Ken Albers: Today is June 26, and if you could just state your name.

Mike Schnoepfer: Mike Schnoepfer.

Ken Albers: Okay. Mike, when did you start using computers, or how did you get interested in computers and technology?

Mike Schnoepfer: I guess my first computer I remember using was a Commodore Vic 20, and this was in the, I guess early '80s. I've been playing with them ever since.

Ken Albers: Okay. And what's your sort of educational background? Did you have formal computer training or was it self-taught or—?

Mike Schnoepfer: Yeah. I, you know, played around a lot in high school and when I was a kid, and then I ended up getting a degree from Stanford in Computer Science. A Bachelor's and Master's in Computer Science is my formal training.

Ken Albers: And what would be the first programming project you remember working on?

Mike Schnoepfer: First programming project? Geez. I remember working on an—and I don't even remember what the game was, but it was a sort of Breakout-like game for the Commodore 64 when I was a kid. I don't remember the name of it, but it was definitely a game. This is many years—probably 20 years ago, so.

Ken Albers: And how did you first connect with the open-source community, or was your, you know, early contributions to open-source projects?

Mike Schnoepfer: Well, I'm pretty new to the Mozilla project in specific. I've interacted with other open-source projects in the past and not to as great a degree as other people here. I've ended up in a previous life using Postgres which is an open-source database, and so I'd interacted on the newsgroups there and helped some other people back with things like that, and some other interactions with some of the Java open-source projects. But Mozilla was the first time I had contributed in a big way is when I came and joined Mozilla.

Ken Albers: And what do you do here at Mozilla?

Mike Schnoepfer: So I'm—my title is technically VP of Engineering at Mozilla Corporation.

Ken Albers: And how was it that you came to get that position and work here?

Mike Schnoepfer: It was—I was introduced to a—I mean, I had been a huge fan of Mozilla for a long time and used, you know, early—I used Mozilla Suite and then, you

know, Phoenix and Firebird and Firefox. And then I got to know a couple of the Mozilla folks through a mutual friend, someone who sits on a board with them, and started talking to them. And this was when they were just sort of gearing up to grow a little bit more. And I—sort of a happenstance of timing I was looking for something new to do at that point in time and just over a series of months as we were talking about some of the challenges they had in terms of growing the project both from participation and full-time employees decided to sort of come on board and help out.

Ken Albers: And what specific projects have you worked on here?

Mike Schnoepfer: I guess the two biggest things—well, the biggest thing was shipping Firefox 1.5. So when I came on board we hadn't yet shipped the beta for 1.5 and didn't have a schedule for it, and so one of the first things I did was just sort of a lot of, you know, coordination basically. Just helping people get ready for the release, triaging bugs, setting a schedule and deadlines and release criteria. Doing everything we could to actually sort of set a final ship date. And we pretty far in advance set the, you know, target ship date and were able to hit it. And, you know, it's a really hard thing to do in software in general and a super hard thing to do in open source because you have lots of unpredictability because, you know, people have other things to do and aren't contributing back. And so more recently it's, you know, I've really helped out to staff up the Mozilla Corporation and try and get some more full-time help and then done a lot of the Firefox 2 planning then again to get ready for that, which we're just about to go into beta for.

Ken Albers: Have you generally worked alone or within groups?

Mike Schnoepfer: Yeah, I pretty much can't get anything done on my own. So it's pretty much impossible for me to do anything useful I guess here. So, I mean, a lot of interactions with folks here I've met, you know. Then fortunately I meet a lot of people in the community. You know, external partners, all sorts of things. I mean, my job is mostly helping to sort of coordinate, you know, what people are doing and help out wherever I can to make sure stuff gets done, and that's basically what I do.

Ken Albers: Yeah, because one of the questions we've been asking people who do coding is sort of how the division of labor occurs. And so is that part of what you do, facilitate that process? See how, you know, where people are working—?

