Olivia Ryan: Okay, it's June 1, 2006, and we're here with Mike Pinkerton. Mike, when did you first begin using computers?

Mike Pinkerton: Oh, wow. Probably back when I was about five, so I guess that would have been about '78, '79. My dad had some ins at the UNLV computer department. And so it was the old, you know, big teletype—put the phone into the coupler and you know, and dial up that way, probably, you know, 100 baud or something, playing just TTY games on a printer, on an actual printer, you know, Adventure and Whompus and football and all of the kind of old mainstays of that time. And then I just started writing Basic programs on a TRS 80 a few years later, and then continuously—so I guess I've been doing it my whole life.

Olivia Ryan: And did you ever receive any formal training, or you're just totally self-taught?

Mike Pinkerton: College.

Olivia Ryan: Yeah.

Mike Pinkerton: I did a lot of self-teaching. I taught myself PASCAL, I taught myself C, I taught myself how to program on the Mac, and just mostly, through trial and error and reading books. And you know, I sort of thought I knew everything and then I went to college and I realized, wow, there's a lot of stuff here that I really need to—I really need to learn. So that was—that was both an eye opening experience and a wonderful experience, because it really, really turned me on to the field as a whole. You know, I knew it was what I always wanted to do, and my dad had originally been telling me—because he had a math, computer background, it's the field that he was in, and he said that you should do something other than computers. I don't know why he would tell me that, but that's what he said.

And so going through high school I thought, well, physics is kind of interesting. And then when I got to college and I took the physics—or physics major's class, and I guess it's you know, it's the weeder classes. I made it through two quarters of that and I said, no I don't like physics that much. And computers were always just a natural for me, so I just stuck with that. I mean, that was my declared major, I just was going to try physics and see if it fit, and lord no, it didn't! A little too much math.

Olivia Ryan: And when did you begin contributing to open source projects and how did you sort of get involved in the open source community?

MP Probably my first open source contribution would be through Mozilla. I was at Netscape when we decided to open the browser source. So that was—that was kind of my, you know, jump in with both feet, welcoming to the open source community. Some in college. I guess I had dabbled around with some open source products. Like, I was able to port ELM TTY mail reader to the AT&T 3B2. That was, you know, it was lame but it's, "Yay, I downloaded the source, I got it to compile and it works, and it works for everybody in the lab and they're happy." You know, so just—I guess that taste of being able to take things and get them working and you know, it's free, it's open, you don't have to—you know, you can just do it at your leisure. There's not somebody, you know, staring down at you and watching every move.

That sort of appealed to me but it didn't really appeal to me that I set out and said, "I want to take an active role in open source." You know. It was curiosity, but you know, I wanted to go make some money. And then when we announced that we were going to take the Netscape browser source, the Navigator source, and open source it under Mozilla, you know, like I said, it's like jumping in with both feet. You know, we had three months to take this code base that never expected to see the light of day, and make it not just—not just buildable and usable, but also something that wouldn't get us sued, you know. So we had so many—I mean, I could talk for hours about the stories of that three-month Source 331 project. I mean, it was just an incredible time at the company, at Netscape.

Olivia Ryan: Well, you had to have maybe one or two moments that really stick out in your mind at that time?

Mike Pinkerton: Yeah, so I guess there are a couple. The first one that I always thought was sort of interesting was, we were getting really close to the end and—there's two stories that are close to the end. One, we were like a week from the deadline, March 31st, 1998, I guess, is when we open sourced. And so we were like a week before. And one of the outlyers was some menu code that we'd taken from Apple's sample code and so we had that in the Mac product to do some stuff with our menu bar. And we didn't have the license cleared, at least to legal's approval, to open source that. So we were like, "Okay, well, what do we do?"

So, Scott Collins in the cube next to me, he stayed up all night one night writing the whole thing. And so he completely replaced all of the code, just ground up re-wrote it, and sort of simultaneously our management was calling up Steve Jobs and saying, you know, "Please bless this, we'd really like to ship this on time." And you know, eventually he acquiesced. But there was just a lot of frantic back and forth between high-level Apple people and high-level Netscape people and it was you know, "Are we going to be able to ship?" And—but you know, it came through and everything was fine.

The other story which is kind of interesting on a lot of different fronts, sort of telling of the industry—I was at home one night—we were probably about a month from releasing, and my boss calls me, like, 9:30, and he's like, "Pink? How hard do you think it would be to rip bookmarks out of Navigator?" I'm like, "Um, well, Gramps, I think that would be pretty difficult. Why?"

Apparently, this old company named Wang, which you know, was very big in the microcomputers in the 70s, 70s and 80s, had a patent that they wanted to try and enforce on storage of—identification of remote locations of information. So, effectively a bookmark. And so they were suing us because you know, it's not just so bad that we've been violating their patent all this time, now we're going to release how to do that to the world, knowingly, in about a month. You know, and knowingly violating someone's patent is about a hundred times worse than just doing it by accident. And so we were panicked, we were absolutely panicked. It came out later that Wang had recently been bought by Microsoft, and so it was really just Microsoft's lawyers going through—because Wang was nothing but a shell company at that point, it was all lawyers. Just going through the patent files, seeing who they could sue. They came across one and said, “This will really mess with Netscape.” And so they just, double barreled shotgun, you know, full force.

And so we’re running around like chickens with our head cut off, we don’t know what to do. And in his one shining grace, the only reason I have any respect for Marc Andreessen at all, he told us, you know, kind of sat everyone down, and he said, “Don’t worry, just business as usual. We’re going to fight this. We’re going to win this. It’s not a problem.” You know. And he just got everyone back on track and you know, we challenged the patent and it was overturned, because there’s tons of prior [indiscernible], it was a bogus patent to start with. But that just kind of let us get back into our focus and deliver on time, you know, and I always thought that was kind of—that was great of him to be able to stand up and be a leader and say, “Don’t worry, we’re going to put all of our guns behind this as well. You know. It’s not your problem, just keep moving forward.”

Olivia Ryan: How long did it take to go through the legal process? Did that all happen before March 31st?

Mike Pinkerton: Yes.

Olivia Ryan: Oh, it did.

Mike Pinkerton: Yes. So, it was like January of ’98 when we announced that we were going to do this. And so we had like, three months, literally, to go from you know, beginning to end. And we had to scrub all the code, because there was, you know, like I said, it was code that was never meant to see the light of day. There was code we copied from other companies. There was code that other companies paid us to include. There was code that had comments that you know, not necessarily mentionable in polite society. And in addition there were comments that were very slanderous to our partners, you know, “Such and such at Price Waterhouse made us fix this, grumble, grumble, grumble.” You know, and in more colorful language, of course. And that’s not the kind of thing you want to be putting out for the world to see.

And so we had to go through millions and millions of lines to scrub it. And while we were doing that we had to make sure we weren’t violating anybody else’s copyrights, we weren’t violating any other patents, you know, the whole thing. And the security actually became a really big problem, because back in ninety—the late nineties, the federal government wasn’t as lax as they are today about exporting crypto. And so at the time, you were not allowed to export anything that even had hooks to possibly implement crypto. So, you couldn’t just rip all the SSL stuff out of the browser and ship that with hooks in it so someone else could put it back in because that enabled somebody to put encryption into your products, and that was not exportable.

So, we had to find a way to basically reverse engineer all of our security libraries, you know, fundamental core of you know, our SSL stuff, in a way that the federal government would allow, in three months. And since, all that’s changed. We wouldn’t have to go through any of that today. But it was just, you know, one more thing.

You know, just tons of stories like that. Like I said, I could go on for hours.

Olivia Ryan: How many people were working on that at the time? You said you had to clean up all the code . . .

Mike Pinkerton: I'm trying to recall. There was probably about—and this is just a guess—I'd say about 30 to 40 engineers fully dedicated to that. That was the browser team. The browser and the core team went and did that. The mail/news guys, they went and shipped Netscape 4.5, so they didn't open source—we just—we forked, and they went and shipped Novo which is Netscape 4.5, which is all of the Enterprise improvements to mail. So, there were a few tiny improvements to the browser and mostly just rewritten mail. And so they're off doing that in closed source while we're off doing the Source 331 project. And then after March 31st, we had to try and get the two back together. And that was a lot of fun. Not so much.

So, you know, so that whole wing of the company, of Client, I guess, was, you know, had nothing to do with it. They were off shipping their own stuff. They—you know—I don't know if they were—they weren't openly antagonistic about it, but none of them thought it would work. Of course none of us thought it would work either. It was kind of one of those Hail Marys that, you know, where can you go where Microsoft can't follow? You know, that was the basic, underlying premise behind doing the whole thing which is absolutely brilliant, whoever came up with it. Whether it was Jamie or Marc, or whoever. It was just absolutely brilliant. You know, where can Microsoft not go? They can't go open source. You know, they've put their stake in the ground and it's not that direction.

And you know, so we envisioned that we would get the huddled masses of programmers, you know, little Linux geeks and all these open source people, and they'd flock to our project, and we'd be the greatest thing ever, and lauded and celebrated. You know, obviously, none of that really happened. And it's not so much that our management screwed it up, because you know, we screwed up quite a bit around that time. It's just the open source community doesn't work that way, you know.

The open source community really works on street credit, you know. You have to sort of be established before you're taken seriously. And so, you know, Netscape was—we were relatively well established and then you know, we decided to do this open source thing. So, got all of this press, got all of this interest, and we started working on it. And we found that, you know, partially the develop—a lot of developers didn't really come flocking to us. And we said, okay, well, this will change over time. So, we kind of went on our daily way. And you know, all of this is after March 31st. We were trying to be as open as possible, at least engineering was trying to be as open as possible. You know, we'd wear our two hats. We'd have our Mozilla hat and our Netscape hat. You know, and to make any decision as to whether or not we wanted to include a feature, we'd have to consult both hats, you know, which is kind of schizophrenic.

