanese text input on the iPhone eventually. Of course, the very first release of the iPhone was English only. It didn't have any text completion at all. It had only autocorrection.

Hsu: Oh, okay.

Kocienda: If you-- to the extent that there was no spelling correction either, there was only input correction. And it was only if you typed five keys, you would get five letter autocorrections. It was tied to the number of keys that you typed. It was a very, very simple system focused completely on input correction that you couldn't get-- the basic idea was that since you couldn't tactilely feel the keys that the software would help to figure out, give you what you meant rather than what you did.

Weber: But for the literal keystrokes, not for a higher level of either grammar, spelling--

Kocienda: Figuring that you know how to spell what you're trying to spell and that you tried to type the right keys, but you probably missed a couple, so give-- figure out what one of the neighboring keys would be better to sub in here or there.

Hsu: But then you said that the group that owned Kotoeri later on took over just Japanese, or--?

Kocienda: Just Japanese.

Hsu: Okay. So, then the English autocorrection--

Williamson: And Mandarin and--

Kocienda: That same group, yeah, the same group.

Williamson: Aki Inoue, Deborah Goldsmith.

Kocienda: Yeah, yeah, Lee Collins, yeah Kita-san [ph?].

Williamson: Kita-san, yeah. It was a great group.

Kocienda: Yeah, yeah, experts, obviously.

Hsu: The English autocorrection and text completion was-- that was a separate group? Or were they also drawing on techniques that had been pioneered--?

Kocienda: For the first two releases of English, it was still my software. And then we cut over, as I recall, for-- it was Big Bear, whatever that was.

Williamson: Internal codename.

Kocienda: Yeah, I think that was three, the third release, third major release. Then we cut over to some software that Wayne Westerman wrote, which had roots going back to FingerWorks.

Hsu: Oh, okay.

Weber: Really, how interesting.

Hsu: Earlier you mentioned the management chain, Henri, and then Scott Forstall. What was it like working for them? What were they like as managers?

Williamson: Working for Henri was great. He was-- I wouldn't say a hands-on manager. So, he had a-- well, I suppose still has, he's not dead yet-- has a great ability to kind of see all of the details involved in something and keep track of what needs to be focused on. Not necessarily the most technical of people, but he would send out daily and weekly roll up reports of all of the outstanding issues that had to be addressed. And he had great graphs the he would want us to drive down the point on the graph to get to as few bugs as possible. And he was always on top of those details. And one of the things that we did as a management organization is roll up a lot of information to Scott. So, I spent a lot of time talking to Henri about what the teams were doing. And then he would roll it up to Scott. And Scott would put together booklets of kind of status to-- briefing books, I guess you could call them. And so, we spent quite a bit of time doing that. But not-- it wasn't overly burdensome. We always had arguments about status reporting and how much time we should spend doing that. And I think we struck the right balance. But on a kind of personal interaction level, Henri was great to get along with. He was kind of the opposite of Scott. Scott was like hard charging and like, "Do this. Do this. Break the laws of physics," you know, "We can make it happen." And Henri was like, "Okay, let's give it a try." So, he was great to work for.

Kocienda: Yeah, to me, I agree with what Richard said. And I think that's an important dynamic that he mentions that Henri was so different from Scott that you would get-- and sometimes like a good cop/bad cop sort of approach to things. My recollection of Henri is that probably nobody lived on the software from the earliest stages as much as him. And even as-- in my memory, going [to] later on, it's like he-- it's almost-- he was like an extra QA engineer. He'd always be installing the system, always be doing the erase installs or the different kinds of-- or the install, the update installs. And he'd be doing-- and so, he knew what was going on with the software. And was-- and this dynamic that Apple always had is that-- we say dogfooding, or at least that's the term in technology. We didn't say that so much. But that's what he did. He was-- he acted like a customer.

Williamson: This is actually an interesting point on QAs, little tangent from the question about management style. But we had very little QA-specific engineers working on the original iPhone. And that meant that individual engineers had to really be on top of the quality of their software. And I think-- that was because we had dedicated engineers and they were able to do that. Down the road, there were some other things that relate to this that we can talk about later. But I think that was-- it's kind of unusual in an organization to have so little QA support and to rely on the engineers so much.

Kocienda: But I consider, for me, again, my responsibility being text entry and the keyboard-- keyboard and then the widgets as well. Nobody used it more than me. I mean I was on it. It was my responsibility. And so, it wasn't a matter of well, I'll just code this up, and this API looks good, and whatever. And then the testers will test it. No, it was my responsibility. I felt a personal-- if somebody found a bad bug, I felt terrible. How did I not catch that?

Williamson: I think it's true of the whole organization. There was never a sense of just write the code, throw it over the wall, and [the] QA teams will figure it out, never. It was-- I mean it was pride of workmanship.

Kocienda: Yeah.

Hsu: There is this funny story of a woman locking herself in her office. Were you there for that?

Williamson: Yes.

Hsu: Anything you want to say?

<laughter>

Williamson: So, yeah. So, we had almost daily meetings with kind of the key management folks. And we had-- John Wright was in a meeting. I was there. And Scott was there. Kim was there. Kim is the woman in question, Kim Vorath. And the-- I think Nitin was there too probably. I don't remember exactly. And John and Kim got into a shouting match. I mean they were swearing at each other. I mean we'd spent very long hours. Everybody was tired. And they were just insulting each other. And Kim stormed out and-- out of the meeting. And I don't even remember what the issue was. It wasn't even that important I don't think. And she walked towards her office. And she slammed the door really hard. So, the whole building kind of shook. And then she tried to open the door, and it wouldn't open. And she's trying to get out, and she couldn't get out. And then Scott came out of the meeting, and he tried to open the door, and he couldn't get out-- couldn't open the door. And then he called security. And-- but before that-- oh, I think the baseball bat.

Kocienda: Baseball bat.

Williamson: The baseball bat, yeah. He got-- there was a bunch of toys lying around. He picked up the baseball bat and started hammering the door.

Kocienda: Aluminum baseball bat.

Williamson: And I think he broke the lock.

Kocienda: Oh, he broke the whole-- yeah, he broke the whole--

Williamson: He broke the whole mechanism.

Kocienda: Fixture for the lock. I mean, yeah, the next day, you came in. It's just a hole in the door with hardware lying around.

Williamson: Security came by, and they were like “euhhh”-- and Scott said oh, it’s okay.

Kocienda: It’s okay.

Williamson: We need maintenance. But yeah, that was-- I mean it was-- it’s been written about a lot, that incident. But I think it was just kind of an indication of the pressure that was building towards release. And--

Hsu: Oh, so that was close to the-- close to release?

Williamson: I don’t remember the-- it was close to some of the dead-- some important demo deadline.

Kocienda: There’s always a deadline. I mean, really, there was always-- it was always intense.

Williamson: But afterwards, everybody had kind of a laugh about it.

Kocienda: Yeah, I mean Kim worked with John Wright for years after that.

Williamson: Oh yeah. Kim had a tough job because she wasn’t in the engineering organization. She was direct reporting to Scott. And she actually ran the QA teams.

Hsu: Oh, okay.

Williamson: And as well as--

Hsu: She was an EPM and doing program management?

Williamson: Yeah, Program Office.

Kocienda: Program—Program Office, yeah.

Hsu: Oh, Program Office.

Williamson: And the Program Office was responsible for scheduling, some EPM kind of functionality at that point. But mostly it was scheduling and QA. And she was just trying to marshal everybody and get everybody moving in the right direction.

Weber: But you were saying there were not many-- there was not much formal QA testing.

Williamson: No.

Weber: So--

Kocienda: There was some.

Weber: She had-- Okay.

Kocienda: There was some. But not--

Weber: And she was in charge of that.

Kocienda: Not much. If you-- probably-- I'm going to make a wild guess just to get you in the right ballpark, four to one engineers to QA.

Weber: Wow.

Williamson: Or even less.

Kocienda: Maybe-- there weren't--

Weber: Yeah.

Kocienda: Let's say there's twenty-five engineers, and three, four, or five QA people.

Williamson: And that changed after the initial--

Kocienda: It's changed over time.

Williamson: Initial launch.

Kocienda: And I mean QA went up quite a bit.

Hsu: Was it-- were there any-- was she the only woman on the project? Or--

Williamson: One of the very few. And so, we had Vicki, Vicki Murley. So--

Hsu: Who had been with you on Safari, right?

Williamson: Yes, and I brought her over to the [phone] team.

Hsu: And she was a QA engineer.

Kocienda: Yes.

Williamson: Yes. Who else? I think--

Kocienda: Meriko was on the project for a short time.

Williamson: Well, in charge of camera systems.

Kocienda: Yeah.

Williamson: Yeah, but she was an EPM for camera systems.

Kocienda: Yeah, she worked directly for Kim for a little while too before she went back to doing camera stuff.

Williamson: Very few.

Kocienda: But-- HI designers?

Williamson: Oh yes, yes.

Kocienda: Caroline.

Williamson: Caroline, yeah.

Kocienda: Caroline? Caroline Cranfill.

Williamson: But yeah, when it comes to the diversity question, not diverse at all.

Kocienda: No.

Williamson: Very, very, you know, white men.

Kocienda: Couple people of color, couple women, and then mostly white men.

Williamson: Yeah.

Hsu: Yeah. But I mean did any of these social dynamics-- was it affected by that, or--

Williamson: No, well not as far as I was concerned. I mean it didn't make any difference whether you were a man or woman, black or white, or-- I mean the work was what was important, not who you were or how you looked. But you know it's kind of a self-selecting group that now we recognize more and more that it's-- diversity's really important, but we-- that was never even a topic of conversation.

Kocienda: No.

Williamson: The work was so intense.

Kocienda: I mean and I-- again, it shows from the perspective of now, even though it's not so so long ago, that it was so not on our radar. Now, I will say that I don't think that anybody ever disrespected Kim. "Oh, she's just a woman," that was not ever part of--

Williamson: Yeah, her personality--

Kocienda: If Kim asked me to do something, I would just do it because I knew that was her position in the organization. She needed something done by a certain time. And she was the owner of the schedule, and so I just did it. I mean it was not a matter of oh well, she's just a woman. I don't think that anybody thought that way. So, I mean we can wish going back, revising, making a product for the world, that it would have been conceptually better, of course, to have the development team look more like the world. But it just wasn't--

Weber: But do you think it looked very different from Apple at the time in related areas?

Williamson: No. I mean the whole tech industry was like that.

Kocienda: Yeah. I mean it was a cross-section of Apple. But again, I mean that doesn't-- that's not a cross-section of what the world looks like. It's-- I mean, again, you're kind of this self-selecting-- I mean I think that-- I don't really know because I've never spoken to Scott directly about it. But I think he was the person most instrumental in choosing the team. Obviously, Richard and Henri and Nitin had considerable input on that as well. But I don't know that we really hired-- the great majority of the people working on the project just came from inside Apple.

Williamson: Certainly, in the early days.

Kocienda: Yeah.

Hsu: I mean, do you think--

Williamson: Not the great majority, I think all of them.

Kocienda: Francisco.

Williamson: Francisco Tolmasky, yeah.

Hsu: I mean that was an interesting phenomenon that I remember being at Apple at that time was that there was a sort of-- there was-- the kinds of people who would self-select or get picked for these types of secret projects were sort of the cream of the crop of the engineers that were kind of well known to certain managers, I guess. And then there was everybody else who would just be sort of stuck in their whatever organizations they were. And I'm just wondering if there's sort of this-- I mean you mentioned self-selection, right? There's sort of this self-selection process of I'm willing to work eighty hours a week kind of a thing that tends to select for a certain type of person.

Williamson: Oh, absolutely. I mean it comes from the kind of person who wants to work at Apple in the first place, then within Apple, the kind of person that wants to really become that superstar. Not everybody wants to do that. But I don't think that necessarily indicates white male or any kind of-- or gender. But there is-- it's merit-based. If you have people who work really hard, they kind of rise to the top and work on those projects. This whole diversity question is really interesting because back in a decade ago, it wasn't on the fore of anybody's mind. But now, it is. And I think that's a recognition of the maturing industry and the sense of how important it is to the world at large. It's just-- those were different times.

Hsu: Mm-hmm. Do you think that the-- I mean the idea of “merit-based” itself and being a “superstar” and being in a highly-- high pressure, possibly even competitive, environment tends to select for male engineers over female?

Kocienda: See, here’s what I would say about that. I don’t want to use-- I’m not going to use a word like “merit.” I’m not going to use a word like “superstar.” I’m not even going to say-- well, let’s leave those words out. I think that what happened was there was a lot of stress. Scott primarily first and foremost was given a responsibility and a timeframe and a brief. And he turned to people that he had experience with and success with in the past.

Williamson: But I think it’s more complicated than that. And I will use words like “merit” and “superstar” because you need those kind of people to execute a project like that. But it goes back beyond outside of Apple, like just you look at the pipeline of people that were going through computer science, we didn’t have that much diversity back then. And we still don’t have enough diversity. So, it’s really-- it goes back to the kind of the education system and who has the appropriate skills to get into a role, a technology role. And so, already, the pool of potential candidates is kind of winnowed towards-- biased towards white males. So, that’s changing, but those weren’t the times that we lived in then. And-- I’m not afraid to say that we weren’t diverse. We weren’t diverse.

Kocienda: No, I-- no, I’m not afraid to say it either. We weren’t. We weren’t. And I-- to me, I think it has-- it’s as much of a personal trust issue as opposed to-- which I think is very subjective, as opposed to objectively, somebody is-- has merit or has skills.

Hsu: So, you mean like “culture fit” or something like that.

Kocienda: Yeah. And again, it’s just-- it’s a matter of “you trusted me. I trusted you.” I mean, I d-- what necessarily that-- I don’t want to maybe rathole on this all day. I think we probably could. But--

Williamson: I wouldn’t call it “culture fit” because that implies a bias that’s not related to the work.

Kocienda: Yeah. That was-- it’s really all about the work.

Williamson: When you say-- culture’s an important word. I think what you mean by “culture” is a certain kind of work ethic, rather than you like this kind of movie or that kind of movie.

Kocienda: Yeah, yeah. That’s--

Williamson: It’s a work culture.

Kocienda: It's a work culture. Yeah.

Weber: But also, what percentage of the hires do you think were people that were well known to somebody else because often, it just reflects--

Kocienda: High. Very high.

Weber: If people's network is people like them, then when they look for people they know, I mean they choose people they know who are like them.

Kocienda: Right.

