

Tom Scheinfeldt: Okay, if you'd just say your name.

Asa Dotzler: Asa Dotzler. And I've been with—or should I wait? Okay. And I've been with the project since 1999. I actually attempted to get involved, right, in early '98 when the source code was released. But I wasn't a developer. And so I failed miserably.

And I spent the next six months trying to find my way into sort of the core of the project. So, that meant going through the news groups and trying to keep up with what was going on, trying to find actual binary so I could download and test.

In the early days of the project, there wasn't anything. If you weren't a developer, and didn't have the ability to pull a source code and compile it into a build, you didn't get to play in the game. And so I spent about six months trying to get at that. And a few friendly developers would make some of these executables and put them up on FTP servers. And then it was the hunt to find those.

And did Mike Weinholt [ph] who had one two months ago, has he put up another one yet? And is it going to annoy him if I send him an e-mail wanting to see, you know, what was going on with this development project?

And there was actually a website that was born and sort of catered to that around me and four or five other people who were interested in getting at those. The site was called MozBin. It was the—in a source code tree, when you compile a build, you go to the “dash slash bin directory” to get the program you actually run and “Moz” for Mozilla, and the site was started by a fellow named Jason Kersey. And he was like me. He wanted in on this because it was open source and it was Netscape. And that was sort of two things that we were both interested in coming together. But neither of us were developers, and in those days, open source really did mean developers.

And if you look at some of the—even some of the famous tomes that have come out of the open source world where you see things that sound pretty open, you hear, you know, ‘with a thousand eyeballs, all bugs are shallow’. But those eyeballs were developer eyeballs they were talking about, right? They weren't even talking about, you know, testers and things like that. And so I realized that it was all about developers and their co-developers. And I was going to work my way into it because it was interesting enough to me.

And so Jason set up this site where he would—he and me and four or five other people would figure out where that latest testing program was and link to it so that others like us could come along and test those. At the same time, a site popped up called MozillaZine started by Chris Nelson. And Chris Nelson had an interest in Netscape and in open source and decided he'd put together sort of a new site where they'd have, you know, weekly updates on what was going on with the

project. And that site actually grew pretty quickly and became one of the places that the ‘not core’ development team could find out what was going on.

The press would come there to figure out what was going on. People who wanted to—other people and other open source projects who wanted to figure out if there was an opportunity for them to get involved would go there, and general discussion about the state of the browser wars and the decline of Netscape and things like that.

And then two of those sites actually merged. MozBin and MozillaZine merged. Jason and Chris joined forces and created a new mega site where mega means it’s got, you know, hundreds of users instead of dozens. And, actually, no, it was more significant, I think, within a couple of months they had several thousand registered users.

And around that time, I finally got into some pretty consistent communications with developers about certain problems in the browser. And they—convinced them that I was valuable, that I would help them find the problems they were introducing, or problems that affected their areas of code. And I started to learn how to create a good bug report that would be sufficient information for them to work with.

And they had an open bug database, but it was really, again, just developers using it. And when I first went in, the first three or four bugs I filed, I got kind of slapped down for not providing enough information to be useful and wasting their time. But I sort of just absorbed that and said, “Okay, what should I do next time?” And I got to be, I think, a valuable resource for them.

And then Netscape decided it was going to start pushing a little bit more to grow the audience for these Mozilla, early Mozilla builds in anticipation of upcoming alphas of what would become the next Netscape release.

At the same time, Netscape also, under pressure from sites like the Web Standards Project, which was a new site and trying to get in all the browser vendors to adopt a core set of Web Standards as defined by the W3C and through campaigns at MozillaZine, you know, the petitions, online petitions convinced Netscape that they ought to scrap all of the old code they had, which was Communicator 4, the next rev of the Communicator 4 code base, so it would have been Communicator 5. And so that was like—I think probably like November of ’98, just as I was sort of finally punching my way through. And they did. They threw it all out and started over.

And in the early parts of—in the early parts of ’99, they released this new project based on Gecko. And it was a cross-platform code base, which meant that instead of having three different teams, a Mac team, and a Windows teams, and a Linux

team, they had sort of a single team working on one browser that would run on each of those operating systems.

And a lot changed with that. They moved to this new front-end architecture that was, you designed the buttons and the toolbars and the menus of the browser pretty much the same way you would define buttons and menus and toolbars on a web page using XML instead of HTML, using JavaScript to define the actions, the same that you get for dynamic web pages, and using images in CSS to define the look and feel.

So, now people who weren't traditional application developers, but web developers, could come in and learn pretty quickly how to write a feature on the front end of the browser.

So, the developer community started to grow a little bit, and all of the people who had succeeded in convincing Netscape to throw away this whole code base, a lot of them it was because they wanted to work on core rendering stuff, but that was never going to be able to be made standards compliant. It was a huge mess of spaghetti that no one could decipher and untangle. This new simplified Gecko rendering engine was easier to digest. So, a lot of those people came in from the community and, as sort of a thanks, put in some effort to make this new rendering engine, you know, to bring it up to speed with the state of the web.

So, it was really in '99, a year after the launch of the project, the spring of '99, that things started to take off. And, by that point, enough people were coming into the project. Bugzilla had gotten enough sort of visibility through MozillaZine and other websites that the number of sort of casual testers were coming in and reporting those same low-quality bugs that I had in the early days. And developers were getting, I think, more and more frustrated by the degradation, the signal to noise ratio in Bugzilla, in the news groups and elsewhere.

And so, I decided to take it on myself to go in and be a layer between the developers and a lot of these new people. And so, I would go in and, you know, when someone filed a bug, they didn't have enough information, I'd beat the developer to that one and say, "Here's what you need to get in here and help them get it in."

So, I did about six months of handholding new contributors, and we went from the—I don't remember the numbers too well, but probably ten to fifteen regular volunteer testers, to a year later probably two or three thousand regular volunteer testers. And that was something that you know, I think that few people, and myself included, were pushing hard to try to lower the barriers to entry to open source participation so that it wasn't just an engineering task but that we could find ways for other people who wanted to be a part of the community who understood and appreciated this open source process, which was really appealing in sort of a collectiveness and communalist approach of let's all work toward a

common goal and leverage each other's resources, the massive parallelism and all of that that appealed to a lot of people but there were these big barriers. The first of which was that even if you were a developer, it was a hard code base to learn. It's a complex project.

Browsers are probably one of the most complicated pieces of software that anyone uses in volume, unlike a word processor or an image manipulation software tool. Web browsers have to take unknown and untested and unqualified content from random people, including idiots like me, who put up web pages who shouldn't, have to digest that, have to parse that information and make something workable out of it, and make something standard out of it so that, you know, we can continue to support sort of low barriers to entry and content creation. The browser has to take and make nice with a lot of garbage. So, it takes in a lot of garbage and makes something good out of it. And that is a very complex task to engineer, and to test.

So, that's sort of how I got involved in the project. And after sort of helping to launch the QA and testing community and to sort of grow that up, and then move more into sort of care and feeding of that and looking for other opportunities in the project, I got invited out to one of the anniversary parties here in California. I was in Austin, Texas at the time. And they had some little awards thing for volunteers in the community, and I got one of those for QA and organizing. And I came out. And that turned into—it was a wonderful developer event. It was a great time. I got to meet a lot of people at Netscape. Netscape was getting ready to release the first preview release of a browser we all knew wasn't ready. It was a big tense time.

