Olivia Ryan: Okay we are here with Mike Beltzner and it's June 5th 2006. So Mike when did you first begin using computers?

Mike Beltzner: Wow, ah -- my first "ah" of the interview, this does not bode well. The first computer that I ever encountered was a Hyperian, which my father, who was working for Statistics Canada at the time brought home from work. It was, what's the best way to describe it, like a very wide package, probably about, I'll use inches here, probably about four or five inches in height and about a foot and a half in length and then another six to eight inches in depth. And it was huge sort of a behemoth thing that had two floppy drives and a very small monochrome screen. And he brought it home and said, you know, this is interesting to play around with this and then he put me down in front of the BASIC programming language and had me tinker around with that. I think I was about six or seven years old when that happened. And then shortly after that our family got a VIC-20 and that was the end for me, I just sort of started poking and probing around that time.

OR: Oh great, are you entirely self-taught or did you also have formal computer training?

MB: I am mostly self-taught. I did a program at Queens University in Kingston, Ontario called Cognitive Science, which is a mixture between computer science and psychology. So I don't like to say that I am formally trained in computer science because I haven't taken C programming courses like a lot of my colleagues have. I have been in class myself with a software developer. I understand software languages, I can read through code, if you give me enough time I can generally even understand the code I'm reading. But it's not my forte. I did do a lot of programming of course in school, both in high school and in university. I was very good with this courses so I just tended to pick them.

OR: Okay, and what sort of first project -- I guess how would you classify yourself now if not as a programmer what would you say?

MB: So I call myself a Phenomenonologist, which is a hoity-toity name because we don't like to pick titles that box us in. I was hired to be the User Experience Lead for the Mozilla Corporation, and the idea behind that is that I try to make sure the user experience that we put across, not only in our product but also in all of our sort of

user facing aspects, so that's the websites the marketing campaigns that we put out, I try to sit on the teams that are crafting all those messages and make sure that we are putting across a friendly, positive, and open user experience, the type of thing that we want our users to feel when they use the products. As a result I interact with developers an awful lot, I do a lot of software design, whereby design I mean sketching how the UI should be and maybe talking about how we can simplify the user facing portions by taking on more from the implementation side to automate certain tasks for the user. So at that level I need to understand how computers work, so that I am not asking impossible things of developers and also that I am understanding where we can potentially automate tasks. But I don't try to delve so deeply into the code that I am writing those interfaces myself. Part of that is intentional. It's generally considered to be good to have a degree of separation between the implementer and the designer, simply because as a designer then you don't get boxed in by the limitations of an implementation, and you try to work around things. But at the same time, having some knowledge of what is a limitation of the implementation is very required so that your developers don't throttle you when you send them an e-mail.

OR: And how do you generally communicate with those who you work with?

MB: Depends very much on the project that I am dealing with. So marketing for example, we do that mostly through e-mail lists right now. There's some news groups work, but marketing is an interesting sort of thing and I will be very interested to hear the interview you have with Paul Kim. Marketing is hard to do in open source sort of forum; we try to as much as possible, but I would -- for marketing efforts that are on Spread Firefox, they communicate through the Spread Firefox message boards. When we are talking about changes to code, we do that in two places we discuss designs and implementations and the pros and cons of them in a newsgroup called mozilla.dev.aps.firefox, and I'll also file bugs and comment on bugs. So every morning I wake up, I load my browser, I load up G-mail, and I take a look at the bug mail that's come in and that's usually about 10 to 40, and so keep track of the design discussions that are happening there as well.

OR: Do you ever feel like, maybe I'll put it this way, are there certain means of communication that are more effective than others?

MB: Certainly, so design is interesting. The easiest – or, I should say, most productive way that I have ever experienced design is when you get three or four people together in a room with a white board and a marker and a problem. And hopefully some data about what users are expecting and what the tasks are that we are trying to enable and what the primary use cases are. And you sit there and you figure out what is the user's problem, what are we trying to help them do, how is the best way to help them do that and you start sketching things on a white board. And back when I worked at IBM, that was pretty much the way we designed all interfaces. Its harder to do that in open source sphere, of course, because you don't want to limit the involvement of that design discussion to just four people. And so the forum that I have found now most useful is to start a design exercise perhaps with two or three people shooting an idea around like that and then take the design artifacts, either through digital white board photos or what I tend to do a lot of is what's called ASCII Art, which is trying to mock up what the screen is going to look like using only characters that you can write into a newsgroup, using a mono space fonts, so a lot of pipes and dashes and brackets and stuff to make it look like buttons, and then you put that idea out there for comment and you try to get people talking about it. Where I'd like to take that process is making slightly higher fidelity mock-ups, perhaps even interactive mock-ups using the XUL programming language and using Firefox as a platform, so that you can quickly show someone, 'this is a sort of experience we are looking to portray what do you think of it,' and just generate comments from that and then rapidly iterate based on the feedback. The best communication means in a distributed team like we have at Mozilla, its got to be newsgroups or e-mail, and the way we do our newsgroups is a newsgroup can be sent to somebody as an email list, so they then would have their preferred way of interaction there.