Weber: So, I mean was there a high percentage of-- just there weren't many people being recruited from who were not in the immediate circle of some of the principals.

Kocienda: Right. I mean you take again somebody like Vicki Murley. Why was she brought in onto the iPhone? We worked with her on Safari and WebKit. And we knew she could do the job. And that's-- and so, there. There you go. I mean it's-- she got-- she got the opportunity in much the same way that you did, much the same way that I did. And she got that opportunity. And so, you know, we were in this--

Williamson: Network.

Kocienda: Network. That is just not-- it was very small.

Williamson: But I think that network is true even in a community where you do have diversity. It's just something you need in terms of advancement in a career. You have to build a network, whether you're a woman, a white man, or anybody else.

Weber: And were there many-- I know there were some. But it sounds like there weren't that many foreign-born people of any gender or background.

Williamson: I'm foreign-born.

Weber: Yeah. But you, Henri, right?

Kocienda: Bas.

Weber: That's true.

Kocienda: Marcel.

Williamson: I'd say a lot of foreign-born people, at least first generation. Probably more immigrants than non-immigrants. I mean that's pretty typical, in fact. Look at Steve Jobs himself. His dad came from what? Syria, I think.

Kocienda: Syria. Yeah, his biological dad.

Hsu: Let's move on to talking about the keynote. So, how much work did you guys do for the keynote? Or were you guys working on the keynote? And at what point did you start working on it?

Williamson: Timeline questions. Yeah, I was involved in the keynote to the extent of getting the software to perform perfectly through the script. And all of the priorities became not priorities except things that directly impacted the keynote. So, to that extent, I think I was involved and most everybody was involved in terms of knowledge about what had to be done for the keynote. In terms of actual helping Steve rehearse, that was not part of the engineering team's role. I was there to make sure nothing went wrong on the technology side. But yeah, I mean maybe you know more about the timeline. It's hard for me to say exactly when and how much we--

Kocienda: Yeah, I don't have specific memories about that keynote. Generally speaking, they started a little bit more than a month outside of the-- counting backwards from the day, a month to five weeks, maybe as much as six sometimes. But that's-- I have some memories of some things getting locked down, starting to get locked down at the end of November 2006 for then the January 2007 release. But like Richard said, this was all about so-- it was all about the software. When I went to the hall at Moscone that morning, I had no idea what the presentation was going to be, no idea whatsoever. It was just all a matter of getting the software in order and knowing like that the keyboard was going to get shown, that like New York Times was going to get shown and Safari, certain little pieces like that.

Williamson: So, pretty much for every keynote, there was a script that was mailed out. But it wasn't generally mailed out, so all of the senior managers had the script. And we had to run through the script on our own devices and make sure everything worked and track the bugs that were associated with the script. But Steve did a lot of rehearsals with Phil [Schiller]. And Phil's like the consummate-- they used to call Steve "the talent." And you had to make everything perfect for "the talent." And so, there was that whole kind of showmanship side of it that we weren't involved with other than making sure the software was flawless.

Hsu: What was the actual day like for you?

Williamson: It wasn't. But the one-- I mean it was fantastic. The energy in the room was awesome. But the thing that sticks in my mind the most was after the keynote was over, I was in this small gaggle of people around Steve. And he reached out, and he shook my hand and said, "Good job." I mean that was-- it was one, two words. But it just like-- all of this work, all of this effort came to a culmination in that moment. And Steve had been my boss and hero for so long. To have him give me that affirmation, it felt pretty good.

Kocienda: Yeah, I was standing next to him. And so, Steve didn't know who I was at the time. I met him later and demoed for him a few times later on, but during the iPhone-- so, it's-- but he knew-- I saw him see Richard. And he, yeah, made sure that he recognized him. And that he thanked him. And it was pretty-- it-- to even just be that close to Steve on really the day of his-- it's kind of the best one, isn't it? It's the best product, the best keynote. I thought that he couldn't have done better. It was just a-- it was a great day. Yeah, it was a great day.

Williamson: I mean it's-- I think Steve's probably best talent is his ability to make technology so accessible and to make you want to use the technology. I mean he-- and I think that's because it came from the heart. He wasn't faking any of it. He really was passionate about it. He had that incredible charisma that-- it kind of radiated. It was great.

Kocienda: Yeah. It is true, too. I mean you think that somehow you get up on stage to do this big keynote, and that it's a-- obviously, it's marketing, a huge marketing opportunity to get that attention to talk about a product and to get then people to echo what you say. But for Steve, it was really true. I mean you go back and go back now, look on YouTube, go and look on-- search on the web in articles or whatever, Steve talks about products. That's what he talks about. You say, "What is Apple about?" "Apple's about making great products." That's what he says just over and over and over and over. And that mantra coming down from him, that's why we were where we were because we felt that way too. If you didn't feel that, I don't know how you could have lasted through the intensity and through the work, through the pressure, whatever. It all communicated down. And then it all came back up on that day and came off just wonderfully, yeah.

Williamson: The other thing about Steve was always the lack of cynicism. I mean he really wanted to make something great. You look at a lot of other technology leaders, and there's an ulterior motive. They wanted money. They want to do this or that. But Steve, it was just about making the great products with no alternate agenda. It was-- that was the core of his being.

Kocienda: And the belief that if you do that part right, the other parts of the business are possible. Making money is possible. And getting good people to come and work for you is possible. And then having nice food in the café is possible because the company is making enough money, so you can afford-- all of these things stemmed from, all tied back to, making the great products.

Williamson: Steve would actually say that. In Town Hall meetings, he'd say-- people would ask questions, "What about the stock?" And Steve would say, "Make great products, everything else will follow." He literally said that.

Weber: So, build it, and they will come.

Williamson: Yeah.

Kocienda: It really is. It's his own version of just that. And so, again, that's-- why did we wind up in those positions to be working-- a small group of people working on this product? Because that's what we believed. So, again, this alignment is such a big part of why that product turned out the way that it did.

Hsu: So, then after the keynote, there was still a bunch of stuff that you needed to finish, right? So, let's see. There was-- so, you needed to do the YouTube app and then technical approval. What else did you guys need to do? Or talk about the other things, YouTube or the approval process.

Williamson: Well, YouTube was relatively straightforward. We had a really good relationship with Google at that point. And John Harding, I think, was my main point of contact. And he was-- they just bent over backwards to help. And we wrote the client again, just like the map app. And they provided the service. And I remember several meetings with them on different topics-- kind of, this is very technical detail. But we were talking about HTTP streaming and the new technology that Apple was developing that we wanted them to push. But that's-- they were accommodating. And I spent a lot of time up in-- their offices-- they had a-- I remember they had bought a new build-- we were squeezed for space. So, at this point, we were bursting at the seams in Infinite Loop. And they had just bought a huge building. And they-- it was four stories I think. And they occupied one story. And there were three empty stories. So, I remember going over there and saying, "You got all this space. We have to squeeze two or three people into an office." But it was pretty straightforward to get YouTube finished. They, like I said, at that point, they were very helpful. And Steve and Larry [Page] were still on good terms. And as far as approval testing, I'm not sure what there is to say about that.

Kocienda: Yeah, I don't know. My recollection of that period between January and June, from the announcement to the release of the product, was polishing, getting the product working right in all of-- we-- the demo for the-- at least for my part, the demo on stage in January, everything that was shown was real. I mean there was very, very little faking of--

Williamson: Oh-- sorry. Approvals, yes, AT&T. Okay, now it's coming back to me. This was horrendous. It was horrible.

<laughter>

Williamson: This was a nightmare. That's probably why I blocked it out.

<laughter>

Williamson: So, we had these horrible meetings with AT&T. So, the cellular industry up until that point was coming from a very different background. And they had these books of specifications and stuff that you had to comply with. And it was ninety-nine percent nonsense. But you had to do it. And this largely fell on Nitin's shoulders I think. And but it was a nightmare. And they just-- there were things that they wanted us to do for like SMS that were ridiculous. And they said, "Well, you've got to do MMS if you do SMS." It was-- and we were like "SMS and MMS are ridiculous." These are obsolete technologies. Having a character limit on messages is ridiculous. Having an alternate format for media is ridiculous. But we had to trudge through it. And then that's everything from, what do they call it, E99 approvals for emergency calls. And it was horrible. And it was kind of a sign of a legacy industry and what they-- what slows them down is all of this burden from specifications that really don't make any sense if you think about what you want to do to build a great product. So, but my key goal with AT&T was to ensure that we had a clear pipe, that there would be no transcoding. And that was written in the contract. And they kept wanting to say, "Well, we need to do this. We can't keep persistent connections open. We have to transcode the content so that it will format correctly on the screen." "No, you don't have to do any of that. Just give us the bits. We'll take care of it." So, yeah, the approval process was painful. And Brian Cassidy, I think, was on board at that point. And he was the main liaison with AT&T. And his life was hell having to deal with these people because it was-- it was just completely different mindset in terms of technology development. And we didn't have books of specifications for the iPhone. You couldn't go to a book and say, "Well, does this feature match the specification?" No. There wasn't anything like that. And you know.

Kocienda: No, no, it was--

Williamson: But for AT&T and everything had to meet exactly the specification even if the specification didn't make sense. So, there were a lot of fights. And like I said, Nitin I think probably bore the brunt of that along with Brian Cassidy.

Weber: But early on, hadn't-- because I think Scott Forstall talked about-- was it him? That they negotiated something where AT&T only controlled the lower level. And they would meet the specs for that but not above.

Williamson: Perhaps. Yeah, I'm--

Weber: Maybe that came after this, I don't know.

Williamson: Yeah, but we had to go through hell with these specification approvals. And it may have been that yeah, once we got deep into it, Scott just said no, this is enough. And I wasn't directly involved with those negotiations with AT&T. I sat in on those meetings and fought for my clear pipe.

Weber: And these were big meetings regularly, or--?

Williamson: Big meetings on their side, they would typically bring more people than we brought. We tried to minimize the impact on the team. So, Kim spent a lot of time in those meetings, Brian Cassidy, Nitin, and me. But as far as engineering teams, we just--

Kocienda: I was never in any one of those meetings. Just I was working on the technology.

Weber: And they would come to you, then, generally?

Williamson: So, there was a few meetings where they-- I didn't actually go. I think it was in Virginia maybe. I could be wrong. There were a few meetings off-site. But they came to visit us a few times. They actually came through into the secret area. But I remember we-- what was the conference room called? The big--

Kocienda: "Between."

Williamson: In "Between," yes. We had a conference room called "Between." And it was where we had most of the internal iPhone meetings. And they would come and sit in "Between," which was through the second secret barrier. But anytime they came to visit, we made sure that everything was roped off and--

Weber: Draped.

Williamson: Oh, it's sort of an interesting story. So, I forget which one of the meetings it was at, but we showed Ralph-- was that his name? The CEO? The phone, and it was one of the first times you actually got to see it and touch it. And it was a very early version that wasn't complete. Don't ask me the time relative to launch, but it was before launch. And everything on the phone was localized. And the way localization worked is you have a key. That key is looked up in a table, and the actual string that you represent is chosen using that key. And one of our engineers, who will remain nameless, had used a foul word as a key. And it was something like-- it was an "F you" word. "F you" was in the phrase. And Ralph is playing with the phone. And he sees this string with this "F you." And he just flipped his lid. [He says,] "What's this? What's this?" And that came back to us and caused a huge stir within the team.

Kocienda: Scott yelled.

Williamson: Scott yelled, yeah. The engineer happened to be on my team. And so, I got yelled at. And I yelled at the engineer. And we resolved it. And it would never have made it to the public. But the fact that the CEO of AT&T saw this on the prototype phone was--

Kocienda: Bad.

Williamson: Pretty bad. Afterwards, we all had a laugh about it. But when it actually happened, it was very-- on top of all the other stress, this was a stress that we didn't need.

Kocienda: Yeah.

Weber: And so, the CEO was coming to a lot of the meetings himself, then?

Williamson: No, no, no, no, this wasn't one of the approval meetings.

Weber: This was exceptional, yeah.

Williamson: This was-- I wasn't there when he actually saw this. It was just-- it came to mind as part of the AT&T approvals.

Weber: But who were the main people on the AT&T side?

Williamson: I can't remember their names. I mean that-- my memory capacity is limited. And I'd rather not fill it with details like that.

Kocienda: Flushing the cache there.

Weber: Okay.

Hsu: But-- you said you don't remember the timing, but do you know if it was before the keynote or after?

Williamson: That Ralph saw this?

Hsu: Yeah.

Williamson: It was before the keynote.

Hsu: Okay.

Williamson: Yeah.

Weber: And I mean the talks with AT&T went way back to the beginning of the project almost.

Williamson: Yes, they were ongoing, of course.

Hsu: Richard, you mentioned having the phone very early on. I mean what was that like to sort of walk around with the phone in the wild before anybody else knew about it?

Williamson: It made me feel special. No, it was pretty cool. But I wasn't spending a lot of time outside of work, so certainly not in social activities. So, it wasn't like I was so worried about showing it to people. But my kids were very, very young. So, I'd worry about them spilling the beans when I showed them. But getting Camila and Ian to interact with the device was very, very telling because when they could pick it up and just play with it and kind of intuitively know how to use it, I knew that we'd done something right. And so, in a sense, they were like beta testers. And I think we got a lot of things right like the swipe to open gesture. It was intuitive. Yeah, swipe to unlock. And they-- then when they picked up the device and started to play with it, their face would light up. So, that's the extent to which I really used it outside of the office was just with my kids. And so, it was-- but you know, when you have something that you think everybody else is going to want, and you have it first, that feels good.

Weber: How old were your kids at the time?

Williamson: Pardon?

Weber: How old were your kids at the time?

Williamson: Oh, dates again.

Weber: Roughly.

Williamson: So, Camila was born in 2003, and then 2005 and 2006, so young.

Weber: Really young, yeah. So, perfect for that.

Williamson: Yeah.