But engineering was really taking it seriously. Marketing and management didn't give a rat's ass about Mozilla. They wanted Netscape features to talk to Netscape servers, to talk to the Netscape website, and if it had to tie directly to our website, they didn't care. They were you know, they were paying our paychecks. And so they'd say, "Do it!" Well, this is not something that Mozilla would want. "That's not my problem. You're getting paid. Do it!" You know, so there was a lot of that.

And then there was also a lot of angst around the switch from the old source base, the source code that we originally open sourced, which was based on, you know, the Netscape 0.1, the Mosaic, you know—well, it wasn't based on Mosaic, but I mean, it came from Mosaic and it was that grad student project gone horribly wrong that ended up, you know, shipping five years later. Nobody ever expected that code to last longer than a year. It lasted five.

And so we had all of these internal discussions, and I've got hours of stories on those, about switching over to this new, you know, more standards compliant, faster, smaller, more embeddable, going to save the company, Raptor engine, HTML rendering engine, which eventually became Gecko, due to lawyers getting involved. And so everyone in engineering was like, "No, that's too risky. We're getting ready to ship Netscape 5. We're a couple of months from beta. We've been doing a lot of stuff with the open source community. We've been trying to put our, you know, put all of our plans up on the Web and get everybody involved and keep everybody appraised of what's going on."

And kind of behind the scenes, our management said, "Yeah, that thing you were working on—that's dead. We're doing everything on Raptor now." And, you know, nobody consulted the open source community. Nobody told anybody until the decisions had already been made, because it was completely out of our hands as engineers. And, you know, and so in—one of the most wonderful displays of career limiting moves, Dave Hyatt went and you know, as soon as he found out, he went into our open source Bugzilla bug database and he wont fixed every single one of his bugs with the line something like, you know, "Gromit's dead." Gromit was Netscape 5. "Gromit's dead. Everything's moving over to Raptor."

And that's how people found out that management had made this decision. By Hyatt going in, and wont fixing his bugs, which CC'd millions of people who were watching his bugs. You know, all around the world, everyone instantly found out through their bugmail that Netscape had made this decision about an open source project. And so we lost immeasurable amounts of street credit at that point. And I think that's—you know, in the end it was the right decision. It was totally the wrong way for us to go about it and we didn't learn that lesson for years to come.

So, you know, that's what really made a lot of people pull back and say, "Well, if they're not going to take this seriously, I'm not going to put my effort in. I'm not going to, you know, I'm not going to donate my blood sweat and tears to this project if they're just going to keep making decisions from on high, you know, in their cathedral, and not want to play in the bazaar model."

So, I think that's what really hampered us for the first couple of years.

Olivia Ryan: So, how did changes come about?

Mike Pinkerton: Well—

Olivia Ryan: How did it shift--?

Mike Pinkerton: One of the things that we did realize, you know—there were some—there were quite a few benefits that did come out of this, and one of them was QA. You know, we discovered that even though the developers didn't want anything to do with us, you know, the hard core open source folks, people were willing to download nightly builds and file bugs. So, we had this wealth of QA that we never, ever really expected to have. You know, we expected we'd have the developers. What we got was the QA. And that really helped us kind of take it up to the next level. I mean, okay, we were doing everything in-house still because a lot of external contributors weren't applying, but everything we did do had this wealth of QA available to us, so we were able to you know, to make pretty good progress.

We went through several years of growing pains, obviously Netscape 6 was a piece of shit. And we just—we had to release it—our management made a decision, we had to release it otherwise we were going to be completely useless in the marketplace. So we had this thing that we'd been working on for two years, that we thought we'd have done in three months, that took us two years, and it was crap. It was absolute, a hundred percent unadulterated crap. Everybody knew it, and we just—we had to ship it. So we shipped this turd out and everyone was like, "Oh, my god. This is what you guys have been doing for the last couple of years? You know, I'm taking a hike."

But then over time, we were able to use that QA, you know, help us fix the bugs, help us do performance evaluations, help us test on all these machines that, you know, that we'd never have had access to otherwise, test on different platforms. Because we were you know, we had a very cross-platform code base, and so if somebody could check a fix on one platform, and someone could test it on another. So that completely increased the amount of QA resources we had available to us. And also, you know, would help find the rough edges around the platform boundaries.

But you know, Netscape 6.1 was a lot better. 6.2 was even better than that. 7.0 was actually a reasonable product. You know, 7.1 and 7.2, I didn't have much to do with at that point, I've shifted over to AOL. But we started to get our street credit back, you know, and what really did it was execution. You know, we took something that everyone agreed was crap and we kind of hung our heads and said, "Yeah, I'm sorry my name's associated with that." And we stuck with it and we made it better and we played more in the open source community. You know, we fought vehemently to keep Netscape specific things out of you know—Netscape and AOL at that point, Netscape and AOL specific things out of the shared Mozilla code base. And you know, we would—we would raise holy hell whenever anybody would violate—would step over that line. You know, we'd just immediately back them out and run to their manager and you know, play all sorts of childish little tricks. But, it got the job done. You know, it let everyone know that we were serious about maintaining Mozilla as an independent entity.

You know, and again, whether or not management believed any of that, and you know, you talked to Mitchell, and I'm sure Mitchell has a whole bunch of stories about dealing with AOL management. But, I think the community started to notice that engineering was really serious about it. And that and delivering a better product over time, iterating, continuously improving it, really started to get our street credit back.

What really kind of changed the whole ballgame was, for two independent groups, me and Hyatt with Camino, and I guess Hyatt to some extent and Ben and a couple of others, gone off and doing Firefox, or Firebird, or Phoenix, or whatever it started out to be, just realizing that this monolith that we were piling everything into that was Seamonkey, it wasn't, you know, nobody wanted to use it. The geeks wanted to use it, but it was so horribly complicated that nobody really wanted to make it their browser. If you wanted to go set a preference you had to dig through, you know, five panes and three tabs and flip something that doesn't really even sound like the pref that you want. And you know, it wasn't appealable to the masses.

And you know, especially, you know, what really got me started with Camino was, it was a terrible product on the Mac. You know, the Mac was always sort of the third class citizen in the ballgame. And so, you know, people would do everything for Windows, and they'd do everything and they'd test it on Windows, and it wouldn't work at all, or it looked like crap on Mac, and nobody really cared. Except for the few of us who, you know, who's primary—that was our primary platform.

And so, a couple of us said, you know, we can do better than this. There's got to be a way that we can do better than this. We needed to make it simpler, we needed to make it more approachable to the everyday person. We needed to cut out three-quarters of the functionality that nobody cares about and nobody even knows about, and in a sense that makes it faster because it's less complicated and there's less to load, and all that. And so those two efforts really kind of issued in a new era of Mozilla. You know, there's still a lot of people, even to this day, there's still a lot of people in the Seamonkey camp who are angry about the forking of, you know, going off and making a new product. But it really was necessary. You know, the Seamonkey product, the Mozilla product, you know, the all bundled together mail, composer, browser, nobody really wants that. And I think we've demonstrated that fairly clearly over the last year or two.

You know, and so that's what really stepped up the visibility. You know. Camino was pretty good, but we never got anywhere close to the amount of appreciation, I guess, that Firefox does. You know, and—

Olivia Ryan: Why do you think that is?

Mike Pinkerton: Well, it's a Mac only product, I think, primarily.

Olivia Ryan: But it also doesn't seem to be marketed really—I mean, Firefox obviously, there's a big push to market Firefox and there's really not much of a push at least that I can tell to market Camino to Mac users

Mike Pinkerton: Right. There's none by the Mozilla Foundation or, and now, especially, there's none by the Mozilla Corporation. And you know, I don't know how much you want me to talk about this, but this is a giant sore point I have with MoFo and MoCo is that, you know, the way that they make money is by pushing Firefox. They don’t care about anything else, expect for Firefox. The fact that there are these other projects under their umbrella, Mozilla.org, is you know, is something nice that they can tout when it’s convenient to them, but they don’t know how to market more than one thing and all they can market is Firefox.

And so, you know, when you—when you go to their website and you say, you know, download Firefox, oh, yeah, there’s Camino too. You know, what kind of message does that send, you know, to the person coming to the website. And so recently within the last year, I guess, we in the Camino project totally—we separated all of our website from Mozilla. We just said, “All right, screw you guys, we’re going to take our ball and leave. Because if you’re really not going to do anything to support us besides some infrastructure, you know, and just give us minimal lip service, we’re going to go off to another website and build our own community.”

And it’s unfortunate that that had to happen, but thankfully the two communities are still intermixed enough that, you know, we get a lot of benefit from that. But I think that’s what really kind of got the upturn on Camino propaganda, if you want to call it that. It was just being separated from the Mozilla Foundation and people knowing that they could go to this other site that had all of this wealth of information that didn’t just push Firefox actually, with every single page you went to. Because that’s really all, you know, the Foundation then and the Corporation now, want. You know, they want to make their money on Firefox because they got to stay alive. They got to make money.

Ken Albers: We’ve had a couple, well in particular, we’ve read some things about Mozilla Foundation versus Mozilla Corporation and you know, you’ve mentioned a couple of times so far, different aspects of management that you’ve dealt with as you’ve gone through. You know, you said one of the appealing things when you first worked at open source was no one was sitting there watching you do it.

Mike Pinkerton: Yeah. Right.

Ken Albers: And you know, you’ve gone through this process for a long time, you know, with Netscape and then you were there for AOL a bit right?

Mike Pinkerton: Mm-hmm.

Ken Albers: And then Google.

Mike Pinkerton: Right.