OR: Is it sometimes difficult for new comers to get involved in a new project if they not sort of aware of where discussions are happening?

MB: Yeah, and that's probably one of the largest problems we have in terms of user experience, the user experience for new people into the project. And we have talked a little bit about that and how we might benefit from having a buddy system, but the real weakness we have right now is if you come to Mozilla and you say, you know, I want to participate, there's a certain barrier to entry. Some of that barrier to entry might even be intentional to try to weed out some noise from the signal that we could get from people in the world. But some of it is also just a matter of the people who are working on the project are also, -- they have gotten so accustom the way we do things that it's second nature, and so by the time you become accustomed to it through trial and repetition, you no longer -- you are so involved in it, you no longer have time to write down how you got involved in it. So we need to get better at that. Mozilla.org, I think, needs a bit of a refactoring at this point in time so that were you to visit that website and say, I want to participate in Firefox, you can go a project page and find out all the different ways you could participate and then from there link over to the various discussion forums. There are a couple of efforts; there is a discussion currently going on in the newsgroups, I believe its headed by Ben Goodger, to try and open up that process and make it a little more accessible for new comers. But it was something that most people I know who become successful in the project have gotten used to the way we do things, and the way they communicate simply by trying and jumping in and observing and seeing how things happen. Sadly, a lot of that requires that you learn how to use IRC and jump into IRC and sort of lurk in the channel and see how things work. That's certainly the way I learned.

OR: Yeah. You describe your communications a little bit between people who work on the front-end and people who work on the back-end.

MB: Yeah.

OR: Are there any kind of natural tentions between those two groups, in terms of, whatever, philosophy, or --?

MB: I don't think so, certainly nowhere near as much as in other organizations where I have worked where the teams were extraordinary decamped. And I think a lot of that is because that there's such a -- we have a wide group of developers who

participate on the project and certain people live in their little niches, but a lot of our people who are, what I would call leaders of development, so the module owners, the people with super review, tend to sort of strattle areas, right. So, let's use one of my favorite people Vladimir Vukicevic as an example. So he's one of our core graphics hackers right now who is working on the new Cairo back-end graphics platform for Windows, Linux, and Mac, with Stuart Parmenter and some other really good people who are down in the dredges. But both Stuart and Vlad will also throw in a comment on the front-end bugs and they both got very strong design senses, so we don't tend to have that decoupling you have between sort of the front-end camp and the back-end camp, and we certainly don't interact in the same way as lot of other organizations, where the back-end camp sort of writes to some sort of contract named by a product manger or the front-end person nor vice-versa, so its not a matter of them developing an isolation, its all very in plain view of each other. And I think that ends up being a bit of a strength for us because the two camps are bit more in sync. And nobody is sort of considering, you know, that's back-end responsibility or that's front-end responsibility, it is a whole project sort of thing. That said, the discussion for those two sort of aspects of the project is done completly in isolation of each other, and part of that is the current branch management that we're on. So right now Firefox 2 is being developed on what's known as a stable branch, so specifically it's Mozilla 1.8.0 branch, no wait, that's wrong, 1.8.1 branch. And Firefox 1.9, which is the new sort of larger back-end development, is being targeted for Firefox 3 or Gecko 1.9 is being targeted for Firefox 3. And so as a result the two camps haven't had a lot of reason to interact now, but we are already starting to look at planning Firefox 3, and those meetings will be Gecko 1.9/Firefox 3 meetings, so it will all happen together. So I haven't really experienced that. It's a long winded answer.

OR: Yeah, no, that's great. Would you say that strict ownership of specific areas of the code are enforced, is enforced?

MB: Yeah. Module ownership -- we are getting into the area where I should mention that I have only been working deeply on the project now for a little under a year and I am still very much learning sort of pros and cons of module ownership the Mozilla way. What I would say is that from what I've experienced, module ownership is very much in place, fights about it don't come up all

that often. Specifically with Firefox 2 the way we have been doing and managing all of our code to this point anyway, its going to start a little more restricted now that we are approaching beta one, but the way we have been doing everything is that check-in review was simply approval by module owner. So the module owners really did have the little fiefdoms and the process of benevolent dictatorship over the module seems to have been working well. What it really counts on is the module owners being reasonable people who are willing to have their decisions questioned and at least listen to arguments about their decisions. And to the most part I have seen that happening. And disagreements are going to happen and when that happens you trust the module owner and you trust the process to at least to arrive at the right decision. And if it doesn't we can fix it a little ways down the line. I think a guiding principle that I was introduced to early on was from Mike Shaver who told me that perfect is the enemy of good. And I see that a lot in our project; we are not striving to be perfect with every release, we're striving to be as perfect as is pragmatically possible. There are certain things where you do strive for perfection, but at a certain point that's just going to bite you, because we would be spending too much time debating little semantics when really the best way to figure something out is to get the product out there and see how people are going to use it.