And that sort of turned into a job interview. And one of the original staff members at Mozilla—Mozilla was sort of the—I guess about eight or ten people at Netscape who were tasked with focusing on the open source project, being responsible for the well-being of the code, the health of the community, and not sort of AOL-Netscape, sort of—then Netscape but later, AOL-Netscape's long-term goals for what they'd do with that technology, was really, you know, trying to be right with open source by setting up this virtual organization. So that was eight or ten coordinators.

And one of those had just left Netscape, the person who was responsible for quality assurance. Well, I had no background in this. I was in architecture and fine art preservation, historic conservation work, and like nothing to do with this. But they invited me out and said, "Do you want to—you know, you're clearly organizing people here—do you want to do that?" I said, "I've got a full-time job. I'm putting forty hours a week into this hobby, and you want to pay me to do that? And do you want to move me to beautiful sunny northern California to do that?" I'm like, I did a quick check with my wife just to make sure it's okay with her, but I was—yes was out of my mouth pretty quickly on that.

And I moved out. And then my—as soon as I got out here, my role started to shift because Netscape got into a mode where they needed to ship a product, or felt they needed to ship a product. And AOL was pressuring them to ship a product.

And so, they spent less and less time focused on the well-being of the code and the community and long-term viability and sustainability of the open source aspects, and more and more time on how do they ship this commercial product, and how do they hit these dates?

And so, things started changing. There were no longer a sort of a regular milestone testing releases being made. It was when Netscape had an alpha or a beta. I think some of the concerns around bugs that were important to a lot of people in the community were—became confused and disproportionate toward features that Netscape needed. And they stopped spending sort of resources on the project management aspects around shipping Mozilla and focused really on shipping Netscape.

And so, Brendan Eich of Mozilla staff decided that we needed to create a new group to do project management, project goals, roadmap planning and then driving release management. And so, he selected sort of top technical talent in each of the areas that were important to a browser, someone really strong from the layout engines, someone really strong from the application sort of front-end layer, someone really strong from QA, or—in that case, it was me as the organizer of that area, and built up a team of seven people to do that.

So, I moved over from doing QA and community organization on that to doing release management making sure we shipped something because Netscape wasn't interested in shipping Mozilla anymore. And so that was—at the same time, I still had all of those QA responsibilities, trying to make sure the releases were good enough to ship.

And so that's what I did all the way up until pretty much the end of Netscape's involvement with Mozilla and when they decided to sort of divest themselves of Mozilla. And Mitchell was hard at work trying to get the Foundation together. I'd been working with Blake Ross, and Dave Hyatt, and Joe Hewitt as sort of the project person, the testing person, and the blogger who was telling the world about a new project. It was called Phoenix at the time, that would become Firebird and then Firefox, a small stand-alone browser that we had thought would be more competitive with that landscape, no longer sort of doing what Netscape was doing, it was trying to hold onto the dwindling Communicator audience, or chase down the corporate space, but, actually, trying to take what we knew was a better set of technologies and some better features and get them in the hands of regular users.

And so just as the Netscape involvement was ending, Dave Hyatt and myself met with the rest of the Mozilla staff, and we had a big discussion about if this

Foundation happens, do they want to support a large suite of applications that requires a really large development team? And it was—the suite was connected in such a way that if some bug was introduced in the mail module that caused mail to crash and burn, the whole thing would go down, and that would mean that if we didn't have mail resources, the whole project could just die. It really required that we support each of its components.

And Firefox was a little bit different because it was a small project that really required that we maintain some basic infrastructure and could survive with a much smaller team, and a smaller scope for the application. It was also seeing some pretty interesting adoption, even in its early days. Phoenix 0.1 and 0.2 were seeing downloads that were comparable to what Mozilla 1.0 was seeing, which was surprising to a lot of people, I think, probably not to those of us who were doing it because we were—it felt to us like Mozilla had reached a saturation for the audience of power users who wanted that suite of tools, and that there was this other vast audience out there that was untapped, and rather than going after the two or three percent of Netscape users that were still around, go after the ninety-five, ninety-eight percent of IE users.

And so the Foundation agreed, and Mitchell and Brendan decided that that would be a good move, that the Foundation move over to Firefox. The Foundation started up. They hired immediately eight people or nine people, and I was one of those. So, that was Mitchell, and Brendan, an engineer for Firefox, an engineer for Thunderbird, which was the corollary mail project. It started up similar to Firefox. A couple of Netscape engineers who just said, "Hey, let's go do a lightweight mail client."

I guess they hired—I'm trying to think of who else was there. It was me, a core layout engineer, Brendan as the chief architect. They hired Johnny Stenback, who was in core technologies, DOM and some of the features that have to do with web page interaction. And so it was a real tight-knit small team that they figured they had the money to support for a few years. And then Firefox took off. Firefox sort of—we moved over to that, we put in all of our muscle behind that. And when we shipped Firefox—even before we shipped Firefox 1.0, we shipped 0.8, the first one named Firefox in February, six months after the Foundation started. And it had three million downloads or something, which was more than all of the Mozilla 1.0 downloads had had in the last year. And that was in a month.

Then we're like, "Okay, this is good." And by the end of that year, we shipped Firefox 1 and it did, you know, ten million downloads in the first month. And this continued on. We've seen a hundred and sixty million downloads in fifteen months. So, it's been a—you know, we've been able to increase that pace from the early release.

So along with that came a little higher than expected revenues that we get out of search relationships in the browser and that allowed us to ramp up. And we're in

that process right now where two years ago, we were that ten people that sort of seeded the Foundation, and I think we're up to forty-five or forty-eight today, and we continue to hire. We were just mentioning earlier that having our quarterly sort on site for all of the Corporation and Foundation employees and there were fifteen people who weren't here at the last quarterly on site. And I expect it to continue for a little while. So, it's really—it's pretty exciting. It feels, you know, really big.

I think that we had made some really good choices that put us in the right place to take advantage of a bunch of external circumstances. I think this is the case with most successful ventures, is, it's not all about the blood, sweat and tears of the people who make a product. I mean that is a key and critical and necessary component. We were there with Firebox, which was a useful alternative, but other things had happened.

The web had been getting worse and worse to use for people. Pop-ups had literally just taken over in a way that wasn't the case back in 1999 when Microsoft was still developing their browser. They pretty much discontinued development in late 1999 or early 2000, after IE 5. When they shipped IE 6, it was a minor update to IE 5, with not much in the way of new features. And then they disbanded the team, literally just shut the shop down in 2001. It was mostly done in 2000.

And the web continued to change. And their browser wasn't changing, and ours was. And we were keeping up with those annoyances as best we could. So, when the annoyances finally got so vicious that people weren't willing to tolerate them anymore, at the same time, Microsoft security woes had gotten so public and no longer was it the case that Microsoft could say, "Yeah, but you have to stumble on a bad guy's site." Now, the bad guys were putting their code into your bank's site. And when you went there, you know, you were being seriously compromised. And so, the security tech press, I guess it is, started to push back against the sort of Microsoft, you know, minimizing and trying to diminish the visibility and the real threats that were out there.

That was coming together just as Firefox was coming up to speed in terms of usability and clean interface, and as our core rendering technologies had been improving enough that we were finally getting a lot more of the edge cases that weren't rendering correctly. One of the big complaints of Mozilla was because we'd adhered to the standards pages, which didn't; some of them failed and failed in a big way. And so the first ninety-five percent of web compatibility is pretty easy—implement the standards that are in use.