Ken Albers: And you know, and now—you’ve seen the Foundation become the Corporation. And in particular, you know, we’ve read Chase Phillips talking about how he doesn’t know where the Corporation is going, you know, and things like that.

Mike Pinkerton: Yeah, that was really bold of Chase to do.

Ken Albers: Yeah, just—

Mike Pinkerton: Kudos to him.

Ken Albers: Do you have sort of a comment on how the managerial process has shaped the way Mozilla's worked over time? And things like—

Mike Pinkerton: Well, so—I'm not really sure how much I'm allowed to comment on the reasons for the Corporation. But one of the things I can certainly discuss is, you know, before there was the separation, you know, there was Mozilla Foundation and there was a group of people who I respect very much. I respect Brendan a lot. I respect Mitchell a lot. You know, a lot of the people working there are wonderful people. But, it sort of seemed that as a whole the decisions that they were making were all very, you know, it's all about Firefox. You know, and that's kind of what really caused us to take the ball and go home.

For Camino, you know, utter refusals to do any kind of press releases for us, just absolute flat out refusals at one point. And we're like, okay, great, you know, why are we playing with you guys? And then when they made the split with the Corporation—you know, they needed—okay, so now they're a money making entity, they need to find ways to you know, to bring in more revenue, so they hired a lot of product people and they hired a lot of kind of middle to upper level management. And some of the people that they brought in have been in the community for a while, you know, they brought in Mike Shaver, he's been around for a while. They brought in a couple of other developers who, you know, who've been in the community, who were established in the community. Everybody knows who they are.

But they also brought in about four or five people that nobody had ever heard of before, and were, you know—and if the whole open source world works on street credit, then how does bringing people into the decision making positions, you know, the people deciding what is going to be in Firefox 2, or what is going to be in Firefox 3, or do we want this feature or do we not. How do we want to do it? You know, the people making those critical decisions, if they don't know how the process works, because they haven't gone through it, you know, they have their own, you know, 'I got to make money for my company' agenda, I don't know, I don't want to push that on anybody, but it's there in the ephemeral. But if people don't know who they are and they don't—people in the community don't respect these people, because they have no street credit, you know, what does that do for the community? What does that mean for the product? And so there's been a lot of gnashing of teeth over that over the last, probably six to nine months. And I think that's the biggest problem that the Foundation and the Corporation have to deal with moving forward.

Olivia Ryan: Okay. Maybe we should sort of focus a little bit on what projects you've worked on specifically.

Mike Pinkerton: Sure.

Olivia Ryan: So, if you could kind of—I know you've been with Mozilla for a long time, so if can kind of take us through everything you've sort of worked on.

Mike Pinkerton: Sure.

Olivia Ryan: And what roles you've played, and that kind of thing.

Mike Pinkerton: So, in the beginning, before we did open source, I was on the Mac Navigator team, the Mac Navigator front-end team. So there were three or four of us, maybe five tops, who—our whole job was to put the Mac native front end on top of this core infrastructure. You know, sort of the reverse of what we're doing now. You know, everything the user saw, you know, in the chrome was all native. It was like Camino was, or it was like Camino is. And then when we decided to do this open source thing, we kept that going. So, I was working on Mac Navigator stuff, I was working on what we were going to call Netscape 5, the code name was Gromit, and then eventually, for other legal reasons, got renamed the Seamonkey. And so we were working on front-end changes—my team was working on front-end changes to that open source web browser. And then when the mail stuff got landed in, you know, working with all of that—so getting the Mac mail and Mac composer and Mac browser on top of this cross-platform code.

And then management made this decision that we need to move over to Raptor, which then became Gecko, and they kind of fairly bluntly told us, "We don't have enough resources to do individual platform development teams. So, either all you guys can work on Windows or you can find a way to fix that." And that was the genesis of XUL, of our cross-platform XML based UI language.

So, my team, which they took the Linux front-end team, the Mac front-end team, and the Windows front-end team, glommed us all together onto one single team and you know, if we wanted—everyone sort of agreed that one of Netscape's biggest benefits to the world was that we shipped on 20 products—or 20 platforms, simultaneously. You know, every release went out on Mac, Windows, Unix, 18 variations of Unix, you know, H pucks and Solaris and Sono S and things you've never heard of before, and just all simultaneously because we had this great culture of doing things cross-platform and doing them right.

And so, we decided, okay, if we can't do native front-ends anymore, you know, what can we do? How can we go about this? So, the primary genesis of XUL was to do a cross-platform UI in such a way that—I guess, let me back up. So, one of the things we also realized is, we were making this transition and doing user interface work, actually coding user interface work, is very sort of secretive. You know, it's magic. You know, people—some people have the ability to do that magic and they can write the lines of source code to make the pretty picture appear and draw the lines and do all that stuff, but a lot of people can't. You know, a lot of people can't write C++, they can't, you know, they don't want to go and learn Win 32, or Mac OS API or GTK, or whatever. You know, there's a gigantic learning curve to building native software.

And we said, all right, so who are the people who have the experience to do really good UIs? Is it the people that have that magic—absolutely not. You know, programmers make the worst UIs. That's pretty fundamental across the entire industry. So, why are the people who are doing the magic the ones also doing the UIs? It doesn't make any sense.

So, we said, okay, how can we open this up to not just, you know, to everybody, the people who have the skills, the UI designers, the graphic designers, the people that understand human factors and you know, the psychology aspects of user interface design? What skills do they have? Well, predominantly, you know, they're artists, so you know, they probably by this point know—they know HTML, they know how to do Java Script stuff, they know how to do DOM manipulations, you know, they probably understand CSS so they can make their web pages look beautiful. And these are all standards, you know, these are all totally open standards controlled by, you know, certainly not controlled by Microsoft.

So we said, okay, well what if we can design a platform that uses all of the things that people know, that the designers know, so they can prototype UIs really fast and then hand the whole thing over to the developer to fix the bugs, and you know, and open the door to all of these college students who are right now—you know, a lot of them aren't learning C and C++, they might be learning Java. They're probably learning web development, DOM, CSS, XML, you know, because that was really big in schools at the time. You know, put the people who have the skills to work making our UIs better. Because everyone agreed our UIs pretty much sucked.

So, my team became the XP Toolkit team, and we spent the next year and a half to two years making the base for XUL so that people could you know, write Java Script and XML and make cross—fully cross-platform UIs and still plug in when you needed native components, either for speed or to touch the individual platform pieces. That was still completely available. So a lot of the core team, you know—so there was kind of the core team down here, and then the XP Toolkit team built on top of Gecko. So, we used the fact that Gecko did all of these technologies to our benefit. So, okay, we don't have to write something that parses the DOM or does DOM manipulations or handles HTML 4 or CSS, we have that, you know, that's the Gecko team. Let's build an application object model on top of that.

And I think there was, you know, the end result took a really long time to show fruition, to show fruit, to bear fruit. In Netscape 6, obviously: ass. And so everyone took a look at that and said, "Well it's ass 'cause XUL sucks." Well, no, it's not that XUL sucks, it's that we released—we had all of these bugs, it was way too slow, we didn't have any time to do performance tuning. You know, we were building the house while we were still building the foundation. You know, and of course it's not going to be that great.

But with every iteration it got better, it got faster, it got more stable, we got more functionality into the core. And you know, I think Firefox really showed—if you take all the cruft out, if you take all the bad UI that people in the community had contributed, or people at Netscape had contributed, just pitch all that, take people who really know this stuff, like Ben who we hired out of college, Ben Goodger, you know, he knew XUL cold. You know, he could write XUL like nobody could, and just let him do it right and just show off, you know, how easy this stuff really was. You know, it didn't have to be complicated, it didn't have to be complex. And that really sort of issued in a new era of XP Toolkit as well.

So, that was that part. And mostly my role in that was, you know, guiding the—helping to guide the architecture. We sort of all came up with it together. You know, it was a very team oriented platform, you know, everybody had their contributions. Because we had three different platform teams merged together, so everybody knew, okay, this we can do, this we can't, we have to do, you know, if you want to implement drag and drop, you have to do it this way so it'll work on Mac and Linux, even though Windows works this way. And you know, all of the really hard problems—we had the right people in the room to solve them. And so that was great.

And so I did a lot of the clip board stuff, I did a lot of the drag and drop stuff. I inherited the menus from Hyatt and fixed them up quite a bit, so you know, any time you pull down a menu in Firefox that's my code. Any time you copy and paste or drag and drop, that's my code. What else did I do?

So, after that, we, you know, we were bought by AOL, Netscape, and AOL bought us really for one reason. It was as a leveraging tool against the IE browser in their AOL Client. You know, they really wanted to replace the embedded IE with Gecko, and so you know, we'd always been touting that Gecko was small and embeddable and all of these great things. And in fact, for the first couple of years we sold a bag of lies. But, once we actually started putting some effort into it, you know, we created this embedding team and we decided to focus on—not just for AOL but also for other companies, you know, how can we make Gecko embeddable in, you know, mobile phones and in all of these great places that need browsers, that need—that want open source. They want the security, they want the full standards compliance, all these great buzzwords that we'd been touting all these years, you know, let them put them in their own projects, and so they're not shackled by having to go to Microsoft for everything. You know, because when AOL had a bug in browser, you know, what do they tell their 30 million customers? "Sorry, that's Microsoft's fault—please keep giving us money." You know, it doesn't work so well over time.

And so I sort of transitioned off of XP Toolkit onto this kind of new embedding team and my focus was on Mac embedding. And so, me and another guy, Conrad Carlen, who—I think he's at A9 now, really good guy, really good guy—we hired him out of the open source community because he was doing Mac embedding stuff.

Olivia Ryan: On a volunteer basis?

Mike Pinkerton: Yeah, on a volunteer basis. We hire a lot of people. I mean, pretty much the majority of people who are contributors now, were not at Netscape. You know, they got jobs through Mozilla, through their contributions, which is a wonderful way to acquire talent.