Kocienda: So, I recall-- so, it was in January still, it must have been just a few days after the keynote, that's when I got the-- my phone that I could start carrying around with me. And there was a death in my family. So, I had to go back to the East Coast. So, I remember being on the-- and I took the phone with me. I got permission to do that. They said, "Make some phone calls. Check the network in New York." So, I remember being in like New York state with the only iPhone probably that was there at the time and making some phone calls using it, but really keeping it in my pocket, trying to be pretty discreet about it. And so, I wound up in this family gathering because there were some people who I hadn't seen in many, many years, again, because of the reason that I was there. And so, I'd like get like a phone call, put it back. And some older relative comes over and says, "Hey, what's that?" And he grabs it out of my hand, which was just absolutely not supposed to do. I never told you this. And starts to say, "Hey, what is this thing. Show me this thing." And I was all like, "Give it to-- please, just give that back to me." So, I probably-- it's like I said, I wouldn't have told anybody. But yeah, he's like got this thing and started playing around with it, "Hey, what is this thing? How does this work?" And I was just completely probably just was white as a sheet because there was just no way that we were supposed to be giving demos to people or in any way telling them about it. But here's this old relative who I had to be at least moderately polite to and not shout at him saying "please give that back to me."

Williamson: In terms of the phones outside of campus, I remember having several meetings talking about how they could be transported for people that weren't carrying them. And we looked at these different pelican cases. And we thought about who-- how many security should we have with the pelican cases to transport the phones around. I mean it got to that level of insanity for the security. Yeah. But we-- because there were few-- a few the-- the barstool episode. That's a-- remember the-- who was it that left the phone in the bar? That was probably after--

Kocienda: That was probably-- yeah that was-- that was an iPhone 4.

Williamson: Okay, way later. But yeah, the stories around security are crazy-- and transporting phones.

Hsu: Do you guys remember the last thing or the last bug you fixed or the last thing you worked on right before the phone shipped, the last feature you got in?

Williamson: No.

Hsu: Under the wire?

Williamson: I don't. I mean so much, it's just-- good software is the accumulation of thousands of small details. And every one of those thousands of small details has to be perfect for the software to be really good.

Kocienda: Yeah, see, you know my recollection is that it was this process of working-- spending more and more time on less and less. And so, that by the time it got to-- and our job, our job was not only [to] engineer the software but to engineer that process. So, by the time that we got to the point where were getting really, really close, yeah, we were really still very, very intense and spending a tremendous amount of time just like always. But it wasn't a matter of getting something to work that didn't work before. It was just--

Williamson: And the threshold for changes goes up. Every change is a potential bug to be introduced, right? So, you really want to ratchet up the level of what you'll accept. And Kim was very good at this. We used to have BRBs non-stop, bug review boards. And they would be daily or sometimes twice daily moving up towards the release into the launch. And every single bug that was going to be fixed or every modification to the software would be scrutinized. And it's the process you go through. That's part of the craft of making good software.

Kocienda: If you had a bug [fix], you needed to go to this bug review board and defend it. And it's-- there would be the devil's advocates who would just say, "No, we're not--" The base assumption is—particularly, the closer you get to release, the base assumption is no, we're not going to change this software, convince me that we need to.

Williamson: Right, and Kim would-- was the no person. But there were actually features-- I can't think of a specific feature, I wish I could. But there were, I remember on occasions, there were things that weren't really bug fixes that we wanted to push through, especially if it came from Steve. And we had to convince the Program Office, Kim, that it was worth doing this. And I wish I could think of something that we did at the last minute. Maybe it'll come to me. But that process continued throughout all of the-- and I'm sure continues today, throughout all of the releases of the phone. And it's typical I think of most software.

Kocienda: I think it's typical, yeah. That's just how you do it.

Hsu: What was it like for you guys the day the phone shipped?

Williamson: I don't actually remember because I think we were probably working on the next thing. I remember getting the box and thinking hey, this is really cool. I don't remember the actual day it shipped.

Kocienda: Oh yeah, I actually do. We had champagne.

Williamson: Oh, we did.

Kocienda: Yeah, we had champagne in the office.

Williamson: Now, get back to work.

<laughter>

Hsu: So, you guys stayed at work. You didn't go out to see the lines or anything?

Williamson: I didn't.

Hsu: Okay.

Weber: But then you could use your phone openly. Did that change anything?

Kocienda: Yeah.

Weber: Did that change anything?

Williamson: Yeah.

Weber: Did that feel different?

Williamson: Yes. I remember having a dinner with my brother and his wife. And she had a friend, Bill Gross, who's-- he runs Idea Lab, which is a venture capital company down in Pasadena. And he's a total technologist. And actually, my brother ended up getting divorced. And his wife ended up marrying Bill. That's another story. But he was at this dinner. And he hadn't seen a phone yet. And it just felt so good to show this technologist the iPhone and say I worked on it, because he's done quite a bit. And he's funded quite a few successful companies. So, that felt good. And I think-- this was maybe the day after the launch-- the day after the phone shipped. So, it was very close to when other people had them or were trying to get them. But we-- mostly engineers ended up having drawerfuls of phones. You have so many-- I probably had twenty or thirty different variations of the phone at least. And so, the appeal of getting another phone from the store, lining up, you know, no.

Kocienda: So, I left Apple this past May, so May of 2017. And when I cleaned out my drawers, I had an iPhone museum in my drawers, I mean really, just a-- I could probably have maybe every one. I don't know, but close to it.

Weber: And Apple wanted them back presumably?

Kocienda: Oh, They were all Apple's property. Oh, I didn't keep any of them. I mean it's-- all belong to them.

Hsu: Okay, so--

Kocienda: And there was-- one thing to say about that which is those were special phones in that they were made for developers. It was a hardware change that we called them development-fused devices. Right, so there was actually a change in the hardware. So, I could plug in a thirty pi-- the old phone, a thirty-pin connector, and telnet right to the phone.

Williamson: Right, ssh in.

Kocienda: So, they were not hardened in the same way that a production-fused device is, which just prevents that level of-- so-- that level of access to the phone. So, what this meant is that there was-- you could kind of get the phone right where you-- get it in a headlock.

Williamson: You could run top. You could kill processes, do whatever you want. It was just like a--

Kocienda: Just do whatever you want.

Hsu: Basically, like a jail broken phone.

Williamson: Yeah, exactly like that.

Kocienda: Yeah. Yeah, but purposefully built to get everything out of the way of the developers so that it was easy for us to work on. So, everybody made a whole bunch of scripts and shell commands and little development environment things because, of course, way back in the beginning, we had very, very little support from our tools as well. So, we kind of needed to build up a bit of an environment around how we developed.

Williamson: Right. We need to copy over new versions of the UIKit framework and all the other libraries and change everything on the system. There's no way we could have done that without the dev devices.

Hsu: Right. So, you're just basically treating it as another-- like a Mac.

Williamson: Like a Mac, exactly like a Mac.

Kocienda: Like a little computer, yeah.

Williamson: Yeah.

Kocienda: Yeah.

Hsu: Well, actually, that's a pretty good segue to my next question which is sort of the-- going back-- we touched on this last time, but the whole decision to open up the phone to developers to do the SDK. Sort of can you maybe go into like the process of that decision, how that was made, who was involved in that decision, who was advocating for that or against that?

Williamson: So, I think we talked quite a bit about this last time. But so, I would say leading up to launch, there wasn't a lot of thought put into it. We hadn't even got our own apps. They were just getting baked. So, but obviously post 1.0, the issue came up immediately, what would be the appropriate path for developers. And all of our apps were developed internally using UIKit and the other frameworks that shipped with the device. So, but that-- the APIs hadn't really been sanitized for external consumption. So, there was some discussion internally about what could we do. And the-- we'd looked-- Dashboard had been relatively successful of a-- maybe not-- approach to widgets on the desktop. And they were all HTML-based and used web technologies. And we'd augmented WebKit with some technologies to support Dashboard. And we thought possibly this is an approach. In fact, as we talked about, there was some of the early apps were developed in that way. And so, we-- there was-- that kind of debate was ongoing among the developers, web-based versus native-based. And we had come to the conclusion internally prior to the developer story that we could be much more efficient as developers using a native API than-- rather than doing development with web technologies. And--

Hsu: And that had been made fairly early on?

Williamson: Well, it was actually pretty late, but as we clarified with Scott [Goodson, over e-mail], before the launch. And--

Hsu: Right, but after the keynote?

Williamson: No, no-- well.

Kocienda: No, that's what Scott Goodson said. It was after the keynote and before the release of the-- the customer ship.

<break>

Hsu: So, yeah, we were talking about the decision to open up the phone. And actually, during the break, we were talking about sort of the difference between the P1 and P2 and the way that P2, you guys saw the phone as a general-purpose computer. And P1 saw it as an appliance. And I mean in a way that kind of leads to this sort of-- also this decision to open up the phone, right, because if you see the phone as a computer, that sort of logically lends itself to the idea of opening up the phone to developers. I mean, I was wondering more specifically who was on what side of that debate.

Williamson: Yeah so, I mean after launch, there was a clamor for some kind of story to-- or some kind of approach to support developers, thirty party developers. And I and the iPhone Safari team had a pretty deep background in WebKit and web technologies. And so, I was in a position to try and advocate for that, although I had already come around to the idea that we should-- native development was far more efficient than web technologies because of the tools, because of the fragility of web technologies, and the-- if you look at the Objective-C frameworks, they kind of guide you towards good development, guide you towards how to build an app. And HTML was never designed to do that. This is [a] continuous debate that we've had over time. And but honestly, when-- was the announcement for-- it was at WWDC--

Hsu: It was at WWDC, yeah, which was prior to launch, actually I remember. Was it?

Williamson: But we didn't announce the developer story prior to launch. That didn't come until after launch.

Hsu: So, was it July then?

Williamson: It might have been the first WWDC after launch. I'm not-- somebody's going to have to check dates.

Hsu: That might have been July. I think that might have been July.

Williamson: So, within-- I think I said this last time, within the engineering organization, there wasn't a final approach to say, this is what we're going to do for developers. In fact, it was only a few days before the announcement that Steve made that--

Hsu: At WWDC.

Williamson: At WWDC, that we heard that there was a great developer story. And really Scott [Forstall] and Steve huddled and came up with this. It wasn't vetted by me, certainly, and I don't think Nitin or Henri. So, we had to really scramble to try and figure out what that meant. And I think it was pretty clear to the engineering team that it wasn't going to be a great solution. And in fact, it wasn't well-received for all of the reasons that I've already stated.

Hsu: Within the team?

Williamson: Within the team and externally within the outside developer community. So, I look at it really as kind of a stop-gap announcement. And I think Steve said to Scott, "Look, we've got to do something about thirty party developers. What are we going to do?" And it was one of these snap things that Steve just wanted something and had to have something. And Scott came up with the story. So--

Hsu: So, you think that it was actually Scott who came up with the web story?

Williamson: Yeah, that was totally his decision. It wasn't my decision. I was not involved in it. I don't think Don Melton was involved in it. Don Melton was on the OS X side.

Kocienda: I do recall that-- and maybe this is skipping ahead a bit but where we were going to get to it eventually. But I do recall there was like a Friday afternoon where some Safari and WebKit people from the outside team came to our iPhone hallway and were looking around at offices. "Oh, I'm going to sit here. I'm going to sit here," because they were actually going to come and do this web development story. And that was like Friday. And then Monday, it was like redone. It was like a very quick turnaround that we're going to do this story. We're going to do this story. We're going to do this story. We're not going to do this story.

Williamson: But I remember after the announcement, I did spend time working with the Xcode team, and we-- because there was no tools for HTML layout for-- like Interface Builder in Xcode. And so, we tried mightily to make it work. But it was pretty clear that it was not an optimal solution. And I got on board after the announcement, but I think it was, like I said, a stop-gap measure and wasn't well thought through. And you know, it's a footnote in history now.

Kocienda: Right because then we just went and did the Objective-C API and decided to go and--

Williamson: It does actually bring to light some of the enhancements we made to WebKit and the-- getting back to open sourcing WebKit. And we added touch events and a whole touch system to HTML. And WebKit was open source, but we wanted to see what we could do to keep these components proprietary. And so, we went through all kind of machinations to-- we released the tarball of the WebKit source, but we kept some of the components in a separate area. And it was-- the lawyers were getting involved at this point. And we were starting to patent everything. And so, we have a patent I think on touch events for WebKit. And so, yeah, we were trying to add things to the web technologies to make web apps more capable on touch devices. But I'm personally not a big advocate for that approach. I think there are certain kinds of applications that are great for web technologies, but other kinds of applications are great for native technologies.

Weber: What would the user-- obviously, speed could occasionally, at times, be an issue. But otherwise, what would the user have noticed differently if WebKit had remained the primary?

Williamson: So, you can write a web application that works well and performs as well as a native application. It just takes a lot longer. And maintenance of that application is more fragile. So, if you want to evolve an application, it's very brittle. Things will break as you make changes. The other thing is, is deep integration into the system. So, things like text selection, if you try and do text selection on a webpage, it's kind of wonky. And native widgets versus web widgets, table views, these things-- you can emulate a lot of the native components with web technologies with enough work. And it is possible to make a user-facing application that is good. That's not really the argument here. It's the amount of development effort that it takes to do that.

Kocienda: To me, it begs the question like how does the system behave. So, is the native code kind of the canonical set of behaviors? And is-- would it then-- if so, assuming that, then is it the web technology's responsibility to emulate that bug for bug, feature for feature, detail for detail? Because-- and that's impossible. You're always going to get this little-- there's going to be this little bit of slippage. It's going to be this uncanny valley. No matter-- and what Richard said is true. You can make HTML and web technology work well in an app setting. But it's not going to work exactly the same.

Williamson: If you look-- we mentioned Francisco Tomalski. He was on our team early on, very smart junior developer. After he left Apple, he decided I'm going to make UIKit for the web. And with JavaScript, he came pretty close. He wrote this whole framework that was called-- what was it called? I forget what it was called.

Kocienda: Cappuccino?

Williamson: Cappuccino maybe. And--

Kocienda: Was it-- I forget.

Williamson: I forget. But his company was 280 North. And he wrote applications on top of his UIKit-[like] framework. And they look pretty darn good. They look pretty much like native applications. So, you can go that path. And certainly, if you have a deep knowledge like Francisco did-- does of web technologies, you can do it. But that's not the point. The point is maintenance and development and carrying those apps forward.

Weber: Would there be anything a user would notice. I mean are there certain things that are just-- would be so difficult to do in one or the other that it would change the nature of the user experience really?

Kocienda: Again, I think it's just that you get down to the behaviors of the subtleties of the system would just be different. Better? I don't-- they're just going to be different. And so, what I think users would notice is that the finish of this whole phone, I use one app. And it kind of does this subtle thing in this way. And I use this other app, which seems like an app for ninety-nine percent of the things, but it does this one percent of thing that's different. And I can't quite figure that out. Plus, from the standpoint of being a platform provider, now you need to develop two systems. And you need to make them compatible. So, you go, and you come up with a great idea. And you add a feature over here. Now, you're required to go and add that feature over here.