Mike Schnoepfer: Yeah, that's one of the really interesting things. I mean, in a traditional company and every other company I've been at, you know, it's much more top-down assigned, right? Like, you have a goal and usually it's defined by some topline business objective or contract or sales deal or whatever it may be. And this is to varying degrees. And then, you know, the managers will

sort of divvy up the task and work with people and sort of assign bugs and go through it. Here it's much more organic and bottom up. So it's sort of more—people work on what's interesting to them, and the challenge for us is to sort of find out how to also get everything—everyone to be working on things that are interesting and things that they want to contribute on but also makes sense towards what we are tried to accomplish as a project and as a goal. So, you know, it's less sort of assigning things to people and more sort of helping to get a shared vision on what we're trying to build and also just sort of, you know, minor course corrections when, you know, people can work on other things.

There's an occasional time when, like, we need to get a critical bug done or something and I may, you know, ask a particular person to get it done. But usually it's—or assigning things based on ownership, so if people own a particular area and a bug comes up, you know, it's natural that goes to them. And that's, you know, a group process we do through bug triage so I spent a couple hours doing it today. I get a group of people to look at all the bugs that are upcoming for the next release, sort of figure out which ones are critical and which ones are not and then, you know, if no one is working on a particular bug, you know, suggest a couple of people who may be able to help out. And again it's more of like, "Hey, can you help out on this bug?" rather than, you know, "You go work on this now." So I don't know if that answers your question.

Ken Albers: No, you did. And how do you generally communicate during these processes? What forms of commutation do you use?

Mike Schnoepfer: Well, I think—I mean, the thing that's interesting about any open-source project is a definite bent towards electronic communication when you've got both time and language differentials. And so for, you know, a good example is talking to people in Japan, you know, or other countries for whom translation technologies can help. So, you know, command of the language on both sides can not be great but you can actually communicate successfully electronically, but doing it over voice is pretty much impossible. Not to mention the time differential. You know, and not everyone can afford to make, you know, long-distance phone calls and things like that.

So, and then electronic comes in many different forms. You know, you've got e-mail which is probably less prevalent here than in other places. Much more prevalent, you know, the number one form of communication on the project is actually the bug system. That's where, you know, 90 percent of the work of release gets done. We use wikis as a more, you know, to sort of distill topline information because it's easier to sort of see at a high level, you know, what's going on, what are the key things for this release on a wiki rather than trying to trudge through, you know, 300,000 bugs. And then IM

and IRC for sort of instant communication is a big one. You know, everyone here is on IRC. When we get up toward the release we're typically on an IRC channel chatting and getting stuff done in real time fast. And IM's another way to find, you know, people who are not on IRC but you want to grab quickly. So definite bias towards electronic commutation.

Ken Albers: Do you find any of those more or less useful as far as managing the project goes? You know, and documenting the project?

Mike Schnoepfer: Well, I think they all have their uses. I think, you know—and the trick is which one to use when. So, you know, IRC is great to get instant responses, but a lot of times whatever gets conversed there will get lost. And so you'll see a bug filed, like, "per IRC discussion last night," you know, such decision was made. And then everyone's like, "What IRC discussion? Like, what was—?" You know, there's a lot of context missing there. And you can go and recover that in certain ways, but that's just sort of one example of, you know, IRC is a great instant way. It's also—there's so many people on those channels that someone else may jump in with some piece of information you didn't know because Mozilla's a big project and it's a big system. So it's a great way to get people together on a specific topic but it's very noisy.

The bug system is great because everything is, like, persistent forever, so if you write a comment in a bug that will never go away. And again like I said it's also a little bit overwhelming. I mean, we have 100,000 bugzilla accounts. You know, 350,000 bugs. About a thousand comments get added a day. No single human can actually keep up with that sort of volume of traffic. So if you're just trying to, like, pop in once a week and say, "How's the release going?" you know, looking at the bug system is pretty much worthless. So we try to do things in wikis in a form that is more digested down so that other people who are sort of involved in the project but not on a day-to-day basis can sort of pop in and sort of figure out, like, you know, if I'm trying to test, you know, should I start testing now or later or should I start updating my extensions, those sort of things and try and communicate those in wikis or through the news. The other thing I didn't mention is the newsgroups, which is, you know, just old [NNTP] newsgroups, which is a way to have a persistent discussion that's not on a particular bug or not on IRC.