But—sorry to sidetrack there.

So, I was in the Mac embedding team and that's when I moved out here, back to Virginia, to—now, I was still working for Netscape, but you know, I wanted to move back for a variety of personal reasons. So, I was originally going to work at home and just telecommute to Netscape, and thankfully my mother said, "You'll go crazy by yourself. You need to be in an office with people." So, I got in and I found an office with the Mac AOL Client development team. And so I was sitting with them and right about that time—so I guess that was September 2001, so around January or so, 2002, they said, "Hmm, well, Mac IE isn't being developed anymore. And

Microsoft has pretty much given it the finger. Maybe we can shim this Gecko thing in and it'll work."

So, my job effectively became helping the Mac AOL Client team get Gecko into the Mac AOL Client. And eventually, I think sometime that summer, we actually shipped it. And that is the first and only use—well, no, no, Communicator—so it's the first use—well, it's the only use in an AOL Client of the Gecko browser. So, they had it working on Windows in both AOL and CompuServe. They shipped—I don't know if they shipped CompuServe. Sorry. Yeah, they shipped CompuServe with Gecko in it, but they never shipped the AOL Client. And then we shipped the Mac Client with that second. So, oh well, went out second.

But, you know, that was a really great time for Netscape and for Mozilla because now we had this, you know, we got AOL on board. You know, we're going to take over the world, we're this close! You know, and then people in AOL management who shall remain nameless, I refer to him as Doctor Evil, decided that yeah, we're not going to do that anymore. So, began this systematic, just destruction of Netscape engineering and management, and the Mountain View offices. And that was a fun couple of years—not so much.

But thankfully, I was in close enough with the Mac AOL team that when it looked like the ship really started to sink, I was able to transition over to AOL proper and work on some projects there—just a whole bunch of Mac projects there. Some of them using Gecko, some not. None of them ever shipped—but—thanks to Doctor Evil—won't go there. So when all of the major layoffs started happening at Netscape, I was already off that boat. And so I consider myself very thankful for that, because that would have been—that would have been very difficult for me, having put, you know, so much blood, sweat and tears into that project and that program, to, you know, just be handed a pink slip.

But, you know, that was some really dark days for Netscape, that couple of years. And you know, a lot of friends, people you've talked to, I'm sure, you know, everyone knew it was coming but nobody really wanted it to happen to them.

And then, so, skip ahead a couple of years to late last year, I guess. You know, none of the stuff I was doing at AOL was really Mozilla related. Though at the time, I guess that's about the time—I guess we don't need to skip ahead, I can talk about Camino. That was about the time when we decided that okay, Seamonkey isn't going to work. Certainly it's a bad Mac product. Let's see if we can make a good Mac product. So, we put a native front-end on it. We put a Cocoa front-end on the browser for OS-10. And originally we were doing it as a proof of concept for Apple, because they came to us and said, you know, "We'd like to make a better browser on the Mac, you know, do some prototypes for us and we'll see if we can talk." And eventually that never ended up going anywhere, but those proof of concepts were the beginnings of Camino, which at that point was called Chimera. And again, the name changed for legal reasons.

And I forget what—so I guess that was around—that was round late 2001. Mid-2001 is when we started those prototypes and so by late 2001 we were—I had just moved back here—we had a couple of initial version out, like 0.1, 0.2. And Dave Hyatt was sort of the project lead, because he was bored with—he was bored with working on XUL and he wanted to do something else.

And so he sort of, you know, set the initial direction with some contributions from me and a couple of other people, and you know, drove his hands crazy, he had carpel tunnel just coding all this stuff because he thought it was so much fun, and he did, you know, he led the first couple of dot versions.

And then we got a lot of public praise. You know, we got a lot of street credit and recognition from the community. Because it started off as a MozDev project, you know, so we were separate from the Mozilla code, we'd check out the Mozilla code as embedding, and then we'd have, over on MozDev, the native front-end Cocoa code that made it Chimera and you know, you put the two together in your build and you know, you had a new browser. And so that piggy-backed off of a lot of the embedding work that my team had been doing prior to that, and continued on as we were working with AOL. That benefited that greatly.

So, we spent about the next year kind of in this limbo state of just kind of cranking out versions in our free time, developing this community with, you know, Eric Raymond in *The Cathedral and the Bazaar* calls it the 'plausible promise'. You know, you have—you put out to your community this plausible promise that eventually you'll get to this place that'll be really, really cool. And you're not there yet, you know, it may be buggy and it may crash, but people can see the glimmer of hope to get to that promise that you put out there. You know, and the promise that we put out there was, you know, a native Mac browser that had Gecko as a core, fast, small, secure and had a native front end so that it felt like a Mac browser. And a lot of people who were sick of Seamonkey and were sick of Mac IE, stood up and said, "That's awesome, I want to get behind that."

You know, and so we got a lot of initial help, again mostly in QA, which is where we really needed it. Because you didn't need so many people—the old mythical man-month problem, you don't want so many people hammering on the same code because you get in each other's way, but the QA generally doesn't suffer from that problem. It was very distributed. It can be done very easily in a distributed manner. So, we had a lot of QA and that helped the product a lot.

And then I guess in, sort of mid-2002, if I'm remembering correctly, Netscape decided, okay, well, let's put some funding behind this. Let's do this for real. You know, it's starting to get a fledgling community. Let's try and do the right thing for the Mac platform, for once. And so, me—we got some dedicated QA, so there was like, me and Simon and Conrad and Kathy, Hyatt pitching in a little bit, but not too much. He'd started working on Firefox by then, I think, or Phoenix, or whatever it was called at that point, because we'd hired Ben on by that point.

And so we started this project and we're like, okay, well, you know, what do we call it? You know, because if we're going to release—so we have Chimera and the open source, what are we going to release this as a Netscape branded product? And so, I guess 2003, for Macworld 2003, we were going to release Netscape Project X. And that was going to be our beta of a Netscape branded Chimera to say, "We're taking this seriously, we're going to hand out—you know, we had 40,000 CDs pressed, we're going to hand them out at the show, say, 'Netscape's back. We're back on the Mac. We're kicking ass. We're, you know, this is our plausible promise, it's going to get better.'" And then three days before the show Doctor Evil cancelled it, and he said, "No, we're not going to do that."

So, anyway. We were all pretty depressed. That was also the show that—is that the show that Apple announced Safari? I think it was. I think it was either that show or the next show where Apple announced Safari, and obviously, you know, our talks with them had broken down for a variety of reasons I won't go into, and they decided on KHTML, which was probably a good decision for them given the kinds of things they needed to do in their timeframe.

But you know, it kind of—it was kind of another blow that okay, Apple is doing their own thing, and they can, you know, add all of this functionality themselves, they can put all of Apple's great design resources and all Steve Jobs' great ideas behind this product, instead of helping us with ours. And they're going a totally separate way with their rendering engine. You know, and that's causing them problems even today because there are a lot of sites they just don't work in, even a lot of Google sites that don't work in Safari, you know, because it's just different enough from Gecko. And everybody has sort of accepted Gecko to the point that, you know, this is something you need to test against, which is nice, after all those years of hard work.

So we were all demoralized, just absolutely demoralized. And, you know, I drank for a week. And I kind of picked myself up at the end and I got an email—I still have this email—I got an email from my manager at the time, Chris [Ari], who—he said, you know, "You guys are doing a great thing. Regardless of what management says, you know, you guys are doing this wonderful thing and you have to realize, you know, you're building the product that you want. Safari is going to be the product that Steve Jobs wants. It's not ever going to be anybody else's baby but his. You know, if he wants a feature, it's going to be his way. That's not the product you guys are doing and that has its own place in the market." And that, you know, really sort of stepped me up.

And that was also—so Hyatt had already left the project, Hyatt went to work on Safari, which we all knew was coming, but we couldn't say anything until they had actually announced it. So, you know, so I—when Hyatt left I stepped up as project lead of that open source program, and to this day I still retain that title. At that point I was more lead developer. Now, I do some development, not a whole heck of a lot. You know, mostly it's just more management level stuff, you know, setting directions and determining, you know, what kind of features we want with the community, and taking community feedback.

And so, you know, we released a whole bunch of versions of that, you know, we released Camino 0.8 in 2003, I think, 2004. I don't remember. And you know, it had been a year or two from 0.7. And so what we did is, we took, you know, we took what was going to be Project X, which was based on something a little bit later after 0.7, and we said, okay, we have to finish this. You know, we have to keep this project going. You know, and part of that was me stepping up and taking some ownership and saying, okay, Safari exists, but we're going to build something different that has its own market. And enough people continued to believe that, that we had a sufficient community.

And we were—even though we'd lost the backing of Netscape and AOL, and we lost that dedicated QA, we were able to you know, to cobble together enough people to finish it and—I think that was 0.7. It was all so long ago. I think that was 0.7. So we cobbled together enough people and shipped 0.7. And that was very well received, and we said, okay, we've got something here. Enough people see that plausible promise, now let's take some—let's take this and kick it up a notch and do it right, in the open source community. We're not going to get any more help from Netscape, you know, I was off on different projects at the time, I was no longer on the embedding team. You know, let's build a product that is completely from the open source community.

And that was 0.8. And it was even better received than 0.7 was. Which was just a reinforcement that all of the stuff really does work. You know, when you have—I'm not trying to brag about this, but when you have the right people who can see the goal and can make sure that you don't go off into the weeds, you know, "No, we don't want this feature." "No. You know, we're not going to take this patch because it's going to destabilize us too close to the release." Someone at the top who isn't afraid to say no, this isn't right for our defining vision. The process generally works pretty well, you know. And that was also a model followed by Ben in the Firefox community, where you know, he was Firefox. He did a majority of the development work and he—

[*Sorry folks—the audio was abruptly cut off when our disk ran out of space. Many thanks to Mike for repeating several minutes of his discussion for us . . .*]

Olivia Ryan: Okay, so you sort of drew an analogy between Firefox and Camino's restricted access in the beginning. Could you sort of talk about that and how you might find that effective for running an open source project?