Williamson: Here's a specific example--

Weber: But originally, you would have been maintaining native apps only for Apple use, right, and WebKit for everything external?

Williamson: But still, we would have to do two-- for Apple apps. But here's a specific example, maybe a trivial one, but specific. Changing the text size. You can go into the system and say okay, make the text everywhere larger. That's going to apply to native apps but doesn't apply to web apps unless you add that capability to all those web apps. But those web apps would have to use a framework that had that capability, which they wouldn't. So, it's--

Hsu: You're now doing twice as much work essentially to--

Williamson: Yeah, you end up doing twice as much work with these parallel systems if you wanted to maintain that, which is untenable.

Weber: And how about things like-- oh, sorry, shared access to say the address book or things like that?

Williamson: Sure.

Weber: All those things--

Williamson: Yes, privileges, yeah.

Weber: You could make them the same. But it would be more work.

Williamson: That's right. And then you're kind of faced with having to go through W3C if you want to add extensions. Do you do it just as a proprietary extension for Apple, or do you try and standardize it? For touch events, we came up with something that we thought was really great for Apple, but then we went through a big process to try and get it standardized.

Kocienda: So, you also think about the-- so, the WebKit, it would-- we would need to add-- to implement the feature that you suggest, access to the address book. We need to add that feature to WebKit. But then is that feature going to be available to some webpage that you just download from the Internet, presuming that a web app would still be going through some kind of app approval and app distribution system that was under the control of Apple, since it's the-- continues to be-- was then and continues to be the whole notion of the phone. Even if you were going to be shipping web apps, kind of the idea was that you'd be going through this walled garden kind of thing. But now, WebKit would have this feature that can access the address book. It's a whole privacy issue. Any webpage on the Internet can go and read your address book. Do you want that? Probably not.

Williamson: It kind of gets back to one of the fundamental approaches at Apple in terms of APIs. We tried very hard not to duplicate things. There's one path to do this particular thing, not two or three. Microsoft, there's APIs that are duplicated all over the place to do the same kind of thing. So, having two completely parallel systems just would be horrendous to maintain. So, one way to do one thing means native networks-- I mean native frameworks.

Hsu: I mean the situation sounds a lot like what you actually had on OS X, which was Carbon and Cocoa, two completely parallel ways to do an applic-- both native, both fully native.

Williamson: Which was a nightmare.

Hsu: And they didn't work exactly the same way either.

Williamson: That's right.

Weber: But WebKit apps, you could have also theoretically ported to different hardware-- ported to Mac OS more easily or something like that rather than--

Hsu: But Mac OS [X] ran Objective-C already.

Kocienda: So, them running in a space like this on the Mac?

Williamson: In fact, iPhone apps actually do run on Mac OS in the simulator. So--

Weber: Put it another way, what were-- what did you lose by giving up the web? There are some pros to WebKit.

Williamson: So, I think if your concern is writing an application--

Weber: From a user point of view I mean.

Williamson: From a user point of view, if your-- well, from a user point of view, I'm not sure there's any benefit, really. But from a developer's point of view, if you want to run on multiple platforms, if you want to run on Windows, on phones, and on OS X and on Linux, then web approach is probably a good approach.

Weber: And there's no-- just because it's the web, doesn't mean you can create a link from within a WebKit app to the open web or anything like that? I mean does it give more capabilities for cross-connections?

Williamson: Well, sure, I mean you could browse web content in a WebKit based app, yeah.

Kocienda: You can do that in a native app, as well.

Weber: True, right.

Kocienda: There's no reason why you can't do that.

Weber: Or vice versa, that you could link to content in a web app more easily? I guess there's nothing inherent in--

Williamson: No.

Weber: Okay.

Williamson: I mean I think ultimately, it was pretty clear the right decision was to use native frameworks. And but as I say, that doesn't mean there's no place for web content and web apps.

Kocienda: To me, the web is about-- is an open document format. That's primarily what it's about. It's not a system to emulate a native toolkit on a particular platform. That's not playing to its strengths. It can do that. But I don't think that's playing to its strengths.

Williamson: That's what the web originally was, a document format system. But I think it's changed today. Now, it is about applications. You go to Amazon.com, that's an application. You go to--

Kocienda: Right, but-- sure. But that's a-- it's a web native application. It's probably maybe the best one. That's the one that I thought of just a moment ago where I just-- so, we were thinking the same way.

Amazon, it’s an app, no doubt about it. But it’s an app based on documents. It’s a whole different-- it’s kind of a central paradigm.

Williamson: But it’s kind of interesting if you look at--

Kocienda: URLs.

Williamson: If you look at the-- Amazon has a native app, but the native app basically is just a web view. It’s just web content.

Weber: Yeah, I mean the web’s gone back, certainly, the early Java interest was pushing it more in an application direction, whereas it’s gone back and forth over the years.

Williamson: Like Java could have been like the lingua franca for high level frameworks. You know, they could have been implemented in Java. And they would run on all platforms. But it never really took off, yeah.

Hsu: So, then I know you’re not good with timing again, but let me ask this again. So, then you mentioned the turnaround was really quick, right? And I’ve read that the decision to go-- to do a native SDK was announced by October of 2007. So, between WWDC, which was July, and October, that’s, what, four months-ish.

Williamson: That sounds about right.

Kocienda: That sounds about right to me because then we did it the next year, right? We did it in 2008. So, it was this incredible rush to get the native API together.

Hsu: Yeah, the beta version of the SDK was available in March, which is pretty quick.

Kocienda: Yeah. It was fast.

Williamson: I mean, like I said, the-- we had built our own apps using native-- the native frameworks. And the APIs weren’t quite ready for public consumption, but they were pretty darn close. And they were tried and tested by, what, the dozen or so apps that we shipped with the system. So, I think we all felt pretty confident that they would be okay.

Kocienda: Yeah. And we’ve got a long-standing collection of development methodologies for making APIs for developers. So, a lot of what we did was so, you have UITableView.h. And so, we had that-- let’s say for the sake of argument, we had that in our private API for our 1.0 release. So, kind of what we did-- I mean oversimplifying, but it gets the spirit of it, is we copied that whole API to UITableView_private.h, which would never ship out to developers. And then we took those things and copied them over to make public API, sometimes changing interfaces, calling through to the old interfaces from the new one so that the-- you could get this functionality that we had on the existing system but with a cleaned-up interface.

Williamson: But it was really an extension of Cocoa. Cocoa had many, many years of maturity on OS X.

Kocienda: We tapped into so many of those methodologies and conventions of not only developing the frameworks, but for packaging them and delivering them.

Williamson: And Cocoa goes back to NeXTSTEP, which is even further. So, I mean it was very mature in terms of a development approach.

Kocienda: Very mature. We had naming conventions. A lot of things were decided that yeah, this is the way. This is the structure of things, how they should be put together.

Williamson: Pretty clearly the right thing to do. Like I said, the web development story was just a hiccup along the path.

Hsu: How important do you think it was for, especially those first wave of developers that UIKit was basically based on Cocoa? Was it a great-- did you think that the success of-- how much of the success of the App Store do you think was due to the fact that it was based on Cocoa?

Williamson: So, I mean the Mac OS X app developer community is not huge. So, I don't think any of us thought okay, we're going to tap into millions of developers because we have all these millions of apps on OS X. So, in that sense, I don't think we thought that was important. But the fact that we all knew, because we'd been using Cocoa for so long, some of us for decades, that it was going to be a very robust solution. And with Xcode being a very mature product, too, we-- it was pretty clear that apps could be written-- robust applications could be written with those technologies. And as far as the App Store and the success of the App Store, I think we were all kind of taken aback at how rapidly that took off. We all thought the App Store was a good idea. But the fact that it caused an explosion of development in third party developers, none of us really anticipated the scale of that. And I think it's awesome that we created a new platform that has flourished so much and allowed for third party development in a way that kind of-- it had stalled on the desktops. And so, I think from a technology perspective, it was pretty clear we had something robust. But in terms of anticipating the scope of what happened, no, I-- Scott didn't think that. I didn't think that.

Kocienda: No, nobody anticipated that.

Weber: The decision to have the App Store apps vetted, was that in there from early on?

Williamson: Yeah. So, the key idea was security of the phone. And it's something that we wanted to really differentiate the device. And Scott was one of the main proponents of this. And yeah, very early on, we wanted to create a sandbox for the apps to operate in and make security the most important aspect of third party development. Not on, like, OS X, right? There were lessons to learn from OS X where that wasn't the case. And rogue applications—[on] Windows, crazy. So, important differentiating feature, but not just because it's differentiating, but because it's fundamentally good for the platform.

Kocienda: It's not only good for the platform, good for the people using it.

Williamson: Exactly, yeah.

Kocienda: Right, people don't want to be managing security patches and hot fixes and updates or whatever. You just want a product that is sensible and is not working against you without your knowledge.

Williamson: I think this may be Scott's most important technical contribution, pushing for security so hard.

Kocienda: Yeah.

Hsu: I do remember that in the first few years of the App Store, there was a lot of-- among the developer community, there were a lot of complaints about the app review process and also just the difficulty of going through, getting your provisioning profile and all that stuff. Could you speak to that?

Williamson: Yeah. I mean I think that's a consequence of not anticipating the scale of everything and learning as we go along. And I never sat on the review board. I think Henri did for a while. And he hated it. So, I think-- actually, I should probably leave my comments there. I really wasn't intimately involved with it.

Kocienda: I mean it's a bottomless pit, really. It's not-- I don't think it's a job that is easy to address, simple to, "oh, we're just going to approve these apps." Well, if you're going really do-- I mean it's a big important job, and particularly, as Richard is saying, at the beginning, the scope of the success of it made just it seemingly this singularity of content.

Williamson: So, what I will say though is the extent that we were involved is we were trying to help automate the process. So, it was pretty clearly a problem. And so, things like API scans so we could look for external linkage points to private APIs and flag those applications, we, as an engineering team, can help build that set of tools. And Eric Albert, I think, became really involved in this, in the review process, and trying to build more and more tools to automate the approval process.

Hsu: I do remember also that there was some controversy over not just sort of the technical stuff like the bugs and other things or use of private API, but also that Apple was now involved in censoring content or choosing which kinds of apps should be allowed and which kinds shouldn't be allowed just from the perspective of functionality and sort of dictating what kinds of apps developers can or cannot-- Could you speak to that?

Williamson: I don't-- never was involved in anything like that. The one story that really somewhat relates-- it wasn't apps. It was about eBooks. And we had-- we launched eBooks to great fanfare. And there were some textbooks. And one of the textbooks that was going to launch was a biology textbook that put creationism side by side with evolution. And I normally don't get involved in politics, especially within the company. But I just-- this was just-- totally runs against everything I believe. So, I told Scott, "Look, this is totally not acceptable. We need to do something about this." And the issue came all the way up to the board. And it turns out Bob Mansfield is a very religious guy. And he really wanted to include this textbook. So, there was a huge debate about it. And the outcome ultimately was we didn't include the side by side comparison. So, that's the only thing in terms of censorship that I've ever really come close to dealing with at Apple. But for the most part, I don't think there is. I mean other than explicitly objectionable material that's pornographic, say, I don't think I ever heard much discussion about censoring things.

Weber: But explicitly pornographic is a good portion of the Internet, so--

Williamson: That's true. But Apple has like a Disney kind of--

Weber: Right, no, but I mean that's a very significant decision.

Hsu: But I think that there was an outcry I think because I do remember there was a decision like sort of vague-- like things like Maxim or things that were going in that direction were sort of allowed at one point. And then Apple made a decision that nothing-- none of this stuff is getting through. But then they made an exception for Playboy. And there was an outcry that somehow the big players were allowed to get through and all the other--

Williamson: Phil Schiller ended up doing a lot of the-- he was the driving guide behind most of those decisions. So, it wasn't really engineering driven. It was marketing. And Phil has done a bang-up job of being the guardian of Apple's image. So, I think he-- it's his call to make. Yeah.

Hsu: So, let's move on to the other engineering work that you needed to do for 2.0. At what point did you do cut, copy, and paste? Was that for 2.0, or later?

Kocienda: <laughs> No, I don't recall that it was. No, I think it was two full years before we did cut, copy, and paste. So, it was after the API release. Then we--

Hsu: So, iOS 4, was it?

Kocienda: No, it would have been 3. Was it actually called iOS 3 that was-- because I think that might have been the first one where we called it iOS.

Hsu: Oh, right because of the iPad.

Williamson: Right, what to call the operating system?

Kocienda: Yeah, what to call the operating system? Yeah, iPhone OS. Yeah, but yeah, cut, copy, and paste was like a third rev thing, first release, then API, then cut, copy, paste.

Williamson: It's kind of remarkable when you think about it, such a fundamental feature, but yet, we got to three releases without it. And it didn't really slow down the acceptance of the phone.

Kocienda: No. I think it was a good choice. We were pretty busy. And we prioritized.

Weber: But so why do it then?

Kocienda: Why do it then?

Hsu: Because it seemed like it was pretty deliberate that you didn't want to do cut, copy, and paste.

Kocienda: No.

Williamson: No.

Hsu: No?

Kocienda: No.

Williamson: Time.

Hsu: It just-- you hadn't gotten to it?

Kocienda: Just time.

<laughter>

Weber: Oh, okay. There was no--

Hsu: Because it sounded to me like you had made a design choice to not do it.

Kocienda: No. No. Just I'm slow. I was working on the tech system and there simply wasn't time to do it and get it right. And then, the second year was all about taking the current tech system and putting an API around it.

Williamson: Let me say a little bit more about that. So I think, clearly, cut/copy/paste is a very important feature for a modern UI, but the challenge we had was the screen was really, really small. Text selection is tough on a small screen with your big, fat finger. So it wasn't just we didn't have enough of Ken's time to do it. It's we didn't have a really good design for how to make that text selection. So what we ultimately did with the magnifying glass was it took a lot of iteration to get to that point and we just didn't have a design that worked for the first couple of releases. But it's not that we didn't want to do it.

Weber: But had you been--so when you said, "It was hard to come up with a good design," was anybody-- early on were people even thinking about it?