The next five percent is reverse engineering all of the flawed content and how the other browser handles that. People were writing content to IE because that was the browser everyone used and that was what they tested with it. So, we had to go in and figure out what are all these quirks, and how do we build them into Firefox in a way that doesn't negatively impact the people who are writing standards-

compliant code. That's one of our sort of goals in the project, is to support free and open standards. So, you don't want to like, at the detriment of that, support some of these proprietary technologies.

But we had gotten enough and solved enough of the hard problems that I think we had gotten another few percentage of web compatibility that got us to a level that was acceptable. And any numbers on this are pretty impossible to come up with. But I feel like, you know, if that ninety-five percent was where we were in 2000, you know, we've got another few percent by 2004. And that was slow going and hard work but enough of the the web worked.

Microsoft security woes were getting out of hand. The pop-ups and Adware and Spyware was getting just ridiculous. And we had a pretty useful product and that all sort of came together with our 1.0 release. And that was just, you know, wonderful timing. I mean we sort of felt it coming, didn't know if we were early or late. But it did all come together in a two or three-month period of time. All of that just was plainly obvious to the world. And that drove this huge spike.

I think the other component that came in there was, in the same way that we knew that if Mozilla was going to be successful, we had to—that those of us who early came on, who weren't developers, we had to find a way to involve non-developers. Blake Ross and myself spent some time thinking in the summer before the release, how do we—how do we get this in front of real people, right? I mean we know how to get a “Slashdot” story and to get, you know, a few more geeks to take a look at it. But they'd been watching Mozilla for six years. How do we reach out past those first two or three million people?

We realized we were going to have to market a product. No longer was, you know, Mozilla sort of a technology underpinnings of other commercial applications, like Netscape or AOL or whatever. But now we have a product. So now we have to figure out how to get people to know about it, to know what a browser in the first place, and not just that “this is the Internet” but that a browser makes a difference in what Internet you experience, to get awareness of, you know, the brand, and to get people to understand what it meant to download and to install a new piece of software.

And so in sort of brainstorming how we were going to leverage open source techniques to do that, we came up with the idea for Spread Firefox, which would be a tool suite for people to use to spread the word, to convince others, to advertise, to promote out in the real world, to go door to door, all the tools that we had seen people using in political organizing the year before—the early online political organizing you saw at the Howard Dean campaign, and later adopted by most of the rest of the Democratic candidates and to a much lesser degree, the Republicans.

And so we took sort of what they were doing in grassroots organizing, put together a set of tools in a similar way we had done with QA before, and engineering before that, so that people could collaborate online. And so we had that in place with the launch of Firefox 1.0. And I think what they did was that community helped us amass a critical mass of early users. They really got the wick burning on the stick of dynamite in the first few months. And in those first few months, I think we saw a large percentage, maybe as much a half of our downloads coming as a result of the projects being orchestrated at Spread Firefox.

If you look back now with a hundred and sixty million downloads, I don't think it's nearly that high. I think that the number of projects that directly can be attributed—you know, the number of downloads that can be attributed directly to Spread Firefox projects is probably much closer to five or ten percent of our downloads.

But in those early days, when you're talking, you know, ten million downloads, if thirty, forty or fifty percent comes from a project, that's a, you know, a lot of gain for what turned out to be not a lot of effort in terms of building this community. And so I think that really got the ball rolling, and that got us ten million downloads in the first month, which got us press, which got us coverage that led to more downloads, and really allowed the snowball effect to get going.

We also had a couple of sort of exciting things. We had a campaign. The first big visible campaign was the New York Times campaign where one of our volunteers with Spread Firefox said, "We should take out an ad in the Wall Street Journal, right? You want to go after, you know, a new market? Take out an ad in the Wall Street Journal."

He called them up and they said—I don't remember what the price they quoted him, many tens of thousands of dollars to take out a full-page ad in the Wall Street Journal. And he said, "We can do this. We'll raise the money from our community, right? I mean Howard Dean can raise \$30 million in the summer, we can raise a few tens of thousands of dollars." And we all sort of said, "I don't know about that, right? Let's put some banners up on the websites, and let's put some buttons up and get bloggers to talk about us." And he said, "No, really, we ought to try this."

And so we got on the phone with him, and we started figuring out what are the logistics here, and what do we think we can do, and put together a project around it. In the discussions, we decided the Wall Street Journal really wasn't our audience, and didn't quite—didn't seem like the best first place for our community to start. So, we had some debate about what newspapers to try to get into, USA Today, New York Times, LA Times, and finally settled on the New York Times because it's a paper that's internationally recognized. And if you say we got an ad in the New York Times, that's really sort of about as big as you can get for papers.

So, we set aside a month of time on the site, and a framework to try to raise these many tens of thousands of dollars in a month's time. And I think we had a target for the ad, and ten days in, we had double the target, into the month. And we went, "Okay, this works."

And it wasn't just the raising of the money, but we also had a group of graphic designers come together and started collaborating on what would this be. We had a group of marketing professionals, including the guy who launched the project come in and say, "How are we going to position this? Is this an ad for Firefox?" And settled on the idea that we didn't want to have an ad for Firefox. What we wanted to do is celebrate the launch of Firefox 1.0 and the community that got us here. This is going to be about the people who got us there.

And that turned out to be—I mean a key feature of that, which was we're going to incorporate into this a celebration of people who are contributing to it. And one of the designers came up with the idea, "Well, instead of having, you know, just a pure graphic on there, let's build the Firefox logo out of the names of all of the people who donated so that when you donate 5 bucks to, you know, to this, you get your name in the New York Times." And that ties back to that, you know, we're not just about, you know, collecting money to go throw up a traditional ad, but this won't be, you know, why you ought to get Firefox. This is going to be, "Look what this community did."

And it proved to be quite successful as a wonderful, visual ad and as a celebration point for the community.

I don't believe that the number of people who saw that and downloaded Firefox amounted to very much. But the people who saw the forty or fifty other articles covering the event of, you know, 10,000 Firefox fans get together and raise, you know, a hundred thousand plus dollars to take out two full pages in the New York Times, that got a whole bunch of coverage, starting with the tech press but moving out. I mean that got us, you know, magazines and other newspapers. And I think that kind of reach and the fact that, you know, we were standing up as the first—or sort of prototype in software doing open source marketing. And we'd seen, like I said before, the political camps employing some of these same tactics. But it was really a first for user-generated and user-supported advertising coming out of a software community.

And the coverage of that was good enough, I think, to really help, you know, drive some downloads, and to drive visibility of the Spread Firefox effort. So, we went from, you know, 10,000 registered users in the first month when that project kicked off to 50,000 registered users six weeks later or something like that. It really just shot up around that because people said, "Hey, I want to be a part of something like that."

So that's been what I've been working on for the last couple of years, primarily is the community organizing around Spread Firefox and the marketing work.

And now I'm interested in, and one of the reasons I think that I wanted to speak with you guys, I'm interested in—you know, I've been around for six years. We're in an organization here where there's probably only, I don't know, five or six people in our organization, in this building at least, who have been around as long as I have, and have memories of each of the stages this project went through. And we're all starting to publish a little bit—some in our blogs, some in an interview that we give with some magazine who wants to talk about the phenomenon of Firefox, or something like that, we're all sort of reading each other's perspectives on, you know, what were the inflection points? When did we make that pivot? Who was responsible for that decision? There's enough sort of difference in viewpoint on--and I don't think anyone is—there're not a lot of factual disputes. But there are a lot of sort of, you know, “I don't quite remember it that way.”