Mike Pinkerton: Right. So, when we started, you know, there's this period of time before any project can really become effective, and you don't want a huge number of people all banging on it. You know, you'll all end up getting in each other's way. And also, you know, one of the things that we were really I guess, rallying against, was the very laissez faire attitude of the Seamonkey tree for the last couple of years. You know, Mozilla didn't want to be—you know, Mozilla wanted to be inclusive, because we already had all of these problems with losing street credit by making decisions in the cathedral and not out in the bazaar. And so you know, we were desperate, Mozilla was desperate to get any kind of engineers that it could, any kind of engineering effort.

And so it developed into this culture of you know, just being all-inclusive. "Got a patch? That's fine, we'll take it." You know, doesn't really matter if it's going to slow us down, or if it's not a feature that we want, or if it's a feature implemented badly, you know, we'll take it because we're afraid to say no, for fear that you might, you know, get mad and run away from the project. And so after several years of that, after several years of not being able to say no, you know, what are you left with? You're left with Seamonkey, a product that nobody understands and nobody can use. Because everybody had put in their little pet feature that nobody wanted to say no to. A bunch of features are very poorly implemented, because nobody wanted to stand up and say, "You need to rewrite this," for fear of for fear of scaring them off. If somebody writes 10,000 lines of code for you, you're not going to say, you know, you're not going to say no. You know, that's a substantial effort on their part, and so how do you handle that? And Mozilla's answer was to always say yes.

So, we started off, both Camino and Firefox, with this notion of, you know, we need to do these products because we are in a particular place with Seamonkey. The reason we're in that place is because of their very, you know, 'always say yes' attitude. And so we had this knee—not knee jerk, but overwhelmingly opposite reaction to, "Unless you're absolutely trusted, you're not coming anywhere near our code." You know, and that led to lockdowns in CVS. That led to very restricted CVS access. That led to not necessarily even sharing a lot of the feature and design decisions with the community as a whole.

I think every project goes through this phase where you've got to get up enough to where the project can kind of sustain itself and where people see that plausible promise. And you know, and just having everybody going back to the old ways of just throwing everything in, everything would fall apart very quickly.

And so, both projects picked up on that. And in hindsight, probably a little bit too much. But we—I certainly credit both projects with succeeding because we started out that way. You know, both projects had a very strong leader who was not afraid to say no, who was not afraid to turn people away, who was not afraid to you know, if you made a patch that didn't work, to say, "No, rewrite this. We're only going to take good things in the tree." Or, you know, "I'm sorry you spent three days working on this feature, we don't want it. It's not the kind of feature that we want in our product. It's going to make things too complicated—" or whatever.

And being able to say no, and being able to stand up behind it, really helped shape both products to be—to be very focused. To be able to deliver on that initial promise of, we want to do something slimmed down, streamlined, easy to use, and the only way to get that—Steve Jobs knows that. He has singular control over everything that happens at Apple. And they develop really cool, really useful products, not because they have 700 people making decisions, but because they have somebody who knows how to make those kind of decisions and isn't afraid to do it.

So then at some point, you get over the hurdle and, you want to accept more people into your community. You know, you have to. You can't just do everything by yourself. And that's what I discovered, sort of after—kind of midway to the 0.8 timeframe, you know, after we released 0.7 as a Netscape project, and we were trying to release 0.8 as a Mozilla project. You can't always do everything yourself. And so who do you open up the gates to? You want to open up the gates to people that you can trust. Well, okay, how do you develop that—how do you develop that trust? How do you determine who shares your vision and then who doesn't? And a lot of that just like in a lot of projects, comes back to street credit. Has this person demonstrated that over the last period of time, they've acted in accordance with the goals that—the plausible promise goals that everyone has set up in the beginning.

And those people that do get additional—you know, they share in the—they share in the reward, because they share in the work, so they're part of—they're part of the team, they're not just out on the periphery. And the people that don't demonstrate that they know what's going on, they get pissed off and they leave. Well, you know, if that's the kind of devotion you have for the project, that, if we tell you, this isn't the right way to do something and you take your ball and go home, we don't want you on the project. That's not conducive to anybody working well together. It's not about—it's not about the person, it's about—it's about the code. It's about the project. It's not an ego thing, it's a 'we need to make the best project we absolutely can'. And people who share that vision, you know, are people that you want on board with you standing behind you.

Ken Albers: If I could just jump back one second. I find it interesting that you've talked about, I mean, essentially Camino and Firefox were just pulling the browser out of Seamonkey.

Mike Pinkerton: Mm-hmm.

Ken Albers: And, I haven't been able to get a hold on, you know, was this something that say, you and Ben were sort of talking about, and it was like okay, I'll do it for Mac, and he was going to do it for you know—or was it something you thought about at the same time and then you kind of—it developed in that way?

Mike Pinkerton: It was much more the latter. We, you know, so in 2001, we'd started doing these prototypes for Apple for, you know, they'd asked us to do Cocoa based wrappers around Gecko. And then in a similar kind of timeframe, the embedding stuff started taking off because of the AOL acquisition. And so, I think, it was Hyatt, or Hyatt and Ben—I forget what the exact timeframe was there. But again, Hyatt was bored and he wanted to learn C#, because C# was all the rage right around then. And so he went off and started doing a native front-end in C# for Windows. And just to play around, just to see what would happen. And I think Ben kind of joined with him on that and they started working on that, and maybe Blake got in, and that was Manticore, if you've ever heard that name before. It's a C# browser for Windows, just the browser, based on Gecko.

And so we all—after doing some of these prototypes and realizing, you know, taking a look at Seamonkey and taking a look at what we were able to do with native chrome and the amount of polish that we were quickly able to add to these products because they looked and felt native, it wasn't a far jump to say, you know, maybe we're onto something here. And so I definitely went that way for Camino because that's really the way that—I think the only way that Mac users will greatly accept a product, is if it feels like a Mac product. And despite being one of the, you know, founders of XUL, I don't think XUL is ever going to really work on the Mac. Just, you know, we can work on it for years, but it's just never really going to get that last couple of percent.

And XUL on Windows isn't really all that bad. Maybe it's that Windows users have a lower expectation of quality, I don't know. But I think they're willing to accept a little bit more deviance from the norm. And, you know, and since Ben was very good at XUL, Ben just took a look at the Seamonkey XUL and, you know, he almost passed out. And he said, "I can rewrite this better." And, what's one of the best ways is just jettisoning all the stuff you don't need. Who needs Composer? I mean, you know, how often do you need the weight—the fully loaded weight of an editor to surf porn? You don't need that—you just need something streamlined, something simple, you know, and simple is fast. With XUL, the more you load the slower you go. So, you know, it wasn't a very tough leap to go in that direction. And all of us kind of realized it about the same time as we were prototyping and playing with things and seeing the successes that we very quickly had. And we said, ah, there's something here. And we ran with it.

Olivia Ryan: Is strict ownership of specific areas of code enforced?

Mike Pinkerton: Right. Okay. So, in the beginning, it definitely was. The Gecko team owned Gecko. The XP Toolkit team owned XP Toolkit. The XPFE team owned XPFE—layers up the chain. And that ended up causing no end of grief. Like, the Gecko team, the Gecko team would only test their little wrapper around Gecko. So once you—we would take the code and build it in on top, wrap it with XUL and make a front-end with it, we'd encounter bugs in the rendering engine because of where it was in the layout chain. And they didn't care, because they only cared about Gecko, you know. The tag line was, 'works in Viewer' because they had this little app called Viewer, which would run through HTML—just pure HTML tests in a native frame. And they'd say, "Well, it's not our problem, it's not our bug. It works in Viewer." You know, and we couldn't go in and fix bugs because they owned the code and they were very protective about it.

So, you know, and a similar thing sort of happened with the XPFE team and the XP Toolkit team. So there was lots of gnashing of teeth and butting of heads on both sides, and, as I said earlier, you know, I would love to go back in time and be able to change those relationships, because they really drag the product down as a whole. All of this kind of infighting and bickering between the teams didn't really lend to a very productive work environment for anybody.

And a lot of that was also because we were building the foundation while we were building—or you know, we were building the house while we were building the foundation. And, so that's going to cause an infinite amount of stress for any groups. So, we had this very strong ownership and we also, at that time, Mozilla was—Netscape owners were the only people who had any kind of ownership in the source base. Because there weren't a lot of developers in the open source community. And so, pretty much all developers were Netscape developers. Or pretty—all module owners were Netscape developers.

And that ended up causing us some problems when we would come across module owners who didn't really want—who didn't care that they were a module owner, who didn't want to be a module owner but because they had this position at Netscape and they were grandfathered in, because they owned the code beforehand, you know, they suddenly had this responsibility to the community. And maybe they didn't want it. There are people in every company who—they just want to work nine to five and go home and punch a clock and collect a paycheck. And you know, we certainly had some of those at Netscape, because they exist everywhere. And the problem is when those people are in module owner roles, there's more expected of them than just punch a clock nine to five. You know, they have a responsibility to the community.

And so, I don't remember if I said it before, one of the things that we discovered and one of the things I discovered with the strong ownership in Camino, is, a weak owner is worse than none at all, in that, you know, if you have an owner, the community is going to look to that owner to make decisions. The community is going to look to them to say, "is this good? Is this bad?" "Can you review this? Can you not review this?" You know, you are the gatekeeper, everything has got to go through that owner. And if that owner doesn't handle that responsibility well, they either just ignore it or they're very, kind of lackadaisical. The other is—they let everything in which is kind of the general Seamonkey problem, you know, most owners were so swamped they just let everything in. And that doesn't help anybody.