Kocienda: Sure. I was thinking about it all the time, yeah. I mean, and to be clear, what I call the loop, the magnifying glass was available in the first iPhone to place the insertion point. But then, using that as a means to make selection, which just then like the "lollipops" and then the menu, the call out menu that appeared above selections and then-- I mean cut, copy, paste itself, I mean, there was a-- you know, we needed to implement the pasteboard, which
was the underlying API that would go and store, that you could put these data types onto the pasteboard and the pull them off later, but that largely existed from OS X. But it was actually the widgetry and the user interaction model, trying to figure out how to do this with touch and then how to present the menu options since there's no menu bar and there's no hardware keyboard for command equivalence. So that was the design problem.

Williamson: I remember a lot of time being spent on that--

Kocienda: We spent a lot of time on that.

Williamson: --for that release. It was huge, a lot of iteration.

Kocienda: Sure.

Williamson: Getting that to work right.

Kocienda: Yeah. There was a lot of iteration. Because, you know, you ask yourself the question. It's like, "Well, so I make a selection. Where are the commands?" Well, once you make a selection, the menu will show up. Where does it show up? What if the insertion point is on the top of the screen? Well, it can't show up above, so it's got to show up below. It's going to point to it. Where does it point to it? What if it's over on the edge of the display? I mean, all of these questions to try to make a system that works and seems like, "Oh, well, yes, of course it works that way" was a tremendous amount of work. I mean there was easily six or eight of us. There was me, there was you, Scott [Forstall], Henri [Lamiraux], Greg Christie, Imran [Chaudri], Bas [Ording] were the people primarily involved in that, and lots and lots of back and forth. I'd go over to see Bas and Imran all the time separate from the meetings, tossing around ideas and making demos. And it's actually-- it's kind of funny is I recall that Bas and I came up with the idea for the menu and the behavior on the same day. I walked over-- independently. We were just kind of converging on it and it's like-- but weren't quite there yet and I went, like made a demo and brought it over, "Hey, Bas, I've got something to show you?" He's, "Oh, Ken, I've got something to show you too." And we demoed it to each other. It's like-- and so, yeah, we came up with the--

Weber: So there were no significant like dead ends or totally different schemes for doing it or anything.

Kocienda: I mean, there were ideas that didn't quite work. I mean, one of the things that didn't-- that I remember Scott actually showing him and having him not like was that the Lollipop handles, they're very, very small, but the active area for them is quite large. So you get this little pinpoint, but there's this really quite large circle right around that pinpoint where you could reach in and grab the thing and move it. And so, one of my earlier versions was to actually show the whole active area with something that wound up looking kind of like big commas, like two big commas showing you that you can grab this whole area, and Scott hated it. He thought they looked like kidney beans. He's like, "What are these things?" It's like, "These are ugly. These are terrible." And then it was Bas who actually came up and said, "No, no, no. We're just going to have these "lollipops." We'll use Ken's active areas, but here's how it should look." So again, this idea of give you what you mean rather than what you did and giving you a target that looks, looks very, very small, but is actually quite large and comfortable to use.

Hsu: Hmm. Once you came up with the design though I guess was it fairly straightforward to implement? No?

Williamson: Keep in mind those text fields are like little web views.

Weber: Oh, right.

Kocienda: They're little web views. Yeah. So just going and-- no, it was a year of my life, and a bunch of gray hair. And then, I actually wound up needing help from Brad Moore to implement selection actually in web pages. So again, at this time, all of the text system was essentially web content, but the iOS side of that, either single-line text widgets or multi-line text views, was this very compartmentalized custom web content. But then, the open web also needed a selection model. And so, it just got to be so difficult that I couldn't do both of them by myself, so had another bright guy come and help with the open web side while I did just the iOS web side.

Hsu: Let's go onto talking about the iPad. So a lot of work-- you guys had to do a lot of work to support the iPad, especially you on the keyboard side?

Kocienda: Sure. Yeah. One thing that came along the way, in this same general time frame was a-- I rearchitected—I wrote a language, a computer language to describe keyboards rather than having them be-- kind of every one was a custom one-off. So I made a system where you could just describe declaratively what a keyboard should look like and what keys it should have and what the dimensions of the keys are and what keys are next to other keys and all of that. And so, then when the iPad came along, I came up with some more descriptions for how the iPad keyboard should look and what it should behave like.

Weber: And then, what were the key differences between that and the iPhone?

Kocienda: There were no pop-ups on the keys. The keys just lit up, changed color rather than showing a paddle. Naturally, the different layouts. What keys are we going to show? There was a big decision about whether we should use sort of a more standard laptop keyboard design and show a lot of keys or have something that was more like an iPhone where the keys were bigger and just kind of the top qwerty, top row, across the top to get something that might be a little bit more comfortable to touch type on, particularly in landscape. And so, there's just a ton of iteration, but it's the same thing as always. We just did a ton of iteration. And so, one of the things this keyboard language made possible was we made it very, very quickly for anyone to make changes. I could just very, very quickly change the programming code and recompile it and you could look at the new keyboard whereas the other way was very laborious to go in and design and make changes and hook up the behaviors appropriately.

Weber: And I assumed that helped with internationalization as well.

Kocienda: It was part of the idea as well, yeah, is to make a system that could start implementing the Cyrillic keyboards, Japanese keyboards, handwriting keyboards, [not] handwriting "keyboards," [but] you know, handwriting input methods.

Weber: Oh, like defining the active space for that.

Kocienda: Defining the active space, but then it also did have keys for accepting or deleting and so forth.

Hsu: I remember, was it this release or later there was a version of the iPad keyboard that sort of was split so that you could easily type with your phones.

Kocienda: Yeah. There was one follow who did that primarily. Yeah.

Williamson: At that point, we were growing out the keyboard team. So Steve Swales managed that team and there was maybe six people working on keyboards.

Kocienda: Yeah. But at that point-- really by the iPad, I stopped working on keyboards as my chief responsibility. This team took over, but I came in with some of the design ideas for the iPad, but I was no longer maintaining the code every day.

Hsu: Oh, okay. So were you still working in Richard's team or had you split off or--

Kocienda: I think by that point, I started reporting to Henri instead. Yeah. Promoted as you might say.

Weber: I think you talked last time. Swipe was never-- he never wanted to support Swipe particularly.

Williamson: Well, so we had lots-- I don't know if you were involved in these discussions, but we had lots of meetings with Swipe.

Kocienda: Yeah. I was in one in the conference center, outside IL-1. We had one meeting, pretty formal.

Williamson: But, yeah, ultimately, we decided not to do anything with them.

Weber: Why not?

Williamson: They really, ultimately, we believed, didn't have anything much beyond what we already supported. The idea of actually drawing a graph to represent a word was technology that wasn't particularly innovative. And in fact, we kind of had that technology already with what Wayne had done. So it just didn't make sense. And we also, at that point, we entertained the idea of opening up the system to third party keyboards.

Hsu: Okay. Already. That early, which you did many years later.

Williamson: Yes.

Hsu: Yeah.

Williamson: So it was a debate internally. Scott was ambivalent about it. I was busy building the keyboard team and thought that we should because ultimately, there's many ideas that we're not going to be able to implement ourselves. So those discussions were happening.

Kocienda: Part of the discussion that I had, I remember one time with Scott one-on-one, was security. Security was an issue. That if you have a third party keyboard running in process--

Weber: Keylogging and--

Kocienda: Keylogging. And so-- and then, when third party keyboards actually came out of the system many years later, keyboards had been separated out into their own process.

Williamson: Which is true. Plug-ins in general, we know the security issue, which--

Hsu: Oh, because there was a whole plug-in architecture by that point.

Williamson: Yep. Right.

Kocienda: And a whole extension architecture that took View Services, this whole technology for giving an external process a rectangle in the space of another process and not letting that potentially unvetted or not native, not Apple developed process, not giving it any access to the app content.

Weber: But early on you couldn't-- yeah.

Kocienda: The keyboard ran in process. It didn't even have its own thread. It ran on the main thread.

Hsu: Oh, wow.

Kocienda: Yeah.

Weber: No way to defend again.

Hsu: I mean to some extent that speaks to sort of Apple's sort of long-term philosophy of incorporating features or opening things up. Is that-- developers may clamor for some feature, something, but Apple wants to make sure that there's an infrastructure in place before they let developers do that. Is that something that is pretty clear in terms of when you're on the inside that--

Williamson: Oh, yeah. Absolutely.

Kocienda: Oh, yeah. Yeah. We tried never to just do things without thinking through what the implications might be.

Williamson: One thing about delivering a framework, an SDK or an API, is once you put it out there, it's very hard to take back. So the more and more apps that link against that API, if you break it or change it, all of those apps break, or if you release something that's insecure, all of those apps have that insecurity. So I think the conservative approach isn't in the genes of Apple in terms of releasing frameworks and API. It used to be actually. I'll take that back now. The last couple of years, it's like a proliferation of frameworks. It's like everything's a kit now. Every WWDC, there's like three or four kits that are released.

Kocienda: No comment.

Williamson: No comment.

Hsu: You're not there anymore, so you can talk all you want.

Kocienda: No comment, but we can leave the camera rolling. If you were to turn the camera off, but--

Weber: Do you want to?

Kocienda: No. No, it's not necessary. I'm half-joking.

Williamson: But actually, this idea of "what is a kit?" we actually debated. There was a lot of considerations. Is something justified being in its own kit, or should it be part of UIKit or Foundation or some other framework? And our approach was-- this goes even back to Cocoa, to be very, very conservative about introducing a whole new framework. That's, I think a philosophy that's changed in the last few years at Apple.

Kocienda: Yeah. As you were going back, even before the iPhone, when we were doing the Safari WebKit, we had the notion of "WebFoundation" where we were going to be doing URL loading in a separate framework. And that actually wound up getting rolled into Foundation proper. There was a whole big long set of meetings about going in and adding this whole new capability to Foundation, whole new set of APIs. I had people who would like give me mean looks in the hall. “How dare you go and add so many lines of codes to Foundation.” And part of it was in gest, but part of it was real, this real like technical conservatism. “You added all of this, all of these lines of code to Foundation and now we need to maintain it.”

Hsu: This was on OS X?

Kocienda: That was on OS X.

Hsu: Oh, so like NSURL.

Williamson: This was WebKit-based.

Kocienda: NSURL. Well, not NSURL, but like-- NSURL was there, but then-- but NSURLConnection and NSURLRequest and Response and things like that. And adding those things to Foundation was a big decision. And ultimately, we decided that it was the right thing, but I’m kind of half in gest, but just to be clear is that there was a strong technical argument against it because Foundation is something everybody gets. And when you go and you add that #include, now you’ve gone and given people this larger API surface area that they need to work with once they go and include Foundation.h.

Williamson: This gets to the culture question. This is a cultural thing, to be resistant to introducing new APIs. And I think it’s kind of unique in that Apple culture, and we carried that forward into iOS too.

Kocienda: It’s just you better be sure. That’s all that it is. It’s not a conservatism from the perspective of what we want to make available to people or what we want the platform to do. It’s just that we know that there’s a responsibility to then support that thing way, way out into the future, and that sometimes you can’t really see the full implications of what this API might mean. And so, err on the side of giving away less.

Williamson: It’s much harder to deprecate an API than to introduce an API. Takes nothing to introduce an API.

Kocienda: Or to give developers a little bit of a taste of an API and have them go out and use it and say, “Darn it. Why doesn’t it do this and this and this?” And then you can you say, “Okay. Well, yeah, maybe two out of those three things we might want to support and that third one is maybe just a little too specialized or a little too-- that will be left as an exercise for maybe an independent developer to go and add that as a private thing for themselves.

Hsu: Yeah, I remember that too. My recollection was that a lot of that culture had been set by people like Bertrand Serlet, Ali Ozer.

Williamson: Bertrand, I think by-- he was the king of the API review list until he decided to take on a more management job spin.

Kocienda: Yeah. I mean, I had a discussion with him once about the interface for NSDictionary. I had lunch with him and we were just talking about. It's just like "get" and "remove," that's it. That's the API for a dictionary. You put something in. Or say-- three. You put something in. You check something is there and you take something out. I mean that's it. That's all that it is. I mean count how many things-- you don't even actually need that. That was made as kind of an additional API, but it's just like the act of-- core access is just the only things that are in the core API.

Williamson: In a sense, it's just like Apple provides simplicity at the user level, at the development level, that same concept applies to just what's enough, but no more.

Kocienda: Just what's enough, yeah, but no more.

Hsu: And to try to only have one way to do things.

Williamson: Yes.

Kocienda: One way to do things.

Williamson: Which, again, on the phone now is like three ways to do things.

Weber: But along those lines, I mean there was a real decision in the iPhone not to do more conventional kind of files, folders being able to share documents between applications easily.

Kocienda: It's too hard. It's too hard for users.

Weber: But I'm thinking like kind of the other extreme was the Newton had, what, the soup concept or something where everything could be used in multiple apps—almost-- you had no applications, but plug-ins that could modify parts so that everything is mixed together whereas the iPhone is almost the opposite, kind of very much compartmentalized. So what's the thinking behind that? As you said, so users, is that--

Kocienda: Well, to me, there's a bunch of things to unpack there. I mean, even start by saying, the file system is a detail that's unnecessary a lot of the time and that most people who are normal in that they're not computer scientists nor app developers; you know, 99.9, how many nines, percent of the people just want to be able to use an app and not have to worry about this whole abstraction of a hierarchal file system. It's just not necessary. It's a thing that we thought we needed from older paradigms of computing. With the iPhone, we got a clean slate and we said, we're going to do without out. We're going to see if we can make a system that is more people-centric and just get rid of this whole big complicated thing that maybe is more trouble than it's worth for people. And the thing is I don't want to give the wrong impression. It's not that-- we don't think that people are dumb. People aren't dumb. They're buying iPhones, right? We think they're actually pretty smart, right? It's just that most people aren't interested in the device as a computer. They're interested in the device as a message sending system or a photo viewing system or a navigation system. What do you need a file system in order to help me show my photos of my vacation? It's just unnecessary.

Weber: Because it seems like there's two approaches, both have a lot of resonance for the iPhone, both saying it's a general purpose computer as opposed to say the iPod phone, that this is a real general purpose computer, but also one that is very kind of honed for particular uses. It's not a general purpose computer in the sense we've come to experience that for a PC, personal computer.

Kocienda: For the better.

Weber: Right, but I mean how do you see it then, as a kind of new sort of general purpose computer or a user-centric general--? I mean it seems like it's almost-- you were pushing toward a new kind of animal.