And so one of the things that I wanted to do was find a mechanism that allowed us to start to record a history in a way that wouldn't lead to fights or personal fallings out and that would respect that we come at this from a different perspective. And so when I started looking at other efforts, like the Wikipedia entries where there's some good timeline information, and a few good quotes, and things like that, I don't think it's very interesting. It's certainly not fun to read. And it's far from comprehensive.

And so trying to fill something out like that or encourage those who have good memories to go in and just plug more information into that statement, like it wouldn't be very productive. And that's when Raphael and I went sort of looking for, “Has anyone else done this,” when we found the Folklore project. And that seemed like a good model because it does respect the viewpoint of the participants and you know, especially when we're talking about six years ago, and you know, this meeting or that, I think that there's enough distance that things that happened last week, I struggle to remember really what happened at that meeting. So, I was really excited when Raphael mentioned you guys because I thought, “This is some people who have some expertise in this.” And I don't have to invent some way to make it, I think, a fun and valuable project for our community. And that's one of the things that I like doing that I'm—you know.

I'm not a developer. I came and did QA for a few years. But I'm not a quality engineer. And now I'm doing marketing. But I'm not a marketer/advertising person. But I like these projects, and I like building the communities around these projects. And I think that it would be great if we could find a way to employ the same techniques that we've employed or the same, at least, models that we've done for open source development, and then open source quality assurance and testing, and open source marketing and use that to build a good history of this project, because I think already people are starting to try to figure out how we got

here. And they're going back and looking and saying, "Hey, I'd like to replicate some of that magic, you know. How did it happen? And why did it happen?"

And there aren't any sort of—there aren't any good articles really out there that cover why we're successful. If you read business magazines that are covering us now, they'll come in and talk about management style after, you know, an hour interview with a few of us, or something.

But none of that really captures—you know, there are so many things went into getting us where we are, and I think that, you know, there's not a big vision anywhere that's been documented, like you know, what are the sort of—you know, the five or six big things that make Firefox and Mozilla different from other projects, or, you know, or how did the decision to move from the old Mozilla application suite to Firefox, how did that go down? And you know, and what did that mean for the community? And, actually, that was a tumultuous time, and there were big fractures in the community that probably a lot of people don't know or realize, that Firefox actually was a change that was too radical for a lot of people in the community, or they ventured off on a path that wasn't interesting to people who got involved because they wanted to build the most powerful browsing utility they could browse, you know, they could build. And they came in to scratch their itch, not to get mainstream adoption or something like that.

So, there was—I think there've been a lot of large events that are just unknown to the world, and that if they were, would give a much better perspective on you know, where we are today, and I think take, you know, in some ways take a little bit of a shine off of, you know, the—I think we're sort of being a celebrated open source project right now, and the truth is there's a lot of, you know, there's a lot of, you know, watching sausage being made and all of that metaphor, right, if you actually came in and saw what was going on, you know, where everyone here is really dedicated and works really hard. But it hasn't just been this, you know, open source is good, and if you do it right, it works.

What's right? We're inventing nine out of ten of the things that we're doing every day. There isn't a model. So, there's you know, going down paths that don't pay off, and recovering from you know, mistakes and you know, architecture from three years ago, on a piece of the product that now is, just, you know, costing us in terms of maintenance way more you know, man hours than it would if we would've made a different decision three years ago.

That's one of the places where I think gathering a wide range of perspectives will help to fill a lot of that in, and I think, you know, I look forward to a view of Firefox that's a little more in depth, and a little less, you know, management secrets of you know, open source or something like that.

Tom Scheinfeldt: Following up on a couple of those points, for instance, what were some of the tensions when Firefox came along between the kind of slimmed down Firefox product and the beefier Mozilla code base?

Asa Doltzler: Well, it's my view that there were sort of two major tensions. One was, and the most important one, in my view, was it was a change in the development model. Open source has traditionally been, at least in Mozilla, very inviting. And we want people to come in and to work on something they enjoy working on. And we want to ship it to, you know, a million people.

So, if you decide that it would make your life easier if the toolbar had a button that let you manage your cookies. Now, we're going to put a button that let's you manage your cookies, and ship that to millions of people. Never mind if that button confuses the heck out of you know, 100 million other people. If you wanted to work on that, and if your code was of good quality, and didn't introduce new bugs, your feature would probably make it into the product. And there wasn't a lot of sort of, product decision. There was a lot of technical rigor, and that has always been strong.

But it led to this rather large and cumbersome piece of software that had a whole bunch of features that was there in the way for most users. And so what Firefox did was Firefox started off with three people, three engineers, basically, Blake Ross, and Dave Hyatt, and then Ben Goodger, and it grew to five or six engineers by the time it was up into the Phoenix 0.1 stage. But it was really three engineers, and a fellow named Ian Hickson, who's at Google now, who was—he's on the—all of the web standard groups for CSS, and working groups for a bunch of these merging web standards. And I think he's got some great ideas about you know, measuring quality and usability, and about where the web is going. And me and my expertise in shipping releases and figuring out how to drive bug lists down, and get a product out the door, and to get that in front people, using you know, whatever announcements we could use, and whatever little bits of fame we had at Slashdot or elsewhere.

But it was a very small team of people who said, you know, "Open source doesn't mean that everyone gets to participate in every decision. There are some decisions that require an individual or a small group of people acting as, you know, a benevolent dictator in this case." And these are people who did earn their way up through the meritocracy. I mean it wasn't just someone came in and said, "I'm going to be king here," but you know, that opening up the product level decisions, feature decisions, what you expose in user interface to the whole community is probably a mistake.

And so they said, "What we're going to do is we're going to—you know, our software is of course open source. We're using, you know, the same licenses that Mozilla uses. Anyone can take it and go do their own thing with it. But we're going to have a small team, and we're not going to open up to the world. It's

going to just be these three people writing this application, and building on this wonderful sort of platform that Mozilla had been developing over the years.”

And so it was—I think seen as very elitist, and not in the traditions of Mozilla where everyone was welcomed to participate in every decision. That’s an exaggeration. Everyone wasn’t always welcome to participate. But it was more than one would probably want if you’re producing you know, a product for a particular audience, and especially if that audience isn’t open source developers, right? And they had built a browser that was really good for open source developers, but not so great for my mom.

So, that was sort of the, I think, the big change, more than any of the actual feature changes. There was a change in the culture in that group of people where we said, “This is going to be a small team. We’re going to move quickly. We’re going to make decisions, and we’re not going to, you know, debate things endlessly. And we’ll put them out there for users to try. And if the users reject them, then we’ll know we made a mistake. But we’re not going to, you know, committee things for a month, or you know, open up a discussion on MozillaZine for three weeks about how the feature out to behave. We’re just going to move and trust our instincts on this.”

It turned out that the developers had really good instincts, and I think picked a really good feature set that was right for a lot of users.

But there were a lot of people in the core development world who saw their features being removed from this product, people who were developers of back-end code, the guy who writes, you know, the small database, that handles all of the cookies, right, all of these little pieces of content that websites drop off in your browser so they can recognize you when you come back, and things like that.

The guy that was writing that database was also writing a toolbar to manage cookies that would be up there in the front of the browser, and it’s just sort of wrong. This is somebody who has no experience or real understanding of usability and user interface design. It was just, you know, a sort of back-end engineer working on the front-end, and it would be just as wrong as if these front-end engineers went in and said, “You database ought to be organized like this.” You know, I mean, these are people who were sort of working out of their area of expertise, but because it all fell under that module and the code base used this cookie handling, it was sort of allowed.