And the community, because of the way that open source communities put emphasis on reputation and on the street credit—you can't just easily go in and depose a module owner. You know, you can't just say, "Okay, well you're not doing your job, we're going to take over." It happens sometimes but it doesn't happen that often, because people—people want to believe, this person has attained this role in the community through some valuable means. And unfortunately, in the Netscape case, many of them obtained that role not through coming up through the community but they were just placed there because Netscape started the project. And that was kind of the fatal flaw of that entire argument. And so, there were times when we had to say, "Look, you're not doing your job, somebody else has got to do this, otherwise this whole thing is going to fall apart." And generally they would be like, "Yeah, that's fine. You know, I never really wanted it anyway. Thanks," you know.

So, you know, more—more recently, a lot of the—a lot of the butting of heads has come from—historically, like at Firefox, you know, historically Ben was the module owner. You know, he set the vision, he determined, you know—he set the feature set. He controlled how everybody is going to do everything, he—he was the one defining vision of that project. And so he understands the history. He understands all of the reasons that things were done. And so now, the Corporation, or the Foundation, sort of owns shepherding that project. And there's a group of people who have been hired by the Corporation to be in charge of making those decisions, because, you know, they've decided that this is a money-making venture, we need to make sure this is done, and done correctly and done by people we put in, you know, that we want making these decisions.

And so, how do you determine between Ben, who was the genesis of this project and put his blood, sweat and tears into making it as popular as it is, versus this other group that controls not only the purse strings, but control of the organization. So how do you—how do you kind of decide on those trade-offs? How do you make the right decisions there? And I think that's one of the things that ultimately caused Ben to leave the Foundation and to contribute from outside of Mozilla, just because, he and the group put in place to make the decisions, would disagree. Like, a lot of the tab browsing improvements that Ben wanted to make. The Corporation, the project managers of the Corporation wanted to do totally separate things and that caused Ben a lot of stress because in the past he had been this—such a strong owner for the project. And able to make the decisions and able to say to people when they do something wrong, "No, we're not going to take that." And suddenly that power was kind of wrestled from him. And he, you know, he's no longer able to be that strong owner to say no. And I think that really weighs fairly heavily on him.

Olivia Ryan: As the Corporation and the Foundation grows and they're sort of influencing, as you've mentioned, decisions regarding Firefox, how do those decisions impact the other projects and/or do the Corporation and the Foundation influence decisions on other projects directly? Or is it sort of like indirectly, because of their influence over Firefox? Does that make sense?

Mike Pinkerton: Yeah, that does make sense. So there is a lot of direct impact. There's also a lot of indirect impact. You know, certainly—certainly from the Camino perspective, I sort of see this being at the end of the stick. People will make changes to do something good for Firefox and because Firefox is the only really blessed project by the Mozilla Corporation and the Foundation, that's their only focus. So they'll make changes that work great in Firefox and then they'll do—they'll either like, break Camino's build just entirely, or things will stop working, or things will slow down, and nobody will really understand why. Because, you know, there was no communication about, you know, this change might have this effect. There's also a lot of strife back and forth between features that are implemented in the core Gecko in a way that they will only work with Firefox. Or they'll only work with a XUL based browser. So, like, single window mode, which is one of—something really popular in Firefox—the way that they got that to work would not work in Camino. There's nothing that we could do to make that work in Camino because it assumed there were pieces above it that would make decisions that we didn't have. There's no way that we can just put them in without pulling in all of XUL.

So we were left with, well, how do you answer the question, "Well, Firefox has this feature, why can't you? It's a core feature, right?" "Well, not really." So we'd run into situations where we try and implement a feature and we discover we just can't do it. And we kind of raise our hands and say, "Ah, can we get this fixed?" And the answer would invariably be, "Well, it works in Firefox. Who cares?" And to their credit they are a lot of people on the back-end team who make a lot of core Gecko changes, who—they really want to do the right thing. They, you know, they're not out to screw any project, they're not out to short any project. You know, they tested Firefox because that's what they use, because they're Firefox people, and they might be on Windows only, they don't have a Mac, so they can't even test Camino. But then when they do do something that breaks people, there's really no recourse for any of these sort of second tier projects because the Foundation hasn't made it a priority. The Foundation has said, when you check in you're only responsible for making sure Firefox stays building. Everything else—if everything else stays building that's gravy, but if it breaks it's not your problem. It's that other project's problem.

And that's something that eventually made me sour greatly on the Foundation and the Corporation and why we kind of took our ball and went elsewhere, and stopped trying to work directly with the Foundation for the majority of problems. What we would do is, we'd just try and communicate directly with the developers, and just educate them on you know, "These projects do exist out there, there are these other projects, you know, we'd really appreciate it, you know, please, please, please, you know, keep us in mind, or at least just us know if you're going to be making these big API changes, at least when you're making them, think about embedding. Think about non-XUL applications and how this might work there, and even if first round it doesn't work but you've got a plan to make it work, that's awesome, you know, that's all we can think of. Just don't break us." Because it's not anybody's priority to go and clean things up.

And a lot of that just comes down to, you know, on our part, picking which battles we want to fight. You know, stuff is going to go wrong all the time, that's how development works. Things will break, that much is guaranteed. How much can we—can we mitigate that breakage by staying on top of nightly builds—and this is where our community really comes into play, by downloading the nightlies and making sure nothing has regressed. Doing smoke tests. And then when things do go wrong, we have a very narrow window of time to put the blame on—so we know yesterday's build worked and today's doesn't, so 24 hours is the area of check ins that we need to look at. We don't need to look at the last six months because we know it broke within the last 24 hours.

And so that's where the QA really becomes much more beneficial than in a normal project where maybe we would be the first tier and it would be everybody, you know, everybody on staff to make it a priority to fix things. And so it's about picking our battles when it comes to things—when things break. We can run around like chickens with our heads cut off every single time something goes wrong, but at the end of the day is that really all that productive? You know, we need to, you know, if someone slows us down 20 percent, yeah, we're going to get pissed off and we'll probably send off some email. But if somebody slows us down one percent, or they break the build and they realize it two hours later and they fix it, you know, that's not worth getting mad over.

Olivia Ryan: To what extent does Mozilla rely on the work of volunteers and has that reliance changed over time?

Mike Pinkerton: Can I get some water? You don't need to stop it, I'll just keep talking and whoever— Thank you. Sorry. So, volunteers. So in the beginning, there were very few volunteers. Pretty much, like I said, everybody who was doing the contributions were hired by Netscape, were Netscape employees. And usually as people in the community would get better and would start to contribute significant amounts of code, we'd just hire them. So, we'd start pulling everybody inside the fence, and that didn't really lead to too many external contributors. I mean, it was an incredible way to recruit talent because you had to prove yourself on the job before we hired you. We hired a lot of really good people that way. But it didn't, you know, then those people became tainted I guess, by being under the same Netscape AOL control as all of the other engineers. They no longer had the ability to say to put on their Mozilla hat and say, "No, we can't do this, it's wrong for the product," because the people paying their paychecks said they had to do it.

So then over time, we definitely shifted to a lot more external contributions. As Netscape started to get whittled down over the years, between people leaving and the Black Tuesday layoffs that seemed to come every year, the project had to rely more and more on external contributors, otherwise you know—because there was nobody left at Netscape to actually keep it going. And this is where, you know, and this is sort of where opening things up to you know, to determining who the right people to open things up to—when you have projects that are very strong. You know, I guess to some extent, Netscape had fairly strong ownership of Seamonkey in that it had paid marketing people who, it was their job to decide what went into the project and what didn't. So, somebody ultimately had the say over, you know, this is a feature we're doing—this is a feature we're not doing.

And some people in the community could argue, and they could say, well, you know, if this isn't right for Mozilla it can go into the private Netscape tree and only ship in Netscape products. And that happened sometimes. But a lot of case, Netscape really decided what went into each individual product release. And then once that started winding down, we entered this era of you know, more community involvement, because suddenly there was nobody around at Netscape to keep doing the stuff. So other people had to step up, people who weren't Netscape employees. People would step up to help make decisions. People would step up to help you know, to help triage bugs. A lot of developers—most of the developers who are contributing today are not—were not Netscape employees. Or, I guess, generally, if they are, now they contribute through Google. You know, Ben and Darren, and Briner and all of them.

So, the project came to a point where it absolutely had to rely on the public. And that's why—I think that's another reason why Seamonkey sort of floundered a bit, in that Netscape had control, and then it relinquished control, but it relinquished control of this product that never really had great decisions made about it to start with. So, you ended up with this turd that nobody wanted. And then, and there was no real strong ownership over that, so it just kept getting worse. And that's what kind of started me and Hyatt and Ben and some of the others, to, in parallel, come up with—okay, "Well, this isn't working. We need something that is working." And so for a short time we were able to do everything ourselves, you know, the small projects, just getting started. It's very easy to do everything yourselves. But then, you know, you reach a point fairly quickly where you need more help than just you are able to provide. You're certainly, even in the realm of QA, you know, without the QA volunteers for the project, the project never would have gone anywhere. You know, they're the saving grace of the whole Mozilla project, I think. They are—they are absolutely the most valuable resource of the entire project.

You know, developers are great, but the QA people really, they lent a hand when they didn't need to, when nobody cared—you know, when nobody cared that they were lending a hand, they lent a hand. And it's not that things just got popular and QA came back. QA was always there. It was sort of developers that ebbed and waned and flowed. You know, so, just that entire community deserves incredible kudos for kind of keeping the project—for getting the project to where it is today.