Williamson: Well, so, I think there is tension here. I think the primary uses of the phone are for a relatively small number of applications that don't require documents. But then, once you kind of move over into content creation, necessarily, you need to start thinking about documents a little bit. But what we did on the phone is err towards the other side where it's not about content creation; it's not about documents; it's about simplicity of use. But that doesn't mean you don't still fundamentally have a general purpose computer underneath that that could at some point in time provide the notion of documents in a file system as we're seeing now evolve with the iPad. But primarily, on a small screen like we have with the iPhone, it's not about Photoshop. It's not about content creation. It's not about writing essays. It's really about consuming content. This is the way we used to talk about it. Ken is completely right about that. But I think it's not clear cut. I think at some point, we do have to migrate towards a notion of having files or documents because when you're creating-- or editing a photograph, you want to perhaps export that somewhere, so there's tension. It's not clear cut, but I think we erred in the right direction with the iPhone for the first many releases. It's just simple to use. It's just dirt simple. You don't have to worry about complex concepts like a file system. That's-- for 99-percent of the people, as Ken said, that's the right thing to do. But now, we're starting to increase the vernacular of touch interaction and the complexity of applications is starting to increase. So I think we're going to have to pull those notions back very carefully, but, yeah.

Hsu: You mentioned the iPad specifically. The iPad is one of those things where it does make more sense as a content creation device and yet, in the first couple of versions of the iPad, that wasn't so straightforward, right? And so, of course, we're now seeing the concept of files come back, but it's many, many, many years later. Why did that take so long do you think?

Williamson: Well, like I said, I think on a small screen, content creation isn't what you're primarily doing.

Weber: The iPad is not in an especially small screen.

Williamson: It's pretty small.

Weber: Bigger than the first 20 years of PCs.

Williamson: Yeah, well, we didn't have an external keyboard with an iPad for a long time, right? So you're not going to write a dissertation on a cramped keyboard that's virtual.

Kocienda: You hear stories about people [who] wrote novels on their iPhones.

Weber: Oh, even nine key, Japanese cell phone novelists.

Williamson: That's the exception, not the rule.

Kocienda: The exception, not the rule.

Williamson: So the iPads have gone bigger. I think as they've gone bigger, you can actually now think about multiple apps and what not. But it's an evolutionary approach to get there. I think we definitely, like I said, erred on the right side with the small screen devices, trying to really simplify the user experience. Like we said, cut, copy and paste didn't come out for three releases and it didn't hinder the explosive growth of the iPhone.

Weber: But presumably the early iPhones you saw-- the average user would have a computer. I mean this was going to be an accessory of some sort. It was not going to be their only computer in the world whereas as it's developed, that's becoming more and more the case. I mean iOS may be the only operating system people use for their whole computing experience.

Williamson: Sure. Yeah. But how many people do you think buy an iPhone with the intent of writing a book on it or--

Weber: But that's what I'm saying. Were you consciously thinking at the beginning "This may be a general purpose computer, but it's still going to be an accessory to probably the computer people would do that creation on?"

Williamson: So maybe this is a question of definitions, what a general purpose computer is. When I think of general purpose computer, I think of a device that's capable of running a multitude of applications, not necessarily something that looks like a PC. So the operating system on the iPhone is a general purpose operating system that allows multiple apps. That doesn't mean that the interface has to replicate what you see on a desktop PC. So I think it may be a matter of semantics. When you say "general purpose computer," you think it's got to replicate what a PC does.

Weber: No, no. But I mean, technically, your dumbest feature phone is a general purpose computer. It's just --

Williamson: Well, yeah, but this gets back to the discussion of what an iPod was versus what we imagined it to be. So a general--a feature phone doesn't run a general purpose operating system that's capable of running any apps. There's typically there's firmware in there that's dedicated to the specific functionality of that flip phone. With iPhone, that's not the case as we all know. You can pretty much write anything on an iPhone that--

Kocienda: It's UNIX.

Williamson: UNIX, yeah.

Kocienda: UNIX with this whole graphics package on top and a whole bunch of frameworks.

Williamson: So there's two levels of what a general purpose computer is, the interface and then the underlying operating system. And just because we think or I think that the phone is a general purpose computer doesn't imply necessarily [that it] have the same UI.

Weber: Sure, the same features. But I mean now that the evolution does seem to be that in some ways iOS and Android in parallel are kind of taking over the functions that used to be on personal computers, I mean did you guys foresee that or was that too forward thinking necessarily?

Williamson: I think "foresee" is not the right word. But I think explicitly thinking about what the functionality is of the two devices. What is the gap between a PC and an iPhone or an iPad, and where do they meet? Clearly, it's a spectrum from a device that you want to be super simple to a device that's kind of complicated to maintain and use, but has all this super rich content creation application technology. I don't think there's a clear line and a clear set of answers. Microsoft is going one approach. They're trying to combine a PC and a tablet into one thing. And with Apple, we've always erred towards dumbing things down for the end user for the majority of the cases, not dumbing down, but simplifying is a better word.

Kocienda: Simplifying is a much better word, yeah.

Williamson: But now we're-- Personally, I'm a little concerned about the direction things are going. Now with the bigger iPads more and more towards a PC-- I think Greg Christie, Scott, me and other senior people on the iPhone team, we didn't want to replicate that ever, a desktop PC. So there's tension. I think you can do the vast majority of your computing tasks without a file system and without everything you expect on a desktop PC. But sometimes you want to pull out Photoshop and Adobe Illustrator. Sometimes you pull out Xcode. Obviously Xcode isn't going to run on an iPhone. So there's tension and there's no clear answer. But the philosophy we had was towards simplicity and single application at a time, a single window at a time and, yeah. And I think it's actually obviously proven to be successful, with, what, 600 million-plus iPhones out there.

Kocienda: Yeah. So Greg Christie was the manager, the day-to-day manager of the Human Interface team; so the team of designers working on software. He would sometimes use the word "computery" to describe something that isn't good. It seems too much like a geek would love it rather than a person. You know what I mean? We tried to design these systems, both the Mac, but especially the iPhone and even the iPad, for people who don't care about the thing as a computer.

Williamson: In fact, if you ask 99-percent of the people, you ask them "is the iPhone a computer?" they'd say no.

Kocienda: No.

Williamson: That wouldn't make any sense. It just kind of does photos and plays music.

Kocienda: And social

Williamson: Yeah, messages.

Hsu: But clearly over time, as iOS has progressed, more and more "computery" things have gone in, right?

Williamson: Yeah.

Hsu: Like cut, copy and paste, that's one thing. Multi-task-- multiple applications. That, came in, what, iOS 4 was it? And then extension architectures. Now there's iCloud Drive. Like there's a progression towards more and more PC "computery" things.

Kocienda: Well, if you're asking me if these things are good-- I mean, you're saying that those things are there. It's undeniable that those things are there.

Hsu: Right.

Williamson: I don't think there's a clear answer. I think some people say OS X and iOS should merge and it should be one operating system and things should scale from a phone to a high-end PC, but I'm not sure that's true. I think, again, for the underlying operating system, yes, as much as possible should be shared across the platforms. But in terms of the user experience, I think they're fundamentally different. I think a touch-first interface is different than a keyboard/mouse-first interface. And what that implies in terms of how you interact is huge. Mice are great for multiple windows and selecting large blocks of texts. And keyboards are great for fast text entry. Touch, especially a small screen, radically different. So I'm not sure I like the trend either of moving towards putting more PC features into iOS.

Weber: But back then, you were certainly not predicting it or considering it desirable to go that way.

Williamson: That wasn't, I think, in most of our minds. No, not at all. We didn't--

Weber: So if you did think about the future, you were thinking that it would remain a parallel track of something, an easier to use communication device, social device that wouldn't necessarily merge with normal personal computer functions as we think of it.

Williamson: Well, I think it's kind of limiting to say "a communication device." I mean, the way I think about the phone is that it's an extension of our intellect and it's capable of doing radical things, not just what we did at launch, but what we can do today and what we do beyond. Certainly, it's not about emulating a PC, but having a device that's connected all the time to all of the world's information is game-changing and that's kind of what we need to focus on, a device that knows where it is in space and a model where we know about what's in space, and a way of interacting in that space. That's what the phone is about. It's not about managing file systems and that. It's really about kind of augmenting our intelligence.

Weber: But then how can you leave creation out of that? You're saying that it downplays the creation--

Williamson: Well, you don't have to have one device that does it all. I think that you could certainly-- I still have a desktop that I use all the time. You can still have a desktop as well as something like an iPhone, but I have my iPhone with me all the time. I don't have my desktop with me all the time. I'm not writing novels or code all the time. But in terms of what something can do to help me every second of the day, that's what the phone is.

Weber: Okay.

Kocienda: You know, and I do think that probably a lot of people, for a lot of people, the majority of their digital creations are getting posted on social networks and they're creating it on their phones. So they are creating all day every day.

Williamson: But a tweet in itself is not really--

Kocienda: Well--

Williamson: It's a different spectrum.

Kocienda: But the thing is the services and the human activities and the capability of the devices have merged. It's like people are-- our culture is evolving. I mean, there's absolutely no doubt about it, has evolved to meet what the phone delivers to them. I just today was going for a casual walk and it's this beautiful morning and I just happened to notice this guy who had a chainsaw down on the sidewalk next to him and I heard a conversation in the background and there he is looking at his phone. I guess he's waiting to have somebody come and describe to him what branch they want cut off the tree. And while that's going on, he's just going to be standing there by himself looking at his phone. Ten years ago, 15 years ago certainly that wouldn't have happened. He probably would have just been standing there with his hands in his pockets maybe admiring the weather. Now, we can debate whether this is good or bad or otherwise, right? But the fact remains that the device has become this element in our culture and that, yeah, a lot of people are spending a lot of their time not only consuming, but creating in that space. It isn't a traditional PC task, but kind of grading it on that scale I'm not sure is necessarily right.

Williamson: The natural extension of the iPhone is a chip in the head with some kind of projection display and the connection to the world. Maybe when we have holographic displays and you can manipulate things in space, that's going to be a new creation system, but I think we're a long way off from that. But the idea of something that's always connected, that's fundamental to the iPhone. It's different than a PC. And as Ken says, everybody now is connected all the time, even a guy chopping down branches on a tree. You've got it in your hip pocket and it probably stays there all the time, so do you and so do I. That's the fundamental thing that's changed the world with the iPhone is something connected to you all the time that's connected to the Internet and the Internet means the rest of humanity and the rest of humanity's information.

<break>

Hsu: Well, let's talk about the whole-- let's start with maybe the whole decision to get off of Google and then that whole-- go through that and then the rollout until, I guess, with-- until your departure, I guess, would be a good place to--

Williamson: So 2009, Apple's relationship with Google started to sour. Steve was getting more and more concerned that Google was going to be a direct competitor and was leveraging a lot of the work we've done in their own designs. It became clear to me and Scott that knowledge about the world was fundamental to a mobile device and knowledge about the world meant a mapping system, understanding places in the world and understanding your place in the world and what's around you. Maps was fundamental to that. And so, realizing that we had a very thin layer on top of Google's services and we were very much dependent on Google at the same time that they were developing Android became a concern. And so, in 2009, we started to think about what we could do to replace Maps. And there was a lot of concern about the magnitude of this task, which is how do you map the planet and how do you understand everything in the world that's out there? Google spent a decade working on it. But Steve said, "Let's go ahead and put together a plan." So in 2009, I visited a company called Playspace in LA and kind of like what we did with WebKit where we started with a small kernel, I thought we could use the kernel of what Playspace was developing as a basis for a mapping system. And we acquired Playspace and Jen [ph?] Waldman came on as part of that acquisition. Then in 2010, Steve kind of created a war with Google. He called Google kind of the evil empire. And I remember the famous Town Hall meeting. He said, "Google is evil" and then he gave us the go ahead, full steam ahead with this plan to build a mapping service in two years or less. I look at what I've done in my career, there's kind of four big things. There was NeXTSTEP and then Safari/WebKit and then the iPhone and then this mapping platform. It's as big as building the software for the first iPhone, huge, but a different set of challenges in that we wanted to not just meet what Google was doing. We wanted to surpass it, kind of look ahead at what we could deliver in a couple of years. There were a few things that were important. One is that we wanted global scope. Everywhere we shipped an iPhone we wanted to have this service available. We wanted to have much more sophisticated rendering of the map content. So Google up until this point had been developing raster tiles that were pre-rendered and we wanted to create an environment that was based on [Open]GL that would let us have dynamic zooming so you could have smooth zooming. We wanted to render 3D objects in this space, and then it had to be kind of novel in terms of its feature set so we could be comparable to Google. So it wasn't clear how we could do this in a small number of years, but what we ended up building was something that stitched together data sets from 21 different companies. We relied on Axiom for POI database or point of interest database. We ended-- I ended up buying C3, which is a company based off some work that Salv [ph?] had done to do flyover, the flyover feature. For the flyover feature, we had to build basically a fleet of contractors to fly planes and helicopters around the planet. It was a huge endeavor. We can drill down in many different areas in this. But for me, the upsetting thing about all this is how it was received. There were just a few flaws that were very visible, but ultimately, the technology was incredibly good. The data was less than perfect. Kind of quickly getting to the QA thing, the QA team who ran QA for the iPhone was very similar to the QA team for OS X. That team-- the way you qualify an operating system is very different than the way you qualify huge datasets. We had-- and also, the QA team was outside of the engineering organization, so it didn't report into engineering at all. So a dataset that spans 81 countries requires that you actually look at some of this data and there's 81 countries. We had eight people reporting into Kim [Vorath] around the world that qualified the data and said that it was good. So this kinds of gets into this tension of-- now I'm talking about things that are a little personal and close to the core because I was fired by Eddie Cue for failing to deliver a great mapping platform. But the fundamental flaws that we had, we had the flyover issues where there was kind of the edges of flyover. There was kind of a poor blending effect. So we had things like the Brooklyn Bridge melting. That's something we fixed very quick with a technical solution. It didn't require large data kind of field analysis. And then routing. Routing was actually really good, but based on poor data. So we didn't have anybody in the Australian Outback kind of looking to make sure our routes to the Australian Outback were correct. And POI data was just out of date. We used an aggregate, like I said, Axiom, to do that. And we had basically no QA when we shipped. The QA that we had, Josh Williams, gave us the greenlight. They said, "Yeah, it's good." And my team, we tested around California and we tested when we were on vacation, but we couldn't test beyond that. So huge, huge press backlash after we launched. To this day, I'm still sore about it because I think it was a Herculean achievement to build this in, basically two and a half years. It's gone onto be a great success and all the foundational technology we built is really good. As I said, I think understanding the world around us is fundamentally important for a mobile device. And what we think now about things like driverless cars, all of that infrastructure and technology is fundamental to supporting driverless cars, which also need to-- they're basically a mobile device, right, that needs to know about the world around them. So, that's my ten minutes on Maps.