As so as those were all sort of pared out of the browser, I think there were a lot of people who felt like their work you know, wasn’t appreciated, and that this was no longer a browser that they would use because you know, the itch they came on to scratch and had completed and successfully scratched was itching again.

So, there was animosity in terms of someone would come in and say, you know, “I want to help out. Here’s a new feature. Can you add this?” And the developers would say, “No, right now, we’re not accepting feature contributions. We’re focused on the vision we have.” So, it’s sort of those two things. The feature set changing radically, being trimmed down to the stuff that I think was most appropriate for you know, the largest number of users but now are cluttering up and getting in their way, competitive with what was out there in the landscape, which was Internet Explorer, and making it comfortable for Internet Explorer users to transition over.

So, features and the shift in exclusivity, or toward exclusivity, and toward this small group of people working you know, in less than sort of public and discussed environment, and all of that. That alienated a lot of people. And there were some who said, “You know, if that’s the direction things go, then I don’t want to be a part of this project.” And some of them left. Others you know, grudgingly accepted it. Others waited until Firefox was sort of adopted by Mozilla, and once it was adopted as a formal project, you know, the rules did change some. And it was opened up more. And people, you know, by necessity, once it became an official project, then some of the old rules that were there in the early Mozilla days were sort of put back in place in terms of roles and responsibilities.

But it was a tumultuous time, and I think that a few things really helped us get through it. The first was we decided early on that if we’re going to be saying no to all of these features, if we’re not—in order to not lose the current user base, one thing we can do is we can provide a mechanism for them to get those features back. And so we developed a system that’s today’s extension model, which allows us to—allows pretty much anyone to go build a small feature and bolt it on—a plug-in, but most plug-ins operate in the content of the browser, Flash or applets or things like—these are sort of plug-ins for the features of the browser.

And so we actually invested quite a bit in the infrastructure to support plugging these extensions in, and that includes making sure of version compatibility stuff. It really made the browser a platform for other features, and required a lot of work to get right. And it’s just now I think, really starting to come into its prime.

But that allowed an engineer to—whose feature had been sort of pulled out of Firefox, or a new one who comes along who wants to implement one, and says, “Here. Here’s my code. Take it into Firefox.” It gave us the ability to say, “No, we’re not going to take your feature into Firefox. You can build it as an extension, and distribute it through Mozilla’s extension distribution website, and lots of people will get it. But it’s not going to go into the core Firefox download because it’s only applicable to you know, two percent of the world, not ninety,” or whatever those sort of—ratios were, and so I think the extension model helped out a lot.

And the next was it started to become successful. And that had a couple of big impacts. One is it vindicated the feature decisions. Firefox is being adopted by a lot more people than Mozilla is. So, it must have a more appealing feature set. And that's a reasonable argument to make. And that allowed us to say, "Hey, in this case, it turned out more people did want the Firefox browser than wanted the old Mozilla suite." And it was a more universal application.

The second thing is that as it got famous, it became a project that people wanted to be associated with, whereas before, some people might have felt like, "It no longer feels like me. My feature isn't there," or "I'm not allowed to work in that area," or, you know, "I really like my mail and my browser bundled together." Now, Firefox is something that a lot of people are looking to as an open source success story. And so people who are interested in participating in open source are interested in being a part of that. And as the fame around the product and the project grows, I think the number of people who want to be a part of it is growing. And the visibility attracts more contributors. And so we have a whole bunch of new contributors that are in Firefox now that came along as a result of Firefox existing that probably never would have showed up had we continued to go down the path with the application suite.

So that was an interesting time, and it was tricky. And there were, you know, there were I think some hard feelings here and there. Interestingly enough, it didn't just impact the people whose features left, right? I mean we had Blake Ross and Dave Hyatt and Joe Hewitt and Bryan Ryner and the five or six developers who were involved in Firefox up through the first public releases of Phoenix, went off and did other things. And Ben stayed and Blake stayed part-time but everyone else, I mean I don't know whether it was a result of sort of hurt feelings or just other opportunities. But they created something really good. And then Dave went off to Apple, and he's working on Safari there, helped launch their Safari browser. And, as I see it, there're some interesting similarities there. Dave put some great features into the early Firefox releases and went off to Apple and that browser seemed to have a lot of the same features in it. Ian went off to work at Opera in their QA department. I think that, you know, Blake went back to school and then got involved in some other projects.

And so it was a—it was a divisive time, and I think, that—kind of a tricky time for us because as that was happening, it was also the birth of the Foundation, this new organization, this new legal entity and this, you know, who are the people who are now representing Mozilla and are you know, are able to do it full-time by virtue of getting paid by the Foundation. And all of that was happening at kind of the same time. So there was a lot of struggling to try to figure out how is the Foundation going to organize itself? How are we going to work with our existing community, and grow a new community? And, at the same time, we're also going to be you know, moving our investments over to Firefox and away from the old suite.

And then we had, you know, at Thunderbird, there were a lot of people said, you know, “I can’t go without a mail client. And I’m not going to move over to Outlook,” and Thunderbird was a good partner app. And so we had a Thunderbird developer. But it was always sort of a little bit less focused than Firefox, and you know. So I think there was, you know, concerns about our—Do you no longer care about the mail piece of the Internet? And e-mail is, I mean, still to this day is of more widespread use than web browser, it is the killer Internet application. And when you—when you start talking about the Internet, and not just the web, not just http, there is a lot other--else going.

I mean publishing—we used to have a publishing module with Composer, and we didn’t have the resources to sustain that. A project called “End View,” which is based on the old Composer code base started up and it’s a company started by one of the old Composer engineers. So it does exist out there. But it’s deviated from what it was in the suite. Are there other spaces that if you’re going to be in the Internet—you know, if you’re going to have as your mission promoting innovation, and choice on the Internet, well, the Internet isn’t just the web. Should we be thinking about instant messaging? Should we be thinking about VoIP? Should we be thinking about all of these other areas?

And so, these discussions really started when the Foundation started, and trying to figure out what our mission was, and are ongoing today. But I think that, you know, that we probably, in hindsight—not probably—we did make the right decision to move over from a suite that would have been really difficult to maintain, to a lean and fast browser that turned out to be something that the world was looking for, a non-trivial percentage of the web browsing world was looking for.

I think we made the right decisions in investing in the engineering that made that possible, investing in Spread Firefox, and the marketing efforts that made that possible, and then investing in the business today with the corporation ramping up in terms of engineering resources, business development, and relationships with other organizations, reaching out to other open source communities.

We’re spending, you know, we’re spending real money on marketing now. It’s not just you know, me and Blake, and Spread Firefox volunteers, but we’re actually looking to, you know, to run, you know, more mainstream ad campaigns and things like that. And I think that we’re, you know—we’re in a—we’re always in the state of transition. We have been since 1998, where something is changing every year, something big. Like I mentioned earlier, the first big change was a move over to an entirely new code base after a year of open source development on the old Netscape code base. And not long after that, AOL acquired Netscape. And Jamie Zawinski, who was sort of, you know, the face, the personality of the Mozilla project, went off to do his own thing in a way that wasn’t necessarily really supportive of the organization he left behind. I think that, you know, then we had the big transition to Firefox and the move away from being primarily

supported by AOL-Netscape to a Foundation and then into the successes of Firefox and the birth of the corporation, the changes and structure that entails, new business relationships, new hires.

And it really feels like this organization is success—I believe that we’re successful today because we’ve been willing to make all of these changes along the way, and that I think that the project probably would have fizzled had they not made the decision to move over to that new rendering engine. And if it hadn’t fizzled, it would have run out of steam and would have been un-maintainable a year later, and that code base just wouldn’t have survived.