The Camino QA contribution is invaluable. I can't thank those people enough. Certainly Firefox got a lot of help along the way to their 1.0. Because people would see—people saw that plausible promise and they said, "I can help you get there. I can't code, but I can download your nightly builds. I can help you regression test. I can help you speed test. I can help you narrow down bugs. I can help you take a bug and whittle it down to a reproducible test case, so it's not just this amorphous thing." You know, these are all things that take non-trivial amounts of time, in terms of commitment.

Olivia Ryan: So why do you think people volunteer? It takes like a substantial amount of time and a lot of people are doing this while they're working full time, and—

Mike Pinkerton: I honestly don't know. I wish I had a good answer for you. You know, when you think about—when you stop and think about it, what they're doing is crazy. You know, there's no reason why they should be doing any of this, because again, they're not getting paid for it. It's a significant time investment. It's generally very thankless. Now, I try and thank my QA people as often as I can remember to, but I know it's not one-tenth of enough.

Olivia Ryan: Do want to stop for a minute?

Mike Pinkerton: No, it's okay. But so, the students in my class ask me that all the time. They say, you know, "So why do people do this?" And I think it really comes down to, they see something they believe in and they want to contribute to it. It's not so much that—so what is the reward for them? And Eric Raymond talked about this a lot in *Homesteading the Noosphere*. He laid out, sort of, what is the reward system in the open source community? And a lot of it is kind of anthropological bullshit. But in some—if you kind of boil it down to its core, there's no monetary reward. There's—so, what else is left? Well, there's street credit. You know, there's respect from your peers for a job well done, and that definitely goes a long way. And you can parlay street credit into tangible things like, you know, a teaching job, or a job at Google, or a job at Netscape, or you know, coming and talking to you guys.

So you can—you turn those into tangible rewards. But, really at the end of the day, you're living in this kind of gift society where the gift, the reward, is credit, is street credit; is the admiration and the respect of your peers. And for a completely digital society, people—people who, you know—I very strongly respect people I have never, ever met in person. And that's kind of an interesting thing, you know, that I've interacted with these people just through this project on IRC or Instant Message, or just even through bugs sometimes. Some people don't even come on IRC, they're only in the bug system—that I have respect for them. And if anybody ever came and asked me about them, I'd have this litany of really wonderful things to say about them. And I've never met them. I've never spoken to them. I've never heard their voice. And I think there's really something worth exploring there, if you know, if you're looking for research grants.

Olivia Ryan: Do you consider open source projects, either—Mozilla in particular, or just sort of open source software projects in general, as a public service? Have you ever sort of thought about it that way?

Mike Pinkerton: I have actually, I have thought about it that way. It is—I think that's sort of why I do it, I think. Well, not in those words, because that sort of makes it sound like I'm doing something noble. But the main thing that got me interested in doing Camino, and there have been a lot of hardships, you know, it hasn't been easy. We've been second and third class citizens in this open source community, that's already a second or third class citizen. What makes me want to do that? What makes me continue to do this even though I'm heinously busy, I no longer have the full control over things that I used to, because I just don't have time. Is the—four years ago, or whatever, I saw something in this plausible promise that Hyatt and I put forth, that we can do something that can make some people's lives better. That we can fill this gap, this need that people have, just through doing things that come naturally to us.

And what is the reward for me, I guess? Well obviously, I've turned that into financial gain through various jobs, but the real reward for me is reading—we have a Camino feedback mailing list. And it gets probably—so if somebody wants to give feedback you can either click on the website or through the help menu, I think—help menu feedback. Just send an email and it goes into this kind of, you know, no name bulk email at MozDev or MoCo, I think. It probably gets about 40 or 50 emails a day, sometimes a little less, maybe 30 or 40. Ninety-nine percent of them are overwhelmingly positive. You know, this is from an Internet culture where if you gave everyone free cars, they'd complain they had to pay for gas. People on the Internet don't stop and take the time to say nice things. Generally all they want to do is rant and flame and when they have a problem it's the biggest problem in the world and everybody on the planet has to stop and listen to them vent about it. People don't just stop their daily lives and say thank you. But for some reason the Camino mailing list is full of overwhelmingly positive thank yous.

People will just say, "I stumbled on your browser randomly. Thank you so much. It's so great—I never knew—I don't know how I lived without it." Holy shit! You know? How can you not enjoy that? And I think that's really what keeps me going and what keeps me believing that we started off with the right goal, and we're making the right choices to execute on that. Because without execution all of those emails would stop. And that's part of the reason why we don't release every, you know, two months or whatever, is because we've set this bar for ourselves as—w e've set a very high bar. You know, 0.7 was really, really good, you know, given all the shit that was coming out of Netscape at the time. You know, Camino 0.7 was really good. 0.8 was even better, you know. 0.9 was even better. 1.0—nobody has anything bad to say about it. So we have this very, very high bar for ourselves, and just making the choices to maintain that high bar, because I know that as soon as, if I decide to step away and put this on a shelf, say, "Yeah, I don't really want to do this anymore. It's been a good ride," will we be able to keep that high bar? Probably, because I think I've filled the roles around me with the people who know how to make the same decisions that I probably would have made. At least if not the same decisions, then with the same mindset, you know, with the same background, and bringing the same reasoning to it.

They may make different choices, but they're the right choices for, you know, for the way that the project has grown. And so I think that's what kind of keeps me doing it, is I want to be involved in that process that gets, 99 percent people who stop their daily lives to say 'thank you'. You know, is that a public service? Ah, hard to define. But, it's certainly rewarding.

Olivia Ryan: You've talked quite a bit about sort of how Mozilla has changed over time in a variety of ways. How, like, specifically do you think Mozilla's priorities have changed since 1998?

Mike Pinkerton: Do you really want me to answer that question? I'm trying to think of a politically correct way to say this. How have they changed? So, they've changed both in good ways and bad ways. You know, initially, initially their priorities were, "We just did this thing and it's really great, how do we keep it going?" And how do we make sure that we don't turn into Netscape? You know, and unfortunately with Netscape driving it, that was fairly difficult and they made a lot of wrong decisions—or Netscape made a lot of wrong decisions and those impacted negatively on the Foundation, which is very unfortunate, because they weren't the Foundation's decisions. And so trying to recover from a lot of those early setbacks, the priority was, "We got to keep this thing going." You know, so they tried developer education. They tried reaching out to developers. They tried doing evangelism. And none of that stuff really worked, again, because they didn't have the street credit.

Not that those things weren't necessary, because you had to have those things for developers, but it just took time. And then, kind of, they lucked into Firefox. I mean, seriously, they got really lucky that Firefox—that Ben was able to execute so successfully, because I have no idea what would have happened to the Foundation without Firefox. It probably would have just blown up and you know, dried up and blown away.

And so then for a time, their focus was "We've got this thing called Firefox. Let's run with it and see how far it can go." I don't think anybody really knew that it could get as big as it did. A lot of people really latched onto it. A lot of really smart people latched onto it, and were able to do good marketing campaigns and kind of take it to the next level. The fact that you know, Blake ended up on the fucking cover of Wired, I mean, that's just insane. You know, that this little project ended up being so big in the community, especially given its roots, you know, basically rooted in failure, in Netscape's failures.

And so they kind of transitioned from that to, "Well, okay, now we're a corporation and we've got our big ticket item, and we've got to make sure as many people buy it as possible. And you know, and how do we continue to—how do we continue to ensure that revenue coming in?" You know, so they transitioned more from "Wow, we've got this great thing, let's share it with the world!" To, "Well, we've got this great thing, we'd better keep making money off of it." And I think a lot of those decisions have sort of tainted where the project was going. Bringing in—you know, they've got all this money, so now they bring in all of these marketing and manager people who haven't been part of the community, who don't understand why Firefox is successful, because they didn't see what it was. They didn't grow up through that failure to see why it's a success and now they're making the decisions to continue the success. And whether or not that in itself will be successful, I don't know.

Olivia Ryan: Do you think marketing people were necessary? Or do you think sort of, developers could do the marketing themselves and through things like Spread Firefox, through, you know, through engaging the community? Or do you think that once something gets to be so large, it's beneficial to have professional marketing people?

Mike Pinkerton: I think so. I think there was—both sides were necessary. For Firefox to really be embraced by the masses, by the, you know, moms and grandmas and such, who never would run across Firefox on the Internet, regardless, there needed to be some professional marketing. You know, mainstream, get in your face kind of marketing. You know, print ads in Time and Wired and all of those things. You know, those kind of get in people's faces who aren't necessarily—well I guess Wired to some extent, but you know, Time Magazine is a pretty big one. You know, to getting them saying really good things about Firefox, is a huge coup. You know, and especially, you know, to do things—you know, you need a more formal type organization to really focus the energies, to rally around the security aspects and the, you know, 'look at what we're doing to Microsoft's market share for the first time ever!' People who can sit back and have a—you know, this is their job and they can help guide those types of things.

Because end developers aren't going to evangelize like that. They're not going to see the strategic, you know, the strategic angle to it.

But for it to be accepted by the open source community, by the grassroots community that sort of helped it build to what it became, it absolutely needed that grassroots marketing. It needed the Spread Firefox. It needed the, you know, putting the Firefox—download Firefox banners on your websites. You know, that really helped keep it real, you know, keep it grassroots. Even though they were off doing all these you know, glitzy print ads, they're still true to the community that got them where they are.

Olivia Ryan: So, why do you think other projects have not been marketed in that way? Why do you think the focus is solely on Firefox? Or do you think that perhaps other projects will be marketed similarly in the future?

Mike Pinkerton: Well, so, I think there's a big difference, there's a big dividing line between the types of projects that are out there in the open source community. So, the majority of them, I would say, are—tend to be more kind of back-end oriented, you know, like your Apaches and your Perls and your, you know, Pythons. And all of those type of not really user-facing. You won't ever see a print ad for Apache. There's no need. That's not the way you get that market and my mom doesn't care what web server you're running. Sun tries that a lot. Sun and IBM market their web servers and all that stuff. But you know, it's unknown how much it actually impacts things.