OR: To what extent has Mozilla relied on the work of volunteers, and do you, if you know, if that reliance is sort of changed over time?

MB: I would say to a large extent and I would say the reliance has changed overtime but probably not the way a lot of people expect especially since the organization corporation split. I would say that now more than ever, we rely on the work of volunteers. We rely on it in everything, perhaps most importantly, volunteers in terms of spreading the word of it, Firefox. I think that the efforts of our, what I'll call marketing volunteers, go largely unsung. The efforts of our localization teams who are largely volunteer also go largely unsung. We have greater penetration in Europe than we do in North America, and that would not be possible without our localization teams. I think our QA volunteers and just the people who hang out in IRC and help diagnose problems and triage bugs, that is a huge, huge HR investment which we sort of get for free because people want to be involved in the project. And these are things which

anybody really can do, you don't need to have a deep understanding of computers to be able to sit there and help somebody through a problem and if you can't solve that then realize, maybe this is a problem that actually exists in the product so we need to file a bug on it. Or to just sit there and look at bug reports and try to reproduce what somebody had said and say whether or not you see it as well.

Chris Cooper and the QA team have been working – and Zach Brown as well -- have been working on this project called Litmus, which is a way of allowing anybody who is interested to go to a website, and what they'll get is a series of steps, for what's called the Smoke Test, which is like a, a Smoke Test is a set of predictable user actions which would result in a predictable system reaction. So, for example, hitting Ctrl T, and getting a new tab popping up. And whenever we spin out a new build, when somebody's check-in may have broken that behavior, what we need is to make sure that these basic tasks still work from release to release or from nightly to nightly. So, if I was somebody who knows very little about computers but wants to get involved in this cool new Firefox project, I can go to this website, get that Smoke Test, follow the steps and say, yep that worked or no that didn't work. And that in itself is extraordinarily helpful to us because it means that we can take somebody who is perhaps more knowledgeable of the code and put them on something which requires somebody use more knowledgeable of the code.

In terms of design, which is something I'm very interested in, I have been -- and here's where I get to smack down my own people -- I've been very disappointed in the support we have gotten from design volunteers. People like John Hicks, Kevin Gerich, Steven Garrity, the silver orange team, these people have sort of jumped into the breach for Firefox 1.0, and built a Visual Identity for Firefox, and Jon Hicks obviously did the fantastic logo for Firefox. And these pieces of work were largely spurn by volunteer effort, but since then the design community, the usability communities haven't really been engaging in a direct fashion with our community on making our product better. And that's a little bit of disappointment; I actually see it as a bit of a personal failure because I haven't been able to bring them into that. And I have tried a couple of different things, and I don't know if it's a, if it's a tools problem, if it's a matter that open source is seen as kind of the simply geek, and its

not plastic and shiny and sexy, but it's something I would like to get a little better at, because I think that as we grow we need to able to work asymmetrically. We need to be able to do more with less in order to sort of compete in this market where there is some very significant competitors with very large budgets and without the volunteer efforts, I think we are pretty dead in the water.

OR: And so why do you think people volunteer?

MB: I think people volunteer because they like to be a part of something. I mean, if you go into Firefox right now and type about colon credits, you'll see this huge list of names, and that sort of gives people a big community feeling. It let's people realize, you know, I am in control not only of my computer in terms of I can chose what to click on, but I can also be involved in the creation of the software this computer runs on. I think the idea of a bunch of people working together to build something that's better for that bunch of people resonates in a lot of cases. And I don't even know if that's a non-profit thing, people get experience out of it, people get knowledge and understanding out of it, I know that what first attrated me to the Mozilla project was this idea that I can finally understand how software development works, and I could understand how large scale group collaboration could work, and all that's very exciting. There's a large aspect of our community as well which is just purely social, and you get all these people sort of working together on this project, you are going to get jokes, you're going to paling around and laughing around, and there is a fairly large sense of what Cory Doctorow would call Whuffie coming out of that, right, you get some social capital. And I think that's – it's underestimated what that's worth. And I think that our competitors are learning. So just the other day I saw a blog post from the head of the Microsoft Excel team, and what they were talking about was the Excel 12 or Office 2007 preview and how Excel's got a whole bunch of templates for charts, and everybody who uses Microsoft Office knows they come with templates and you got like six or seven and it's the six or seven that the Microsoft design team have decided most represent business. And what had happened was there were a lot of people who were saying that the Excel templates weren't really that professional looking, and the colors were garrish and whatever. And so he has put (inaudible), and said, if you think that they're bad, we are willing to take submissions. And I think that's very powerful because all of a sudden it shows we are

listening to our users, we want you to be part of the process, and we want you to feel a little bit of ownership of the product. So I think, all of those things are being the reasons why people participate.