And I think one of the challenges we've had is there's so many different ways to communicate. There is a sort of funny thing where we're super open project so everything happens not in secret. But because there's so many different places where things can happen, sometimes it's hard to find information, and so people get frustrated because they can't find what they're looking for even though it's all there. It's just difficult to know, like, who to talk to and where is it and is this stupid page out of date, you know, or is it

current? It turns out you've just got to track down the right person and ask them and they'll give you the answer. So it's something we've been working on, too, to make it easier for people to approach the project. But that was a pretty long answer, so I don't know if it answers your question, but.

Ken Albers: How important do you think comments in the code are to the smooth development of Mozilla software?

Mike Schnoepfer: Why do you ask that question?

Ken Albers: Well, I'm just sort of, you know, trying to get at, you know, whether comments are a valuable resource for the developers or whether they are misused more often than not.

Mike Schnoepfer: That's an interesting question to ask. You know, I think our code could definitely use more comments. You know, comments just like anything else are useful in the right context and so, you know, code comments that are most useful are ones that provide information that you can't find by looking at the code. Like, why are you doing something in a certain way, or what are the assumptions you're making that aren't burned into the code? You know, the classic sort of funny comments are like the comment right above describing exactly what the line of code does right below it. Well, that's not actually very helpful because the line of code shows you that and you can figure it out by looking at it. And then when you change it and forget to change the comment, the comment actually does more harm than good.

So the key thing is providing any extra context you can, because that's the hardest thing, when people pop in and out and, you know, if someone worked on a piece of code and then it gets handed off to someone new, there's a lot of information that's lost there in terms of what that person understood, what problems they were trying to solve, what edge cases they're looking for that may not be obvious from what's there. And so things like comments or check-in comments or code comments are pretty critical to make that happen and to make code easier to approach for new people.

Ken Albers: Would you say that strict ownership of specific areas of code is enforced?

Mike Schnoepfer: Yeah, I think it's—code ownership is one of the key parts of any successful software project and particularly successful open-source projects. You know, it's the whole point of module ownership is that those people get to make the decisions that have the most knowledge about that code. They're closest to it, and so they are owning it. You know, the places where it's not enforced are when the module owner goes away, right? People transition and move on. Sometimes they let you know well ahead of time. Sometimes they just stop responding to e-mail and that's, you know, just the way things go. And so I

think that's the only time there's a little chaos is if a module owner's not being responsive but there hasn't yet been designated sort of a new person. Like, they didn't hand it off, hand the baton on the way out the door.

But the notion that there is a owner of at least everything and, like, if that owner's not there someone else will step in at least interim and do it I think is pretty critical to the project in general.

Ken Albers: So do you do development on actual code as well here, or—?

Mike Schnoepfer: No, I don't. No. I mean, I review patches and look at code and do things but I don't, you know, I don't have time to actually write code.

Ken Albers: Have you noticed any tension between front-end and back-end developers?

Ken Albers: Who fed you these questions? [laughs] No. Well, let's see. I'd say there's not a tension between front-end and back-end developers. We try hard to not have big divisions, you know. There's lots of easy ways to set up artificial divisions between developers, and so we try pretty hard not to do that. I think there's a tension between objectives. So developing new features for example and security and performance for example are often tensions you have. So I've added a new feature for example in Firefox 2 that allows it—if the browser crashes it will bring back the state of all your tabs. So if your machine crashed or whatever and you had ten tabs open and you restarted it, it will say, "Hey, do you want all those tabs back?" Right? It's a nice handy feature.

Well, it's doing more stuff so it may impact performance, and in particular when the browser starts it has to figure out whether it crashed before and if so where the tabs are and that sort of thing. So doing that in a way that doesn't make the browser start in general slower is hard and it takes effort. And so the original version of this patch slowed down start time on the browser, right? And so it was written by a front-end developer and someone on the platform team caught this, you know, and said, "Hey, you know, you're impacting performance." So they spent many, many, many weeks sort of refining this to the point where they were able to get it with no noticeable impact on start-up performance.

So I think it's less a tension between front-end and back-end and more a tension between future development and keeping the platform stable, secure, fast. And those are always natural tensions. It happens in the platform as well. Like, you want to implement a bunch of new features which may break a bunch of compatibility with existing plug-ins or extensions or code, and you have that sort of same tension there as well.

Ken Albers: To what extent do you think Mozilla has relied on the work of volunteers?