So, there's kind of that group who mainstream marketing probably wouldn't be all that beneficial. There's also the other group of projects like Open Office, you know, more user ended like GIMP, user-focused type applications. Things that would benefit from broader marketing. "You don't like Office? Here, use Open Office." You know, that makes sense. The problem is, most of them suck. The one thing that really made Firefox mass market worthy, is that it didn't suck. Ben made the majority of the right decisions to take exactly what the people who were fed up with IE wanted, add some extra things like tab browsing, which kind of revolutionized things. Build on top of Gecko security in the standards compliance, and make something that, you know, people really do want to download. I wouldn't recommend Open Office to anybody. It's crap. I mean, no slight to the people who have been slaving on it for years, but it really hasn't come to—you know, I certainly wouldn't use it.

Olivia Ryan: What about some of the other Mozilla projects, like, could there be a Spread Camino, or—something like that.

Mike Pinkerton: So, I think it would be—it would be nice. It would be beneficial to some extent. Because one of the major complaints that we hear is that people just don't know about the project. You know, they go to Mozilla.org, they hear about this thing, and they're just bombarded with "Get Firefox". And so our primary—probably our primary roadblock to mass acceptance is just not enough people know about it. It's interesting though, I hear from people sometimes—they'll go into an Apple store complaining, you know, they go to the Genius bar complaining about Safari or something, and the Apple Genius will tell them to download Camino. You know, that's always fun.

But so why I don't really think that would be all that effective is that even if every single Mac user used Camino, that's only, you know, three percent of the world. Three percent of the computer market. And given the fact that Apple's—Safari is going to take sixty to seventy, Mac IE still have a very large, strong hold because people don't like to change. Firefox is going to take up a significant amount because of the marketing that they get. So we're less than one percent of three percent. There's not a lot of spreading to go around. So it would be nice, you know, it would be nice to get visibility for—I'm not just doing it because I want to get my name out there and my product. The team has worked pretty hard on it and it would be nice to get them their recognition. But ultimately, would a Spread Bugzilla help? Probably not, because, you know, Bugzilla's great, but the people who really need to use it already kind of know about it. I would hope to think, at least.

And Spread Seamonkey? Yeah, right! That'd be a fun one to write copy for. "Sucks less than it used to!" I forgot to tell you this really great anecdote. When, you know, we'd just shipped Netscape 0.6 and it sucked, and then we shipped 6.1, and it was leaps and bounds better. It was just so much better. You know, and then we shipped 6.2, and it was even better. And so you know, we're kind of getting this feeling like, all right, we're back on track and we're not just shipping crap anymore. Scott Collins, a really good friend of mine, the guy in the cube next to me for years, he says to me—because we were still getting panned, you know, it still wasn't this great product but it was so much better than it used to be. He says, "You know, I feel like I'm finally starting to strike out with a higher class of women!"

All right? I can go with that. I can go with that.

I don't remember where I was.

Ken Albers: If you don't mind I just want to jump back to something you were talking about earlier.

Mike Pinkerton: Sure. Absolutely.

Ken Albers: When you were talking about the changing priorities in Mozilla, you know, you said it started out, we did something great, how do we keep it going. And now you—it's more how do we keep making money. Do you see those as opposed, necessarily, or do you see that maybe then, you know, you talked about it a little bit, about, well, they're making and now they're hiring middle management and maybe—and then sort of, that's more the problem than, you know, that—

Mike Pinkerton: Them focusing on making money.

Ken Albers: You know, making money could in turn make better products. Could in turn be bigger market shares. You know, things like that. I was wondering—

Mike Pinkerton: Yeah, so you're right. Those two aren't necessarily opposing. There is obviously—they need to make some money to keep the whole thing going. You know, there's infrastructure, there's all the Tinderbox machines, there's the Bugzilla machines, there's the local network at the Foundation. There's the salaries of the people that they employ to, you know, to do good work. So they can't just live on handouts alone. So, yes, to some extent, they need to—they need to find a way to be self-sustaining.

Whether or not—whether or not their current decisions are in that vein, or if they're in, you know, 'let's try and capitalize on this as much as possible', I don't know. The whole reason for them starting the Corporation are varied and legal. But you know, I mean, obviously they didn't just start it because they say, we have this cash cow, let's go, you know, let's go cash in on it. That's not the reason they did the Corporation. I just fear that with this influx of money they're not spending it as well as they should be, I guess that's the better way to put it.

Ken Albers: More decisions made by people without the vested interest, things like that.

Mike Pinkerton: Right.

Ken Albers: Okay.

Olivia Ryan: What, if anything, do you think the popularity of Firefox will do for the open source software movement as a whole?

Mike Pinkerton: I think it does a lot. You know, like I was saying earlier, a lot of the end-user type open source applications, you don't want to use them. Nobody can figure out how to use GIMP. I'd rather use PhotoShop. I mean, at least I can barely figure out PhotoShop. I can't figure out GIMP. Open Office works sometimes, you know. And so the fact that it's made it to the mainstream, that—people have even sort of told me that, you know, another reason why we should get out, we Camino, should get out from the Mozilla umbrella, is because nobody recognizes what Mozilla is anymore. They know Firefox but they don't know Mozilla. Like, someone even suggested that we rebrand ourselves as not a Mozilla-based browser but as a Firefox-based browser, which could not be further from the truth, but it gets people—it uses words that people understand.

And to see such a very public acceptance of this product in the mainstream media, even just by people that really don't even understand open source, for them to be using it is this great, kind of bridge to this whole new audience that most open source projects never, ever come close to being able to bridge. And so once it becomes a household name, you know, understanding a little bit more about it, about the open 'sourceness' about what makes open source, about what makes a community, isn't *right* behind it, but eventually that starts to seep over as well.

You know, I don't think the mainstream media—I haven't read most of those articles. But my guess is they probably don't explain open source as eloquently as they could. But it puts the idea in people's minds. You know, they're reading about it because they're interested in this product and that kind of comes along with it for free. And then that opens the door for other products, for people to say, "Well, you know, Firefox wasn't so bad. Maybe I'll give this other open source thing a try," if they hear about it. And hopefully there will be others who are of that level of quality, who can bridge that gap. I'm sure it's only a matter of time. Firefox was a stroke of luck and a stroke of genius at exactly the right time for, many aspects of the community. Can that be repeated? Probably. It's just, it's not easy.

Olivia Ryan: What about applying open source principles to things beyond software? Like, do you see open source techniques in other areas of production within society? Or do you think that that way of producing something can work for other things beyond—I mean, we've already sort of seen it in marketing, for example. Do you think it's as effective or do you think it's a little bit more specific to software?

Mike Pinkerton: Well, one of the—I mean, one of the huge benefits of open source is that you're working with things that aren't tangible. It's not like you're on an assembly line with nuts and bolts that 20,000 people can easily pack into that factory. So you have this very distributed workforce that also, in a lot of the ways, and Eric Raymond sort of describes this as well, doesn't suffer from a lot of the mythical man-month problems, where throwing more people at something that's late makes it later.

Because of the 'distributedness' and because of the way that different people can work in small groups to affect the whole, sort of separately, and not getting in each other's way. I'm not sure how necessarily applicable it is. So, you know, something like marketing definitely there's no real tangible, physical things. Would it work in industry? I don't know. You know. Could you build cars open source? Probably not. I don't know. You know, one of the other benefits is that, you know, to open source, there's no—there's so much—what am I trying to say. So there really aren't any restrictions. You know, bandwidth isn't a problem anymore. Disk space isn't a problem anymore. You know, so there are no real barriers to entry in order to help out with software. And I don't know if those barriers would exist for other areas. No—that's a really good question. I haven't thought too much about that, obviously.

That's another grant right there! I tell you!

Olivia Ryan: Well, this is the last question I think and it's extremely broad, but if you have any thoughts we'd be interested in hearing –

Mike Pinkerton: Blue! No, green!

Olivia Ryan: What you think the future of open source is? And it can be specifically for Mozilla or Camino if you want or just sort of broadly speaking.

Mike Pinkerton: I think despite early failures, especially in Mozilla, Mozilla was a very big public failure at the start, it's come a long way. Firefox has helped it a lot, and will continue to help it a lot. It's also something that's not going to go away. I believe it's reached enough critical mass that between Mozilla and Linux and Apache, and some of the other really big open source communities, there's not really any chance of it drying up and blowing away at this point.

I think that's really good for the software industry. It's really good to have checks and balances. That's kind of one of the other reasons why I feel good about doing Camino even though Safari cleans its clocks in terms of user base, is someone's got to keep Apple honest. You know, someone’s got to keep them pushing the envelope, because if they don’t have any competition, even if we’re small competition, we’re still competition. We still make a splash to them. And somebody’s got to keep them on their toes.

And Firefox certainly helps Microsoft doing that. I mean, Microsoft had shelved all browser development—they’d given up, they won. They moved on. And now they’re releasing IE 7. They’re putting all this effort into it because there finally was a viable competitor. And I think that, you know, keeping everyone in the industry honest, you know, it helps competition. It’s the whole thing behind monopolies. If you just have a monopoly, there’s no impetus to innovate. And we absolutely saw that with Microsoft.

And whether or not all these little projects will ever become as successful as Firefox, it’s not really—not really here nor there. I think it’s the, there is still this growing community of people who want to donate whatever they can to something that they believe in. And again, it goes back to the public works kind of public charity thing. It’s a donation of blood, sweat and tears, and that’s not going to go away either.

Olivia Ryan: Great. Well thanks Mike.

Mike Pinkerton: Sure.