And so as I look back at all of these big decisions that were really hard, I’m really glad we made them, and it makes sort of wondering about the next one. Now, I expect that we’re going to have to continue to do that, and that’s—in some ways, that’s a piece of, you know, of the innovation that’s required in this space, and it moves really quickly.

I don’t have too many years left in this. It moves too fast. I feel like I’ve lived a pretty significant career here already, and it’s only been six years. And I think that the next couple of years, I’m going to have to—my wife and I are starting to look at the alpaca farm in coastal Oregon, and figuring out how we can settle down in something that moves at a little more normal pace than this.

Tom Scheinfeldt: What are some of the things across these transitions? What are some of the things that have stayed the same?

Asa Dotzler: Well, we have had a core set of principles, I think, that have always been there, starting with the fact that we’re open source. And that is not—and I cannot foresee it ever changing. Our source code—the work we do to build the technologies and the products we build is freely available under the Mozilla Public License, the GNU Public License, and the lesser GPL. And that means that anyone at anytime can come grab everything we’ve done, make some little tweak to it, call it their own, and push it out the door. And that means that we will, by necessity, have to be innovating. And if we don’t, then someone else will come along and leverage the work that’s gone before, and build a better product, build a better set of technologies.

So, I think that we are open source keeps us on our toes, that we know that we have to lead. So that’s something that has been there from the very beginning. And there’ve been a million discussions about where we’re going, and what we’re doing. And it seems to me that’s just always understood. If there’s ever something we do and we write a piece of code, and it is an open source, it isn’t because we didn’t want it to be, it’s because of expediency, and we’ve got various tools and things to help us get through our daily lives. But what we do, all of it is with the goal of making that freely available under open source licenses.

The second thing that I think is that we've always had a commitment to sound technical decision-making. That is we—the decisions that go into building the technologies and the applications that we build will always reflect sound technical decision-making. There will not be a time when, you know, when someone in marketing says, “Now, Thursday's the day we have to do this, and that's the window. And if it's, you know, if you can't fix that bug, sorry, it's going out Thursday.” The technical—the technical teams here will trump those kinds of decisions.

And we've had a lot—and in the end, I think, a lot of them do come down to some, you know, to some resolution that's comfortable for everyone. But the technical dominates it.

And another piece that may not have been through all of the transitions but has been for the last two or three years is, that I think we're really starting to focus on users. We've had a long-term focus on the open protocols and the standards of the web, and the open technology that we need to preserve so that there is an opportunity for choice and further innovation on the Internet. So supporting the standards and implementing new, you know, new protocols as they come out is really important.

But in the last three or—I'd say about three years since the birth of the Firefox project, we've said that we're going to respect the users. And that's been really key to the last three years when—and maybe even before Firefox—in Mozilla we developed pop-up blocking technology before anyone else had it, before the ISPs were all shipping it. As browser experts, we said, “Hey, a website makes a pop-up by calling this bit of code.” If he calls this bit of code and it wasn't as a result of a user clicking on a link, then let's just say no, right? We can say no, we write the browser that makes the pop-up window.

And we put an implementation of pop-up blocking into Mozilla. And Netscape shipped their next release with that turned off because Netscape and AOL generated a non-trivial amount of it at the time, generated a non-trivial amount of revenue from pop-ups in their web properties.

And the press, you know, it was kind of amusing—looked at that and said, “Here's Mozilla, and here's the Netscape version that shipped from it. Mozilla blocks pop-ups, and Netscape doesn't. And there's a pretty clear reason why they don't. You know. Who would make a decision like that?”

Well, it turns out that the business people made that decision at Netscape. And in Mozilla, we looked at that and said, you know, “What's going—what's going to—what are the factors that we're going to consider when we make this decision? And do we care about or should we care about the owner of the website who's making money off of that advertising. Should we care about the business model of the pop-up advertisers, the double clicks of the world? Should we care

about, you know, should we care about adding a myriad of new features that let people do whatever they want, right? I mean, you know, is it good enough that, yes, pop-up blocking can be there. It cannot be there. Netscape can turn it on; can turn it off. Joe Shmoe can turn it on; can turn it off.”

So, we looked at it, and we finally said, “Really, this is about the users. We’re trying to make a browser for people to use, to make their experience on the web you know, easier, safer and more functional.” And so, you know, that was sort of the beginning of, “We’re going to side with the user.”

And that’s motivated all of the decisions that we’ve made about the product since then. And I think that’s held true, and it will continue to hold true. And, you know, as we go forward, I think that we’re in a position now where we have a large enough user base, and we’re interesting enough to other organizations who now want to be a part of Firefox, who want to get their feature into Firefox, want to be the default search engine in Firefox, want to have, you know, some—you know, Netscape did these really wild things. They had a relationship with Hewlett Packard printers, and so, you had a menu where your—you know, “print” and “print preview.” And one of those items was “purchase print cartridges,” you know, literally advertising right there in the menu.

This is pretty funny, but actually when we first started Phoenix, Ian and me and Blake and Hyatt were sitting around on IRC chatting, and said, you know, that we’re landing the first code changes. We’d create a new directory and start building this browser, and it built on all of the foundations and the work of the Mozilla project but just this new UI layer, we need to tell people why we’re doing this.

And so we started to put together a little manifesto. And so Ian said, you know, “We’re going to have ten points in this manifesto.” And these were going to be the guiding principles of what was then called M/B for the directory of the Mozilla slash browser where we created it and then became Phoenix. And we checked in that file along with the source code so anyone who was interested from a developer standpoint could go find this manifesto. It was a little text file. And I think number six on the list was “the browser will never have a whore bar.” And what that was in reference to was that Netscape had sold all of the bookmarks on the “personal toolbar,” this is your link bookmark toolbar that’s Netscape’s name for it, the personal toolbar, the thing that is yours but it was filled up with paid for sponsored links. And some of them weren’t just links. They were other services. If you clicked on it, it would go out and fetch an executable for this, you know, VoIP product, and install on your system, and launch it, you know.

And it was like—it was so clear that they were going to monetize every pixel on the screen that they could—that that was a you know, a fundamental principle in the development of Firefox, was we’re not going to whore out to Corel, and we’re not going to—the interface is sacrosanct.

Now, if someone wants to do a distribution of Firefox and include their own links or whatever, that's different. But what we're going to do is we're going to build a browser that treats the user right first. And all of the decisions we make about features and about any relationships with other companies, any you know, financial agreements or sponsorships are always going to be subservient first to, is this good for the user; second, is this technically a strong decision and viable over the long term? Is this good for—is this good for all of the people participating in development? Is it going to sit well with all of them? And then is there some opportunity, potentially, of revenue or something like that? But that's really down at the end of the list instead of the way it was, Netscape-AOL where the first decision was. Is there revenue opportunity we can attach to this piece of the browser?

Tom Scheinfeldt: What are some of the maybe persistent tensions or conflicts that have kind of stayed with the project over your time, and—or were there kind of persistent tensions that have been resolved? What are kind of some of the problems that keep cropping up or cropped up for a long time?

Asa Doltzler: I think our problems are really not—we don't have a lot of those kinds of problems. The problems that we face, and that we are sort of rehashing, never really solidifying is, are things like, how do we stay vibrant? How do we attract that next group of contributors who are going to be the Blake Ross, or the Dave Hyatt, who help us over that next hump, and help us innovate into that next space? How do we reward those talents? What are fair ways for us to credit people for this? Is it you know, good enough to put their name in a text file and their credits that ship with the browser and that's available in the "help" dialogue?