Mike Schnoepfer: What do you mean by relied on?

Ken Albers: Well, or to what extent have they utilized volunteers or how much of the production work at Mozilla utilizes volunteers, or the testing, or—?

Mike Schnoepfer: It's hard to categorize a sort of any, you know, specific number. I'd say that being open source and having passionate volunteers are pretty fundamental to what Mozilla is and its success. I don't think you could build a product like this. I mean, we're—you know, so we have the fortune of being able to have people work on the project full time and get paid, but we're still a, you know, very small staff compared to, you know, most people building equivalent products, and we'll continue to be so. You know, we're not going to be a 300-, 400-person company. So, you know, to some degree by definition—and we have a bunch of other things we do. You know, our multiple platforms and all the rest of it.

So, you know, by definition yeah, we do. We really rely on the passionate help of people, and it comes in lots of different forms. It comes in code contribution. It comes in particularly in testing. You know, people filing bugs and responding to bugs. It's quite amazing to see how rapidly if you, you know, file a bug, how rapidly people just organically will—who are watching bugs and who write modules and things like that will say, "Hey, I've got that problem too," or, "Here's a reduced test case," or, "I've noticed it's fixed by this bug," or, "Hey, this may be a duplicate of this other bug out there." Like, I filed a bug on Saturday morning at 9:00, 10:00 in the morning. Before I had actually finished attaching more information to the bug, someone else has commented in there that, "Oh, I think this may be caused by this other bug," right? So, like, in between my two comments was someone else clarifying the bug.

And that's one of the examples of just having such a broad community of people who care about the product to really sort of contribute. So I think it's pretty—you know, having volunteers, having people who are in the community is pretty fundamental. I mean, you know, we have on any given month a couple, you know—over 100 active committers to the code, and there are, you know, less than 30 on, you know, Mozilla Corporation's payroll. So some of those committers work at other companies, you know, that pay them to work on Mozilla full time. Some of them are volunteers. So it's a pretty broad open-source project.

Ken Albers: And what do you think it is that makes people volunteer, that gets people involved in this?

Mike Schnoepfer: You know, I have a hard time sort of generalizing the motives of people. I think lots of different reasons why people would want to, and I think everyone's a little different. I think, you know, one of the unifying themes is passion for the product itself, right? Bottom line is people want to work and participate and be part of building this thing that, you know, millions of people around the world use and love and, you know, promotes web standards and is an alternative and is secure and is open in the sense that everyone gets to communicate, work on it, gets to participate in it and gets to have some piece in it. It's a great chance for people to, you know, work on something that's used by millions of people. It's a product they use themselves often, I mean, so, or their friends may use.

It's a great way to learn stuff. There's just a tremendous amount of knowledge and skilled people on the project. So, you know, there's a great—one of the great things about open source is it's a great way for people who don't have a professional education to, you know, sort of take a hobby and turn it into a career or just improve it if they just want to as a hobby. Because by working with others, you know, you post code, you get it reviewed, people explain things. You learn a lot more. You work on IRC. So, you know, I think there's, you know, desire to be part of this great product that you love. I think there's—for some people it's, you know, a great way to learn and develop their skills which in itself is sort of interesting. Any number of reasons.

Ken Albers: Mozilla, Firefox particularly, has sort of become the public face for open source, you know, to a more general audience. You know, that's how a lot of people learn about open source and find out. And it's largely, you know, Firefox has been particularly successful. What do you think it is about Firefox that sets it apart from other open-source projects in that way?

Mike Schnoepfer: Well, I think, you know, fundamentally Firefox is the right product at the right time, right? It's a product people love that, you know, it's not something that they get by default. It's not something that's hoisted upon them. It's something that, you know, tens of millions of people went out and actively chose to do and install and download and run and continue to use and in many cases evangelize to other people. So, I mean, I think for most people it was just, you know, whether it be tabbed browsing or, you know, being sick of pop-ups or able to customize their browser with extensions or being faster and sort of a slimmer tuned-down UI, again every individual's reason for downloading is different. But those are, you know, probably some of the top.

And they've just used the product and love it. They use it because they liked it better than all the other alternatives out there. And honestly that has nothing to do with open source. I think there is an aspect to it, to open source, that's interesting. But, you know, fundamentally it's about building a

great product that people want to use because they like it better than other things out there.