Kocienda: I did a little bit of work with Maps, but I was largely outside the effort. The thing that-- of course, I was there and of course, I was still meeting with-- seeing Richard regularly. The thing that surprised me at the time was-- comparing it to the experience of Safari, Safari in some ways has-- when we did Safari/WebKit has the same problem is that you've got the web, this huge dataset that you're trying to QA, right? And you're going to have this app, which is going to give you this view on this dataset, which is totally out of our control, right? But the thing is when Safari was released, it was a beta. We didn't say, "None of your other web browsers will work anymore." If you try to double-click Internet Explorer or iCab or Netscape or whatever it is that you still have, the operating system would [not] go brr and wouldn't work. So it was kind of a soft launch and Safari when it was released initially was beta. The "beta" was on it.

Weber: How was it--

Williamson: Safari for OS X, not for iOS.

Weber: So how was it decided to do the Maps different?

Kocienda: I was absolutely not involved in the decisions.

Williamson: So it was a difficult decision. I was involved. Phil was involved and Kim was involved through Josh. We had a number of different proposals. The proposal I was advocating was we keep

Google and we have Apple Maps on the store, like several of the other apps that we had shipped and you could download that. And then as it matured, we could replace them. In other words, to actually market as beta, but that was a little weird to have both Google Maps available and Apple Maps as a beta, two apps doing the same thing. So ultimately, I gave a great demo at Top 100. Everybody's there and everybody loved it and there was a lot of excitement around that. Phil was very excited about it and Scott was super excited because he was using it all around Cupertino, and it worked perfectly. And-- but ultimately, the QA team, which has folks internationally, said, "It's good. We're good to go." And so, based on that, we decided to launch. And I think you're exactly right. If we had a different kind of launch path, the outcome would have been very different. But I think for Apple, it's hugely important to have this platform, to have ownership of it. And it needed to get out there, just the way it got out there was horrible. And it led to Scott being fired, and me being fired, and kind of the collapse of this culture that we'd built. So, it was an ignominious end to a long-- two long careers.

Hsu: How did you-- how were you tapped to lead that project?

Williamson: Steve said do it. And I had run the Maps client. And I was very interested in doing it. I'd also been running Siri. So--

Weber: Oh, really?

Williamson: So, there was this-- you know, Siri has a bunch of backend services. And I had kind of saved that from disaster. And Scott said, "Okay, why don't you take a look at this, too?" And I wanted to do it. So, there's a whole set of stories around services in Eddie Cue's organization versus what we were building under Scott's org, under me. And you know, it's going to take a long time to talk about that. But the-- and it's a lot of internal politics. But a lot of good results coming out of that. So--

Weber: I didn't realize--

Hsu: Yeah, talk a little bit more about Siri because we--

Williamson: Sure, Siri was-- we acquired a small team, Adam running the team. And the best-- what they had was a demo.

Weber: But Tom Gruber was already--

Williamson: Tom Gruber, yeah, he was on-- he reported to me.

Kocienda: Tom Gruber--

Weber: He was already within Apple, bef-- no.

Williamson: No, no, he came over with Siri.

Kocienda: No, no, no see it was Dag Kittlaus, Dag and Adam and Tom Gruber were the three founders of Siri.

Williamson: Dag Kittlaus, Adam Cheyer--

Weber: No, and I know the three of them, but okay.

Williamson: So, Adam reported in to me. And the-- what we acquired was-- it was a demo that worked great for a couple of people but wouldn't scale to our user base. So, we had this challenge of taking what was-- there was a lot of smoke and mirrors behind the original Siri implementation. This notion of AI, it wasn't AI. It was more like remember ELIZA?

Weber: It's true, I've heard Tom's story. But for some reason, I associate him with Apple now I guess just because he stayed, and the other guys left.

Williamson: Yeah.

Kocienda: Yeah, yeah, the other fellows left.

Williamson: So, the other guys left, did Viv, and Samsung acquired Viv.

Weber: Right.

Kocienda: Yeah.

Williamson: Tom was-- of that acquisition, he was probably the best guy that we got, honestly. It was pretty troubled from the beginning. But we-- you know, the concept was brilliant. We all obviously think that assistance, and voice input, and voice interaction is fundamentally important to these devices. And Scott made the acquisition with Siri with very little vetting and based on the demos. But we had to take that and make it scale. And that's an engineering problem. It wasn't really-- anything beyond that is how do we get the service to be-- how do we scale them; how do we make the software not crash. It was a hot mess. And so, I helped get that in order. And as far as the fundamental features, Tom and Adam-- totally, it's their thing. And I tried to build out the team, too. And getting back to secrecy, this was an issue with the Siri team is that we clearly needed more AI expertise because what they had was a simple keyword matching, limited domains, no NLP really. And so, we needed to augment our team with some real expertise. And a lot of the folks that we were interviewing wouldn't come to Apple because they wanted to be part of the research community. They wanted to publish papers. And we didn't-- that wasn't part of our policies. So, it was a challenge to try and get people. And also, a lot of people wanted to be based out of Boston. And we didn't have a development office in Boston. So, it was a real challenge. But eventually, you know Siri got to the point where it was-- had faster response times. It didn't crash. We actually got answers. And we started to think about, within the limitations of what the current Siri system was, how could we expand the domains and maybe even think about doing things across domains. You understand what I mean by domains, right? So--

Weber: Domains of expertise for--

Williamson: Domains of knowledge, like it knows about movies. It knows about sports. But you ask movie-- about a sports figure in a movie, it doesn't know anything at all about how to do that.

Kocienda: So, an example-- I actually even ran into an example just the other day. There's a movie called "Score," which is about movie scores. So, I actually asked Siri on my Apple TV, "Score." And it said, "What games? What sport?"

Weber: I was going to say, "What do you want to score?"

Kocienda: Because that word got siloed into the sports domain, it didn't know that there was a movie.

Weber: Right.

Kocienda: It didn't even think of-- it didn't even think about that--

Williamson: It is super easy to trick--

Kocienda: Possibility.

Williamson: Siri. There's no NLP. There contextualization of words. It's just keyword matching. And keyword matching takes you down one path, one domain, or down another domain. And so, you-- we actually pulled Ken in to-- for a project that didn't really go very far, but in terms of how could we make Siri not necessarily understand cross domain information, but more domains, and maybe open up domains to third-party developers, which I guess now Apple has done. But we were working on that when I was still at Apple.

Kocienda: Yeah, that's a whole other story. And that project, for me, it was a year of my life, but wound up meeting sort of a similar fate to Richard's projects.

Hsu: So, it was always the plan to open it up to third parties, but it just took a while?

Kocienda: Always?

Williamson: Yeah.

Kocienda: Well, I worked on it in 2012, from the start of 2012, January of 2012. So, that was pretty early on after Siri was available. They were thinking about, again, this kind of-- more across domains, but then also opening up domains to third parties. How can you take this utterance and fan it out, ask the total knowledgebase that Siri had, get some answers back, and figure out what was the best solution, not just take this word and send it down one path and then that's it?

Williamson: But as part of this effort, one of the things that we did is look at the data center infrastructure that the iTunes team had used for the stores. And it was eye-opening kind of the ad hoc nature of what they had built to scale up the iTunes App Store. And they had a data center in Newark, which, in and of itself, is kind of frightening because it's built below sea level and in an earthquake zone. And it's cheap rent, though. But the-- it was a hot mess.

Hsu: Newark, you mean in the Bay Area?

Williamson: Yeah, Bay Area, yeah.

Hsu: Not Newark, New Jersey.

Williamson: No, no, Newark here.

Kocienda: Newark, California.

Williamson: Yeah so, we built up inside the Newark data center at the time, alternate racks, that we actually had managed infrastructure. And like I said, it worked out pretty well, ultimately. And based on this, we thought that we could scale up the infrastructure necessary to build maps.

<break>

Hsu: So, I guess sort of the last two minutes, let's finish with you, Richard.

Williamson: Okay.

Hsu: You know, just sort of looking back on things, and-- I mean I guess there's obviously difficult circumstances with your departure. How would you just summarize your time at Apple, and also briefly talk about the work that you've done since?

Williamson: Sure. So, Apple and NeXT pretty much were my life. I didn't have a lot of work/life balance. Work was my life. And I'm really proud of the things I accomplished. And I got to work with some brilliant people and I think in ways changed the world that are undeniable. So, I love the culture that Apple had under Steve. I loved Steve as a leader and as a technologist. And I feel fortunate that I had the opportunity to work for him. It was a great time. And I was-- I had some of the happiest moments of my life working for Apple, also some of the most horrific at the end there. So, I have so much respect for so many people at Apple and what Apple did. I have concerns about Apple moving forward. I think with the departure of a lot of the old guard, and a lot of new people coming in, there has been some cultural shift. And you can see it in how the iPhone is evolving. So, I can't-- it's in my blood so much that it's-- I don't think most people think about their work life like that. So, yeah. In terms of what I've been doing, I worked for Facebook for a couple years. I helped them with the mobile app. I helped them do location tracking to better optimize advertising. And I also helped them build their POI database, nothing really radical, game-changing. But I was fortunate to be there at the right time given the stock market. So-- but towards the end of the-- my tenure at Apple, my wife had started to exhibit erratic symptoms that we found out was a brain tumor. And so, I was fired by Eddie. My wife was diagnosed with a brain tumor. It was a horrible time. But I took the job at Facebook and was there for a short while but then quit to take care of my wife full time. And then she passed away. And since then, I've been taking care of my kids. So--

<Williamson has to leave at this point>
<break>

Hsu: Okay, so I guess moving on. So, Ken, you continued-- actually, while we're on that topic of Richard being fired. How did you feel about that whole episode?

Kocienda: Oh, I felt awful about it.

Hsu: And what part of the organization were you in at that point?

Kocienda: Parallel, I was doing little projects. I mean I did some work on Maps. And I did a year on Siri. So, at the time that all of this, Scott and Richard getting fired from Apple, I was working on Siri. I wasn't reporting to Richard. I was reporting to his boss, which was still-- well, he was reporting to Scott. So, I was in-- pretty close organization. And I was seeing him frequently. So, yeah, it was pretty rough.

Hsu: So, I mean Richard mentions that he got fired by Eddie Cue. But he reported to Scott, who got fired by, I guess, Tim?

Kocienda: I can only speculate on that. I don't know, Tim presumably. He was the CEO, and Scott was a senior vice president. So, presumably it would take the CEO to fire him.

Hsu: Right.

Kocienda: But I don't know. I have no knowledge of that. I only know what I read.

Hsu: So, then it was-- because I find it curious that even though Richard didn't report to Eddie, that--

Kocienda: Well, what wound up happening after Richard-- excuse me, after Scott left, there was some period that Richard was still at Apple. And in that period-- well, I can tell you what happened to me. I gave a lengthy presentation on the Siri work that I had done for a year. After Scott was fired, Eddie was there. Craig Federighi was there. There was a room full of people. But they primarily were the two people I was presenting to to describe to them what the work that I had done on Siri [was] because I was the DRI for this domain and third-party work. And they heard what I said. And they decided to cancel the project, but I got to keep my job. And from what I know, from what Richard told me, although you would really need to talk to him to hear the details, they had a similar and somewhat longer and more drawn out process. And at the end of that, they decided to keep the project but fire the guy, kind of the exact opposite. So, for mine, they just decided to can the project and keep the guy. And for reasons that-- you would need to talk to Eddie Cue why that was, I have no idea.

Hsu: Now, this is something that we haven't really gotten into with anybody. But the whole thing with Scott, what was your view of-- from where you were, what did that whole thing look like?

Kocienda: So-- well, you know, there's-- from my point of view, I loved working with Scott. Scott gave me the opportunities that changed my life. The work that I'm proudest of, the work that I think has been-- had the biggest impact on technology and the world, those are opportunities that I got because Scott gave them to me. So, I love the guy. Is he perfect? No. Is any of us? No. And so, for me, in one of the-- Scott had a very big personality. And he was very demanding. He was fine. I had very-- I was always happy to meet the demands that Scott gave me because I wanted to do the best work that I could. And he was very demanding in a way that I thought was very positive. So, I interacted with him in a relationship of him being my manager, or in my management organization, and me being an individual contributor trying to get work done. And even though I never directly reported to Scott, he gave me all my projects from very, very early on. I would have meetings with him, and he would give me the work that I was going to do. And so, that's how I interacted with him. I don't really know how he interacted with the other executives because I was in the room only very infrequently when he was talking to them. And well, we can assume that he didn't get along with them very well, right? But the number of times, it's a small handful of times that I was in the room with executives other than Steve. And we called them exec reviews. We would have these big sort of project milestone reviews and whatever. And those were never acrimonious. But so, I can only speculate. I know what you know about what Scott's relationships-- I can speculate. And I can go on hearsay. But I don't want to speak that into a camera.

Hsu: Right.

Kocienda: Because it's not anything that I can substantiate.

Hsu: All right, let's move on to going back to the engineering work. So, one of the things you worked on in iOS 5 was multitasking gestures for the iPad. Could you talk a little bit about that?

Kocienda: Sure. Yeah so, one day, I-- Scott-- I had a conversation with Scott. And he said Steve had this idea to do multitasking gestures for the iPad. You know, one of the things about the multi-touch system on the iPhone is that it can handle lots of touches, as many as eleven touches. Well, obviously, we never used that many on the iPhone. Primarily, the small screen area was a limiting factor there and just also the vocabulary of gestures that-- we thought initially on the iPhone that we were going to have this huge repertoire of gestures to do all these different sorts of things. But when it came down to it, there was touching, and panning, and swiping, and pinching, and rotating. And that's about it. So, once-- for the iPhone because we really-- we just couldn't really think of other really useful things to do given the, again, the limited screen real estate and the number of interesting interactions that we could think of to add to the system. So, for the iPad, once that came around with the larger screen space, Scott said well, Steve wanted to figure out if there were any gestures that we could add to make multitasking work more smoothly or just add this gestural interface to it. And so, he asked me to start looking into it. So, I did. I mean I can pause there or just continue. I don't know if that's going in the direction that you want.

Hsu: I think so. Is there any more detail you'd want to add?