Should we be featuring them in our website and saying thanks to them? Should we be hiring them, right? I think that the equity decisions in terms of contribution has been—it has been a challenge, and resurfaces pretty consistently.

There're issues with operating in a fishbowl, as I call it, right? We work in a very transparent environment where all of the decisions are visible, mostly in Bugzilla and the newsgroup. But you can go in and see the discussion about the bug or the feature and track the interaction between developers. And anyone can—you don't even have to have an account. You can literally just come and read what we were doing. And I think that, you know, that means that anyone with an opinion can come into our system and express that opinion. So trying to figure out how to invite people to participate in a project, but also shield ourselves from some of the noise that's not very productive, and to do that in a way that doesn't just mean getting opaque, right?

And that's you know, one of these things that comes up pretty regularly is, how do we—how do we move forward and how do we, you know, how do we accomplish our goals and still do it in a way that is highly transparent?

There have been some—there have been some you know, some stresses around our platform story, right? We build—we build a product that runs on MacIntosh, Windows and Linux from our organization. We have teams that have got Firefox and other Mozilla products running on fifteen different other platforms. So, there're tensions between, you know, most of the people in this organization spend most of their time developing with an eye toward Windows or Linux or—I mean sort of Windows, then Linux, then MacIntosh. And so the MacIntosh people are, you know, “MacIntosh doesn't get enough love”.

And while we're cross platform, there are these things that are platform specific. There're things from the look and feel to integration with platform system dialogues in Windows. And, you know, do you use the native platform's print function? Or do you reinvent that on your own? How much, you know, there are oftentimes a feature that comes up that literally cannot be implemented on one platform, or it's taking advantage of something that's only available on MacIntosh, or only available on Windows. Do you not do it because you can't keep parity on Linux?

So there's some interesting platform divisions, and I think there're some tensions between the people who care about those, and whether or not their platform is getting the attention it deserves.

There are some tensions between, you know, the developers on sort of the front-end of the project, and the back-end of the project. And where are those lines? And ought there be good distinctions? Our code is actually not—and I'm not a software developer, and haven't been around the industry enough to say this with a lot of authority, but as I understand it, the lines between our front and our back-end are pretty blurred. We actually have sort of about four layers there.

And the very top layer is, obviously, front-end only, right? These user-facing, you know, the visual layout and the feedback one gets when moving around with the mouse, or the keyboard in the browser, but what's underneath that is a layer of widgetry that's reused in different places in the browser. And it's also used over in the Thunderbird e-mail client. And so if there's a menu that drops down as a basic menu function, we've got a shared code there where possible.

And then underneath that, there is a layer that's sort of our platform story, which is that you can build lots of different applications. Like Firefox is an application built on the Mozilla platform. Thunderbird is an application built on the Mozilla platform, and it utilizes our Gecko rendering engine, and utilizes our tool kit and our Widget layer where we make all those menus, and buttons, and toolbars.

And then there's the rendering area, which houses applications of its own on the web, and exposes those to the user, some of which we have control over. Is that the domain of the people who build the rendering technologies, or the people who

build the buttons on the toolbar, how we're going to expose a button on the web page? And so it's—I think that it's not always easy to figure out where those boundaries where, and where overlapping roles and responsibilities in the organization are.

But I don't think that we're terribly tense. I think that, for the most part, people have been able to move into the area of the project they're most interested in working on, develop an expertise in that area, and through our meritocracy here, if they've been around, they've demonstrated you know, good work in those areas, and they become authorities, and become responsible for that area, and get the decision-making authority that comes along with sort of a tenure on the project, then that really works quite well. It's it a pretty proven model in open source development, this idea that you have this ownership hierarchy that's really based on merit.

And so we're certainly not the first product to do that. I think we're doing it on a scale, and with a user base that's larger than all of the other open source projects, which makes things a little more heightened in terms of the sensitivities where some of those boundaries do—or some of those overlaps do exist.

But maybe some new tensions that are coming up are, you know, have to do with—this is a project that is now generating revenue. And what kind of business arrangements do you make? And where does that revenue go? Where in the project is that invested? Is that invested in hiring more people to work on it? Well, maybe money could be spent funding other projects where you're not directly hiring someone. You're saying, “Hey, we would like this feature,” you know, putting up a bounty for it or a scholarship to have someone do this.

And so I think there are going to be some new and interesting tensions to deal with going forward. But, really going back, you know, it's been an engineering dominated organization. Decision-making has been based on technical merit and on what's right for the user, and pretty much everyone, I think, is in agreement on most of that stuff.

Olivia Ryan: You mentioned that you're interviewing people here. Are they—most of the people you're interviewing, or all the people you're interviewing, are they—or have they been volunteers, or do you know them primarily through other Mozilla projects that they've worked on, or--?

Asa Doltzler: In the very recent months, there are a number of people who weren't—who I didn't recognize from our communities or—.

Olivia Ryan: So, they sent resumes in or—?

Asa Doltzler: Yeah, or someone here worked with them at some other company, or they worked on some other open source project. We've hired a number of people who have

open source experience who worked, you know, on one of the Linux projects, or who worked on one of these. But you know, over the last four or five years, overwhelmingly, we've hired out of the community, and it makes absolute sense. I mean this is sort of like having someone come and intern with you for a few years, and you know exactly what their qualifications and capabilities are. Anyone can go look at their contribution, evaluate it and—not just that but because we're a technical organization, we have such a rigor when it comes to technical review and all of that. Every change that's made in Mozilla is well documented with technical reviews, multiple layers of technical reviews.

So then when you've got a couple of years of volunteering on the project, there's this great mass and wealth of information about the detail, very specific quality measures of your work along the way, and how well you, you know, how well you take those criticisms or suggestions or requirements and turn them into, you know, the next rev of your patch. And so it's what you can never get hiring from some other employer. You're not going to go to Apple and say, "I want you to give me all of the technical reviews you ever did on this guy, all of the performance reviews you ever did on this guy. I want to see those before I hire him." That's just not available in the traditional hiring world.

So, it makes a lot of sense to hire in the community. You know they're already up to speed because you're probably hiring them to work on areas that they enjoyed working on this as a hobbyist. I mean that's what I was doing. And I think I turned out to be a good hire for the organization. But you know that you're getting someone who's passionate, who is, you know, in many cases, willing to do it for free. And it makes a lot of sense.

We're growing, I think, at a pretty quick pace right now, and it turns out that, for example, if you look at our core rendering engine developers, well, we hired them all, right? I mean that's a very complex piece of technology. And the people who came in and wanted to work on that and were intrigued, we hired them. And so now if we need another resource in that area, and I'm not the engineering manager hiring, so I can't say whether they're hiring specifically for any one area. But if there wasn't another person, then you go looking for people who have the skill set that would, you know, that would fit working in that area, and so other great places, or other open source software projects. Other browser organizations, people who have worked for other client software organizations, and not just browsers, but if you've been involved with developing any client software.

We have lots of friends in the industry, going back to the days of Netscape where—you know, Netscape was one of those first Silicon Valley successes. And they had, you know, I don't know how many employees they had but they had a lot of them. And those people are all scattered around. And there's a history and a lot of sort of relationships that persisted. So, we know people in every organization out there, and that means that you know, if someone is, you know, if someone has been at Yahoo! and they're getting ready to leave, they can talk

around to other people in Yahoo!. And it turns out there's going to be four or five people that say, "You ever check on Mozilla? I'll bet they'd like you."