Ken Albers: I mean, it's even been more successful than all the other Mozilla projects.

Mike Schnoepfer: Yeah.

Ken Albers: Do you think the same reasons there, too?

Mike Schnoepfer: Yeah. Again I think it's just sort of right product, right time. Being open source doesn't guarantee a success or sort of vice versa. I think the interesting thing about Firefox is it's definitely one of the bigger consumer apps out there. You know, you have Linux which is trying to break into the desktop but is more, you know, server and geeks and things like that. And lots of other really successful open source projects on a server like Apache and things like that are great products but aren't something that my mom would use or my dad. And Firefox is—I've converted my whole family to it. And I think it's just a—you know, there was a Mozilla Suite before that that wasn't that either because it was complicated and had lots of, you know, features and things like that. So I think it really comes to product design and it's really a testament to the people who worked on the original Firefox and, you know, were able to take the code base and turn it into a product that people loved.

Ken Albers: What would you say Mozilla's priorities are both now and moving forward?

Mike Schnoepfer: That's a great question and it's something we're sort of ongoing trying—you know, this ongoing process of figuring out exactly what it is. I mean, the tag line is choice and innovation on the Internet. I think one of the fundamental things—I think the thing that people here are probably most proud of even above and beyond the success of Firefox is how Firefox has continued to push innovation in the browser space. So, you know, you rewind four years ago it was sort of like IE was basically just, like, in deep freeze maintenance mode. They fixed critical security patches but if you wanted layout or dom bugs or JavaScript bugs fixed, you know, mostly forget it, basically. And so the sort of programming level that everyone used on the web was mostly stagnant because you had this, you know, 95 percent plus market-share browser that wasn't changing. And Microsoft wasn't particularly incented to change it because they had—you know, they wanted to lock everyone to the desktop and the browsers gave you alternatives.

So I think, you know, choice and innovation on the Internet fundamentally means giving people the tools they need to build sort of innovative, creative applications on the web that end users can come to with whatever the choice of browser is. They're not locked in to one particular vendor or OS or any of

those sorts of things. So, and the fact that there's, like, competition now in the browser space and innovation in all sorts of ways, whether it be through Opera or Safari. You know, IE 7 coming out which is a huge change in what's going on with IE.

So, you know, even if in five years Mozilla isn't here anymore but, you know, the world of browsers has moved forward a couple of notches and people are continuing to innovate on these web-based applications, I think that's a huge success. And that's one of our big priorities. That's why we push standards as part of being open source as, you know, we can't hide anything. People can see exactly what we're doing. People can participate. So continuing to push innovation on the web is probably our biggest priority. Part of that is delivering great products that people want to use because, you know, by doing that it gives us a voice in the web, right? We have a contingent of users that will, you know—if we implement the latest, you know, HTML standard or JavaScript 2 or whatever it may be, we have this platform of 50, 40 or 50 million users who are out there and now have access to this technology that websites can start to build and to put pressure on other vendors to also support the standard. So it's a great platform for us to sort of, you know, push that mission along of continuing to have innovation on the web.

Olivia Ryan: Is it difficult sometimes to strike a balance between pushing both innovation and standards at the same time?

Mike Schnoepfer: Yeah. I mean, I think you've seen the W3C for example which is the big web standards body was sort of stalled basically. And sometimes the standards bodies, and I don't want to pick on them in particular, can get off the track a little bit and get into never-never land of sort of things that people aren't actually using on the web. And a lot of the web evolved sort of faster and, like, stuff got standardized after it was used. And so we you've seen a bunch of folks here for example participate very heavily in the WHATWG, the WHAT working group, to push along new standards for things like off-line storage so web apps can store stuff on a local disk and do, like, off-line e-mail and all those sorts of innovative things. And then those sort of start to get picked up as quasi-standards, implemented by a bunch of browsers and then eventually moved over to W3C. Or the canvas tag which, you know, Apple originally implemented in Safari we looked at it and said, "Hey, that's a great idea," and again tried to push it through as a standard.

And, you know, so we try and work with all the other vendors and see if there's some way to get some form of interoperable thing in before all the ink is dry on the standard and get it out there in front of people to see what they do with it. Because once people use it and see it and it's cool, then you can get it sort of standardized.