Kocienda: I mean I spent a year of my life on it. I could talk to you about it for, not quite a year, but I could spend a long time talking about the details. Basically, this is-- one of the interesting things that I'll pull out to talk about, that I think is interesting, is when you have multiple touches coming down onto a surface, to recognize those touches as a hand. So, what's the difference between something that looks like this with five fingers and something that looks like that. And so, I actually worked with Wayne Westerman to help me. He really was the guy to crack the problem to have-- to figure out some code to determine whether five touches looked like a hand or not and to have something like that not-- be rejected. But something like that be accepted. And so, these gestures wound up requiring either four or all five fingers to work. So, we had a scrunch to go back to the home screen and a swipe to go between apps and then kind of a push up to bring up the-- a multitasking bar. And that's pretty much what we wound up with, adding this repertoire on top of all other apps. There's just one more thing that I would add on top of that then, which is the decision to-- when touches are made on the screen to answer the question of whether those touches should go to the app that's front most, or whether the system should process it because going from the app back to the home screen is actually done by Springboard. It's actually done by the home screen, by the icon app. And so, figuring out event routing wound up-- we needed to make a really quick determination, once these touches came in, whether it was a "system gesture," I called it, or whether those touches should get plumbed through to the app. And that actually turns out to be something that makes the iPhone X work because those-- that same system of determining whether a gesture is something that belongs to the system or belongs to the app is fundamental to making a gestural interface to control the apps on the system as well as interact with the content in an app.

Hsu: Thank you. Let's move on to iOS 7. So, iOS 7 represented a major design change from what had gone before.

Kocienda: Sure.

Hsu: What motivated these changes, and what was your involvement with that?

Kocienda: So, well Scott left. And there was a decision, I don't know where it came from, to really do a radical redesign of the whole operating system.

Hsu: Coming from HI, or elsewhere?

Kocienda: Yeah, coming from HI. I mean I think coming from Jony mostly.

Hsu: Oh, this was after HI had been moved under Jony?

Kocienda: No, no, not yet. No.

Hsu: So, Greg [Christie] was still there?

Kocienda: Greg was still there. So, I was just coming off of the experience with Siri. And so, I-- my project was cancelled. And so, I needed something to do. It was just right around that time that iOS 7 started getting cranked up. So, I was essentially a project leader for the year on Siri. And I just decided to jump back into programming. And so, for iOS 7, I did blurs, worked with one of the smartest people I ever worked with in my career, John Harper, who did the graphics level implementation for blurs in Core Animation. And once he figured that out, then I did the whole higher-level part of it making it work in apps, figuring out what a kind of a blur recipe should be, styles, light styles, dark styles, what the-- kind of giving this frosting look to the way blurs looked. And basically, did that for a year, then the other part I did in iOS 7 was text legibility. A lot of drop shadows were taken away from text, so figuring out how to make text legible without shadows, which basically meant adding shadows back in but selectively. And so, it was basically doing those two things, really graphics related programming and really making-- trying to make the user interface look good for a year and then providing APIs for those things, working a lot with designers, working a lot with the app teams to get those APIs adopted.

Weber: One last Siri question--

Kocienda: Sure.

Weber: So, I had heard afterward that the Siri folks felt Apple-- I mean Viv was partly because they felt Apple didn't let them do some part of what they wanted to do with Siri. What were those parts? And what's the-- or does that make sense?

Kocienda: Well, I mean you'd have to ask them really. My take on that was there was just a, I think, a difference in overall approach in what the goal of technology was. For Apple, it's always to make things easy for people. Whereas, I think they had some ideas that were much more tied directly to technology. I don't know if that's really fair to say. I think they wanted to-- I remember having multiple discussions with people like Adam who-- he wanted to make it possible for people to order airline tickets with Siri. And my argument was people are not going to talk to their phone and make a multi-hundred-dollar transaction until we can prove to them that this is a system they can trust. So, they-- let's do that maybe. I mean I didn't really ever think that that was something that people were going to want to do. But my argument was okay, let's actually get a couple of wins first, get people so that they really, really love this system and trust it. And then we'll see what maybe we can do next year or the year after that. One of the approaches of Apple that I think it's done very, very-- a very good job of over the years-- and I don't know if I mentioned this-- just a brief idea of this. This idea of picking a point over the horizon, that you can't see it now. It's over the curvature of the planet. But you know that point that that's what you're headed for even though you can't see it right now. But you start heading in that direction. And then eventually, you kind of-- you might wind up heading right toward it or going a little bit to a different direction. And but that point is that you have these goals that are outside-- that exceed the reach of your current grasp. But that's what you're headed for. And to me, that's the argument that I wanted to-- that I tried to make to Adam and others on the Siri team at the time about ordering airline tickets. That's-- maybe that's a maybe a year or two year-- it's over the horizon right now.

Weber: That makes sense.

Hsu: Okay. Can you talk a little bit about your work on the watch?

Kocienda: Yeah.

Hsu: How did you get-- how did you end up working on that, and how much did you know about it even?

Kocienda: Oh, I knew about it because I was disclosed on a lot. So, I basically just volunteered to work on it. And so, yeah, I just a little bit by-- little by little, I wound up contributing more and more to doing watch faces, so just actually making the watch faces work, making that API and infrastructure work with other people who were involved doing that. And I just wound up taking on responsibilities. Again, it's this-- kind of this very sort of-- Richard said the phrase, kind of old line Apple, that's kind of what it was. You see something that needs to get done. And you kind of just go over. I'm not on that team. But I'm going to be sitting here now. I'm going to be working on this thing. And so, that's what I did and wound up taking some responsibility for a couple of watch faces and some elements of the infrastructure and little bits of layout and contributing with some really other pretty terrific people, but helping out to get the thing done. So, I wound up doing a couple watch faces and helping out.

Hsu: I mean the watch, in a way, was-- at least for a time, was touted as the next big platform, right?

Kocienda: Uh huh.

Hsu: So, I mean how do you compare that experience to working on the original iPhone?

Kocienda: Oh, it was really different. I mean there was a lot more people working on the watch, a lot more people, three, four times as many people, engineers.

Hsu: Was it just as secretive or less so?

Kocienda: No, not really, not really, no. I mean some of it-- it's still-- the culture changes when the people change. But there's still the same basic-- there is a same thread that runs through it. But it was different. And I think that you-- just the potential of the watch, as a product, is different. Personally, I was in favor-- I wasn't really involved-- I was a volunteer. And so, I was doing app level work, even though it was kind of a fundamental app on the system.

Hsu: But you weren't core to that team the way you were core to the iPhone?

Kocienda: Correct. Joined much later, was not in it from the beginning at all, and just really started pitching in because try to get the schedule done sooner and try to make the product good. And yeah.

Hsu: So, it seems like you had sort of gone up into a management position--

Kocienda: No, I-- no.

Hsu: No. Or at least a project lead?

Kocienda: I was a project lead on Siri.

Hsu: Okay. But then you went back to being an individual contributor.

Kocienda: Yeah.

Hsu: Or I guess as a project lead, you're still an individual contributor.

Kocienda: Yeah.

Hsu: Okay. So, there was never like a more-- you decided that you didn't want to go into management. Or you didn't want to come down from management. It was more--

Kocienda: Well, I like making things. I like working on projects.

Hsu: Yeah.

Kocienda: That's what I like doing. So, yeah it never-- things just didn't evolve that way. I just kept getting interesting technical problems to solve, so I just kept solving them.

Hsu: I guess-- let's-- can we talk about the most recent stuff you did at Apple, iOS 9 and 10? Or is that too recent?

Kocienda: Yeah, I'd rather not talk about that.

Hsu: Okay. So, then are you able to talk about your decision to leave?

Kocienda: Well, there comes a time. I worked at Apple for almost sixteen years. And when-- I'm not a trained computer scientist. I'm not a programmer. I wrote programs. But I'm not a programmer. And so, I don't see myself in that way. And so, I'd started looking around and seeing that a lot of the people that I worked with in the past who I had my happiest times with were no longer there. And so, you start looking around. It's like where are all of my friends. It's not like I didn't have new friends or new people that I didn't like collaborating with. Of course I did. But in other ways, it wasn't the same. And something happens when you do a project like Safari/WebKit, the iPhone or the iPad even, but really those first two for me, Safari/WebKit and the iPad-- excuse me, Safari/WebKit and the iPhone, is that when you have these experiences with people, something changes about the relationship. I mean I tried to get at it before, this notion of trusting people. And trust is something that is earned over time and is built up kind of brick by brick, these experiences. So many that-- we've talked about so many over the time that I've been here speaking to you. And there's just so many more that keep coming to me as I think back on those times that there's just not time for. But I looked around, and there just weren't so many of those people around anymore. And so, I asked myself a question. It's like, "Well, I'm over fifty now, how many more new things might there be in my life? How many more things in my career?" And I thought that maybe well, one more big one that might take me ten or fifteen years to figure out, just like computers took me that long. So, I'm busy figuring that out now. And I couldn't do that in Apple. So, I had to leave.

Hsu: That-- oh, this speaks to-- I mean the-- you just spoke to like the fact that a lot of people are not there anymore. Could you talk more about just like how much Apple has changed in this whole time?

Kocienda: Well, it really is just-- it's people. The culture around what we did was essential to what we did. We made the culture. We made the things. The culture made the things. The culture affects us. And there's this whole sort of mutually dependent swirl of things that go between people and culture and projects, companies, and things like that. But people are really the essential element. So, when the people change, everything else changes with it. One of the things-- the iPhone project was very, very small. Projects aren't that small anymore. Well, that's just one thing that's changed about the decision making that goes into how to structure projects. I work best when people just carve me off something and say here, go do it. And that became-- it became less possible over time to really do that. Things that used to be done just by me were now done by teams of people. And as an individual contributor, I mean at that-- I found it difficult to really find enough elbow room sometimes.

Hsu: So, a lot of the change has to do with scale.

Kocienda: Yeah, just scale. And part of that is just look, I mean this is not-- I don't mean to depict this somehow as being some sort of evil nefarious plan. I mean in some ways, it's just-- Apple's a victim of its own success. You just wind up all with these huge platforms and these many more products. And so, that's just the way that it turned out.

Hsu: Do you think Apple is headed in a direction that it won't be able to replicate its successes?

Kocienda: Well, look, just about now, virtually, Apple's stock is at an all-time high. iPhone X has come out, and people seem to be liking it. I just-- what was it? Horace Dediu just said that the Apple Watch soon will eclipse the highest quarterly sales for the iPod, which was obviously a pretty influential product in the history of Apple and judged a huge success by everyone. If you go to the somebody and say the iPod wasn't successful, they'd look at you like you're crazy. It's the middle of daytime, and you're saying that it's night. And so, obviously, these products have a lot of potential. With the watch, as they add more health features to it, I mean this product could be saving people's lives. So, it's just-- for me, I hope they keep making great products for a long, long time. Even if, like Richard said, there's a certain feeling about-- there's part melancholy, part ego that you want things to be like the way that they were. But they never are. So, but there's this strong feeling to how things were in the past that I wish could somehow still be retained. But that's not the way the world works.

Hsu: Do you want to add anything?

Weber: Well, if this is what we wanted--

Hsu: Did we already talk about Eazel last time? I thought we already discussed Eazel last time.

Kocienda: I don't know. Did we because what do you want to know about Eazel?

Hsu: Because I thought we had already-- no?

Weber: So, with Richard, we wanted to go back to--

Hsu: Yeah, that was okay.

Weber: Okay, if we did talk about it, then that's fine.

Hsu: Yeah.

Kocienda: Yeah, I could give you sixty seconds on Eazel. It was a company that-- I wound up doing a couple of startups that failed. And I was in Sausalito. I decided to move down to Silicon Valley proper. And I found this company just by searching on the web, founded by Andy Hertzfeld and Mike Boich and Bud Tribble and one of Andy's friends, Bart Decrem, and make Linux on the desktop easy to use. I thought that sounded great. And I thought it would be a great opportunity to meet these people who were my heroes. And so, I did. I went and walked in and filled a couple of whiteboards up with some code. And they hired me. And I started working on their services. The whole premise of Eazel was a GPL version, GPL file browser, some proprietary network services, get those two apps working together, software catalogue, web file storage. And there you go. But it failed. We didn't get the software done very well. Nautilus finished, the services really never did. The integration was pretty much non-existent. And so, and the company ran out of money right in the dot.com bust, so it got shut down. Then I went to Apple.

Hsu: So, yeah, I guess well, the last-- I guess, we're wrapping up then. Actually, there is one more thing. Could you talk a bit about-- I guess we touched on this a little bit earlier, but like your interactions with Steve Jobs, and also when he passed away, what was that like for you?

Kocienda: Well, I demoed to Steve probably about six or eight times. That was the limit of my personal interaction with him. But-- and so, I never spent much time with him. I didn't have a personal relationship with him. But these times that I did spend with him, showing him work, were amazing. He surely taught me more in a shorter amount of time than any other person I ever met. He was just so clear about what he wanted. And he asked great questions. He was always open to different ideas. There were a couple of times that I changed his mind in a demo.

Hsu: Wow.

Kocienda: And he said he wanted something, and I showed him a demo for something different. And he said, "Yeah, okay. That's better. That's better than what I was thinking. And you've already got it." And that kind of just turn on a dime willingness to change his mind that you hear about, and yeah, it's real. I saw it. He had no reason to take my word for anything. He didn't really know me. But it was about the work. And I could show him that I was thinking about the work in a way that was the same way that he thought about it. So, when he died, well, that's Apple's soul. He's not replaceable. You can't-- you typically replace founders, right? But he was special. And that's no reflection on anybody who comes afterwards. He's special. So, yeah, I remember right when I heard about it. So, yeah, I mean it's-- I was always proud to work at the company that he founded and that was guided by his principles, that bore his imprint. And those were the best years of my career.

Hsu: Mm-hmm. So, looking back on your time at Apple, what would you say was your legacy and just sort of summarize that whole experience.

Kocienda: Legacy? I don't know that I have a legacy. I mean that's maybe a bit pretentious sounding. I could tell you what I-- I did the best I could. And as fate turned out, I got an opportunity to work on a project, a couple of projects, that got very, very wide distribution to a lot of people in the world, more people than can be counted. And so, I feel fortunate. And if there is a legacy, right, it would be that gosh, when you get a chance like that, do the best you can. And if things turn out right, and you work hard, and you make some decent choices, and you're with a whole bunch of other smart people, yeah you can change the world.

END OF THE INTERVIEW