So I think that it's helpful that we're well connected, and that's a result of, I think strong technical people sort of keep in touch. They put in a lot of—lot of hours together on projects and stuff. And I think some pretty strong bonds form there. So, I think we're pretty—hiring here is—it's important that we do it right because we're still small enough that each of those decisions really has a big impact on the long-term health of the organization. But we've been pretty fortunate in that the pool has been good that people want to work on Firefox, many of them enough to come in and do it as volunteers.

But there are lots of people with great expertise and experience in this industry who now see Firefox as a great career opportunity, and not just a fun project to work on. And that's really wonderful that we're in a position now that we can—first, that we can sort of finally reward a lot of the people who put a lot of years, hours, months in volunteering, and that we're an attractive place to work because we have such a high profile product in Firefox, and that we're moving the industry forward. Right now, there are a few companies that are moving the Internet forward. And I think we're seen as more and more as one of those companies that is driving change. And in the last couple of years, things have started to pick up again. I think Firefox is one of those places where it's really picking up. So, those all advantage the hiring manager. I'm not actually hiring for any position, and I don't manage any position. But I think having been here for a while I want to be involved in it.

Back in the day when it was just, you know, six or eight of use, of course, everyone would be involved in every hiring decision because if you bring in two people that no one can—or three other people can't work with, an organization can just fracture and crumble. And now I'm sort of just interested in getting a feel for people who come in and say, "Is that some DNA that will inject well into this system? Is that, you know, someone who is likely going to be able to deal with open source development? If you don't have experience and some don't, in developing open source projects, it's very different. It does expose you to, you know, that fishbowl world. It exposes you to a lot of—in many cases—unfair criticism and, in some cases, legitimate, but public criticism is hard for some people.

And having done it for six years, and done it in a very public way, and taken my share of public beatings, I think that it's useful for me to sort of, you know, sort of evaluate people along those lines as well.

So, I won't be doing as much of that anymore. I think we're a large enough organization. We have just sort of distinct enough teams now that it doesn't make sense for everyone in the organization to meet every person before they come in, and have that kind of input.

But I enjoyed it the last couple of years being involved in that, being able to sort of, you know, I think I have a pretty good feel for some of the things that have made us successful and trying to look out for those. And those aren't more of what we are. In most cases, those are, you know, trying to find some new piece of something to bring into the organization, and someone who will see things differently than the way we've seen them.

Tom Scheinfeldt: As the organization grows, and as the paid staff grows, how does that change the relationship between the paid staff and the volunteer community?

Asa Doltzler: You know, I think that—I think that you know, people volunteer on the project because it's either, you know, some interesting technical challenge for them. And we have a lot of people who are college students and other—a lot of young people, who look at this as, "Hey, that's an interesting challenge. That's an interesting intellectual pursuit. I'm going to go do that, and they'll let me in. And I get to work on something, you know, with a lot of other really bright people."

And I think that audience really doesn't spend—or that group of participants really doesn't spend very much time thinking about paid or not paid. It probably has more of an impact on the people who actually come in and are now paid, and who are now you know, a part of an organization that has a specific set of goals that are not always you know, what he was doing when he was out there scratching his own itch, right, when he was in it because he wanted to be working on this one problem. It's not necessarily the case that now he's an employee, he's still going to be working on that one problem, dedicated.

So, I think that it has an impact on those we hire out of the community, probably more so than those who are volunteering. But there's always been a large core of paid developers on this project, unlike a lot of the other open source projects out there. This project started with 100 percent paid developers, all paid by Netscape the day they open sourced the code. Two or three days later, they had a couple that weren't paid that were working on it and it's—you know.

But for the first four years or so, I mean Netscape dominated the developer and the developer crew. where they had you know, probably 80, 90 percent of the man-hours going into the project. And so, I think that we saw a bit of a shift when we sort of were jettisoned from that, and it was a real small team of four or five developers. Then we had this large community, some of them that were Netscape people; others who had been volunteers all along. But that sort of shifted the dynamic a little bit. That's also when Firefox and Thunderbird models came in, and we had a stronger sort of leadership role, or we had one person who was sort of guiding the application development.

But I think that, you know, it's never been the case that we were overwhelmingly open source volunteers. And so it's not weird that now we have a bunch of paid

employees. I think that we have hired heavily out of the community as sort of a nice way for us to acknowledge the value that these people have given by saying, If it is something you want to do, and you'd rather be doing this to collect a pay check, then whatever else it is you are doing, to be able to do that is, I think, great.

And I think that this organization and this project is fortunate that there are resources to be able to do that because there are a lot of open source projects with you know, dedicated people who are putting in as many hours as anyone in this project is putting in who won't have that opportunity because there just isn't likely to be, you know a revenue stream that can support that for—you know, SourceForge, 150,000 open source projects, and that was maybe a year ago that I saw their announcement. 150,000 projects. That's a lot of projects, right? And most of them are probably small projects with, you know, one, two, three, five people working on them. And I think very few of those are ever going to be in a position to be able to fund the development, and those who are interested in you know, full-time work. And we're in a position that we can do that. I think it's wonderful.

It is interesting to try to figure out how to be good stewards of the community, and where is the right balance in terms of do you put money out to community projects, or do you bring community into employment, right? And those are just some decisions that this organization's going to have to be thinking about for the next few years as it makes those decisions.

I'd love to see a mixed—I want to see us, you know, going in and, you know, we could be putting people through school, right? I mean there could be a Mozilla scholarship fund, right, that puts kids through CS programs, or, you know—that puts, you know.

I think we could be—we could be sending volunteers off to W3C Standards meetings to get a taste of those, right? We could be bringing people—offer developer events. We could be putting bounties on features and saying, “Hey, you know, here's \$1,000 for this bug fix. Who wants it,” right?

I think there's a lot of ways to use that revenue that isn't just hiring people out of the community, and I think we do ourselves a disservice if we decided that we were going to try to just buy up all of the community.

One of the things that really makes this project so exciting is that people are out there sort of fully independent but with, you know, a piece of connected vision. And the fact that they've got other interests, and are doing other things, they're bringing you know, new life all of the time into our project because they've got time to also be involved in two other projects, or employed by some other company with a complete different set of motivations. And that keeps us lively.

And I think the value in this organization, a long-term value, is all in the community. I mean we live well beyond the means of the organization in terms of the project, the product we support. There's no way that the people in this building alone could continue without the involvement of our community, could continue to be successful with Firefox. And I think everyone here understands that.

And it's one of the, you know, I was getting questions last summer when we were creating the corporation stuff of, "Well, what if the Foundation just decides to sell the corporation for a bunch of money?" It wouldn't make any sense because the value isn't in the corporation. It wouldn't be viable as just the corporation. We are—we are in a necessary and long-term partnership with our community, in a participatory arrangement that requires, you know, both sides. So, I think we need to do a lot to invest in the care and feeding of that community and to not just attempt to sort of co-opt it into employment. I mean that's my view. And I think that one of the things that makes open source special is that new blood is coming into it all the time because those people are motivated to work on that I think, and not because we've got a little more money to hire another person.

And I've only worked in one large software organization. I was housed at the Netscape offices for the first few years of my involvement in this project. And I saw that in operation. And that's not a great way to build great software. I think that we've had some good software come out of those systems and maybe great software can. But I actually think this is just a far superior model, and that's why, why I got involved with open source. And that's why I'm still involved with it today.

Olivia Ryan: Great.

Tom Scheinfeldt: Do we want to take a break and come back either later on today or another day?

Olivia Ryan: Sure.