Ken Albers: How do you think open-source principles have informed the marketing techniques of Firefox and other Mozilla products?

Mike Schnoepfer: Well, we don't do much marketing. I mean, if you think about all of the sort of marketing we've done, you know, whether it be the *New York Times* ad which was sort of, you know, again volunteer community driven, Firefox Flicks, which is all user-generated content, right? So people out there are just producing videos. Spread Firefox which is a community marketing site. You know, I think a lot of it is one-to-one marketing, figuring out how the community and the fans and other people can participate and help spread the word rather than sort of buying billboards on the highway. And so I think it's—the first thought on how we do it is how do we do something that sort of fits in with what we are which is an open-source, you know, project with a different mission than maximizing revenue and how do we leverage that as a strength and sort of get people involved? Because, you know, people get fired up and, you know, both the comments and the content. I'm sure you've seen the Firefox Flicks stuff. It's really cool, right? It's really neat to see what people do. And again it's an opportunity for people to have their work shown, you know, film makers to have their stuff shown and get it out in front of people, and so it's just sort of everyone wins by the process. So I guess it's much more community driven rather than sort of just straight top-down marketing.

Ken Albers: Have you ever worked on any other open-source projects? I think you mentioned before—

Mike Schnoepfer: Yeah. I wouldn't say I've worked on them to a great extent. No.

Ken Albers: What about with commercial products? Did you have background there?

Mike Schnoepfer: Yeah, so, like, in develop—you mean working on them?

Ken Albers: Yeah.

Mike Schnoepfer: Yeah. You know, I've worked at a couple of startups. I started a company myself and sold it to Sun Microsystems so I've worked on, you know, everything from digital effects software to data center management software in the commercial arena.

Ken Albers: And how is that different than working on an open-source project like Mozilla?

Mike Schnoepfer: You know what's interesting is it's really different in some ways and more similar I think than I originally thought. So, I mean, the first thing that's

different is that, you know, everything is open. So, you know, here at Mozilla our staff meeting, our project meetings, you know, are on a wiki that's public that has a phone number that anyone can dial into, right? So all of our discussions happen in newsgroups that are archived on Google. So, like, pretty much everything we do is completely out in the open. So that's probably the biggest difference by far. And so it means that, you know, anyone is welcome to participate, anyone is welcome to lurk and come in and see what's going on. So that's a huge difference.

You know, you have a lot of people participating because they want to, and so, you know, like I said earlier it's much more about finding out what people are interested in doing, right, and figuring out how that intersects with what the project's trying to do. And so that's, you know, that's very different than a commercial environment where, you know, you may have some sort of goal that you have to hit at and it's, you know, a question of what this person needs to do to accomplish sort of the corporate goal, you know, even if that's not really what they want to do. But, you know, you can paint this sort of black-and-white picture and this sort of, like, in open-source everyone gets to work on whatever they want to work on and in the commercial sort of you're in the military and you get ordered around to do exactly what you don't want to do. And neither one of those is exactly true, right?

I think the way that they're similar is, you know, if you—in an engineering organization if you get talented people who are passionate about what they're doing and get them to buy in on sort of what we're trying to accomplish together as a group, because everyone wants to see it succeed. Like, everyone wants to see a great product ship. Everyone would rather ship on time than slip. Everyone would rather have less bugs, right? So there's not a lot of argument in the end goal. It's just all the details in those conflicts. So if you can sort of get everyone fired up about that, then the two start to look pretty similar, right? Because people help out where they need to and they sort of self-select and say, "Hey, I can help with that, and I know it needs to get done, so I'm going to jump in and help." Or, you know, just generally get involved and fired up on it.

So, I mean, I could go through a lot more specifics but openness is the big one. The fact that you have volunteers, you know, it's probably the next big one. And then from there it gets to be more similar than you'd think.

Ken Albers: What do you think it is that defines sort of a successful open-source project? What makes it—you know, what's the driving factors there or elements or practices that—?

Mike Schnoepfer: Do you mean how do you create a successful one, or what does it mean to be successful?

Ken Albers: I guess you could look at it both ways.

Mike Schnoepfer: I think everyone will probably give you a different definition. I mean, for me it's I think you have to have a nucleus of people who are, you know, involved and passionate and take ownership over things, right? So that notion of code ownership. You know, sometimes people call it a benevolent dictator, right? Whether it be at the module level or for the entire project. He's sort of helping to organize and corral things. I think you've got to have some sort of shared goal. Like, what is it we're trying to do? You know, are we a browser? Are we an operating system, you know? And I think you've got to ship product, and product may be code that you ship that other people use for things or it may be something that someone installs directly.

But, you know, there's plenty of other, you know, Linux is one. Apache is, you know, obviously another huge one. And they've got a passionate user base, a couple of key people who really drive the project along, and they ship products that eventually get into someone's hands to be useful for something. Because no one wants to toil away on something that no one ever uses, right? Half the fun of working on something and building it is watching other people use it, you know, and having them get excited and sort of get involved. And so you have to have that part of it to keep it sustained over time, otherwise it's just sort of a side hobby that you do on your own and that's not nearly as fun. I mean, part of open source is social, right? It's a chance to interact with other geeks who share similar interests and experience and sort of getting to learn and interact with them and then collectively working on something and sort of producing it and having other people be excited by it and using it. And that's the payoff of doing any sort of, you know, development and stuff, so.

Ken Albers: Do you consider open-source software projects to be—and the work that you do—to be a public service?

Mike Schnoepfer: That's a really interesting question. I don't know how to answer that because part of me feels like public service implies that, like, I should be—like, it should be hard and I should hate it and there should be some sort of sacrifice. But it's a lot of fun, and so it's hard for me to separate that out. I actually don't even know how to answer that question. I think it's great that it's there. I think the world is a better place for it. I think that there's a lot more—I mean, the motives here are, you know, much more about that than, you know—are that instead of financial and other places. So I think that that's a big difference. But calling it a public service, you know, I don't know. That's hard to say.

Ken Albers: What if anything do you think the popularity of Firefox will do for the open-source software movement as a whole?

Mike Schnoepfer: Well, I hope that—I think it already has to some degree—starts to bring the attention of open source to the mainstream in terms of validating open source as a way. Two things. One is I think people have already figured out that open source is a great way to build software of high-quality because, you know, again you're harnessing expertise of people all around the world purely based on their ability to contribute, right? So it's just if they log into bugs and they write code and it's good code and people acknowledge, that's how they get authority on the project, not by any other sort of indirect means. And so that meritocracy to me is fundamental to—one of the fundamental reasons why open source produces good code. You know, it's open so it's out there for inspection and anyone to participate.

But I think people have sort of already figured this out. Like, Linux and Apache and a whole bunch of other things have proven that there's a lot of code we rely on that's written in the open. I think the thing that's a little bit different about Firefox is that you can actually build a UI and a user-driven application that people like in the open. And I think that's something that's a lot harder to do, to drive it where, you know, the target user isn't necessarily just, you know, another geek like me. It may be sort of a much wider audience. And so I think it's a demonstration that you can do that as well. And that, you know, there are ways to make sustainable open-source projects, right? Where you can have both volunteers and, you know, some financial backing in order to have people work on the project full time and sort of put more fuel on the fire. So, you know, I hope all of those things.

Ken Albers: And you sort of see that as the future of open source, where it's headed in general? Or, you know, what do you see ahead, down the road for open source?

Mike Schnoepfer: I think—I mean, the other thing I'd say is I think that one of the areas—whenever you talk about sort of very horizontal technology, like, if you look at where open source has been successful, you know, there's a couple of areas. One is a really horizontal technology. So what I mean by that is operating systems, right? Like operating systems used by thousands and thousands of people and you build lots of specialized applications on top of it. So, you know, you may do photo editing, I may do development or whatever it may be, but we all sort of use an operating system. And I think the browser is an equivalent horizontal platform used by, you know, hundreds of millions of people to do everything from surf to blogs to, you know, applications.

When you talk about sort of technologies like that, open source is really important because of that sort of innovation and choice element, because you don't want to really entrust one single company to completely own that horizontal platform because they're going to do what they're going to do which is try and lock people into the platform, try and monopolize interests and sort of, you know, use it to extract dollars out of it, which is usually to the detriment of pretty much everyone else. So, you know, I would hope that in any sort of those big vertical—big horizontal technologies that affects lots of people and affects the ability of other people to innovate on top of it, you'd see open source really take off. And so, you know, operating systems—you're seeing it in—you saw in web servers, you're seeing it in application servers now with JBoss, you're seeing it in databases, right? You're seeing it in browsers. These are all technologies that have broad applications that other people use to build sort of creative things and are useful not to have locked up by one corporate entity.

And so I'd hope that anything in that pattern would continue. Cell phones for example would be a great area for open source to take over a little bit more and break some of the gridlock of the carriers. And I think a lot of corporate entities—you know, I spent much time at Sun, and they've been trying to open-source sort of everything they've got—will start to ask pretty much every time, like, should this project be open sourced or not, and start to think of that as a deliberate question rather than assume it shouldn't, like, unless, you know, someone tells them otherwise. And I think that will be to the benefit of lots of people because it will sort of foster sharing of technology in lots of important ways.

Olivia Ryan: What do you think some of the challenges will be for a big company like Sun or some of these that have been around for awhile in going open source? I mean, Netscape would be an example of how to—

Mike Schnoepfer: Yeah, it's s a huge deal. I mean, there's a whole bunch of logistical problems, so, I mean, just on the practical level most of these people don't own all the code that they ship, right? They may have licensed technology from other vendors and that license probably doesn't allow them to redistribute it in the open. So that's one of the hardest parts is just, like, figuring out which of the code you actually own and you can ship and separating that off, getting all the right licenses in there. You know, there is—a lot of people are embarrassed by the code they have. You know, they actually don't want people to look at it because it's either terrible or, you know, has comments they don't want people to read, which is true in a lot of code. So there's lots of logistical problems. I mean, that's like once you decided you're going to do it what you actually have to do.

But then there's a big emotional problem. I mean, you know, software companies, provider software companies, you know, their fundamental sort of asset is the IP, is the code that they write, or that's the way most of them were built. And that's what they sell, right? So they, you know, you pay me money and I give you sort of some binary bits of code that I wrote. And open sourcing it in some ways is sort of giving it away, so it's sort of turning what they have into a commodity. And so that's a big business challenge, right? So you have to replace that with some other thing of equivalent value for them in order to make that possible. So that's why you're not seeing Microsoft open source, right? Because it's not good business for them. So I think it'll be interesting over time. I don't think the answer's necessarily that everyone has to open source everything. That's why I try to make a distinction between horizontal and vertical stuff which is, you know, people have a right to try and make money on, you know, things if they want to, and if they can add value and people find it valuable. It's just when sort of they can take it over and squash innovation in other people where it really becomes a problem and I think where open source is most valuable, so.

Ken Albers: Do you think that these sort of open-source techniques can be applied in other areas of production of society beyond software, beyond technology production?

Mike Schnoepfer: In other sort of industries and domains?

Ken Albers: Yeah. Or even academics or, you know, any area where people are—.

Mike Schnoepfer: Yeah, well, you are seeing—I mean, so content generation, that's one easy one, right? So Wikipedia's the obvious one. Wikipedia's effectively an open-source database. Open-source encyclopedia, sorry. And, you know, that's worked out pretty darn well and funny enough isn't actually that much different than the way some of the early dictionaries were built, right? They were sort of farmed out through lots of volunteers who'd submit and do peer review. And if you look at academia, right? Academia's all about in theory peer review, right? You publish a paper. It only goes into a journal if, you know, peers who are recognized authorities in that field review the paper and think it's of merit and should be published, right? And your results are out in the open and reproducible. And so and, you know, in theory you're being judged on your merit by your peers. So I think the concepts are not that different than things used in other industries, for sure, in terms of how they're—you know, having open— Really when you boil down to it it's like everything is in the open so there's open communication which is important because it means that there's no sort of secret handshake you have to do to jump in and participate, right? And then, you know, your level of participation is defined by sort of your level of contribution. If you're, you know, writing good code and saying things that other people think are helpful

in the project, you'll sort of get more authority and participation. And so meritocracy, so openness and meritocracy are probably some of the key elements of it. I think you see that in other industries as well—in other places.

Ken Albers: Okay, great. Thanks.