OR: You mentioned, you touched on marketing…

MB: Hmm.

OR: And people volunteering to help with those marketing efforts. Do you sort of see, think that professional marketers are necessary to the process or do you sort of think that developers and designers can sort of start off something like Spread Firefox or whatever and the community can just sort of run with it?

MB: I think having professionals in place to guide the efforts of the community is always extraordinarily valuable and the same reason why -- and I'm in Canada so I'm going to use a hockey metaphor -- the same reason why the teams that generally win the Stanley Cup are teams where you got somebody who has been into the playoffs before. And newspapers who always talk about it, you know, the benefit that team gets from having a veteran, and it's just understanding what's going to happen and being able to predict how people are going to interpret something. So, if I were a volunteer marketer, I might think that a slogan I have for Firefox is fantastic without realizing with the sort of professional expertise that an experienced professional gets, how that might be misinterpreted, how legal implications apply, and even just how to sort of shine and polish rougher ideas. And I think that also encourages a learning cycle which then helps those people who were volunteers get the professional experience they are looking for. So, I don't think that what we are advocating is a destruction of all professional realm, I think what we are advocating is a more colleagial and collaborative atmosphere where professionals work with the volunteers to help share their experiences and grow that way. I am a big believer in the co-op sort of model of education where people learn most by doing and observing people who do, rather than by sitting and reading books. I think reading books is good for foundational knowledge. I think that, once you get in it, foundational knowledge is important to have but applied knowledge and experience just trumps it every time. In the specific terms of marketing, I think there's also a certain industry requirement that

the recipients of traditional marketing avenues require. So no one's going to go to a press conference if that press conference isn't done in a way that press conferences are noramlly done. Once you get the credibility of having run a couple of press conferences one way, then you can start branching out and experimenting. But there's a certain sort of low bar that you need to gain credibility, and I think that sadly or no -- I'll let that be something that you and the listeners can decide -- but that's something which requires professional sort of entry.

OR: And why, if you know, or if you have sort of an opinion…

MB: I have an opinion on everything.

OR: Why then -- the marketing efforts that have been sort of open sourced, if you will, have focused mainly on Firefox, and to a lesser extent Thunderbird, and have not really focused on other Mozilla products?

MB: I think it's a matter, that's interesting, I actually, I'd just ask which other Mozilla products you were thinking of?

OR: I guess any of them, Sunbird, Camino...

MB: Sunbird, Camino...

OR: Yeah, Minimo…anything.

MB: Seamonkey, sure. I think a certain part of it is to not bifurcate the message to make sure that we're -- and this is where I am going to start postulating from user experience perspective -- it's a lot easier to tell somebody three things than seven things. It's a lot easier to point somebody to this is the good answer for web browsing. I think that we are careful not to slam self-competitive products, is that even a term? Things like Camino, and things like SeaMonkey. I actually have all the browsers installed in my system, I use them for different purposes. Some of them valued (inaudible) -- I think, what we try to do is push a product message because from a purely receiver's point of view what I care about when somebody comes to me with a marketing message is what's it's going to do for me, and it's a lot easier to say well Firefox will make your browsing life better for these ten reasons. And then there is a

secondary thing we push the and it's open source, and here's all the benefits of it being open source, and the community aspect of it, and how you can even get involved and control it's destiny. And in that second message we sort of give benefit to the other products.

Sunbird and Minimo are interesting examples, I think those are products which we would should be talking about more, not necessarily to the mass consumer public but to the developer public because I think those are two very exciting projects which we need a bit more focus on. Mobile browsing, everybody tells me, is all the rage. I have been trying to mobile browse for about three years now on a variety of different cell phones, and I've found it nothing but a sea of tears and disappointment. I now have a Blackberry, which doesn't have the greatest browser in the world, but the way that I end up mobile web browsing the most is I hop on an IM client or IRC through my Blackberry and ask somebody a question, because that's just way quicker than any other sort of mobile browsing experience that I've been able to find. But Minimo is very exciting and I know Doug Turner is doing some great stuff with it. And getting more people focusing on what mobile browsing is actually going to be and not letting the cell phone providers and the data carriers and the wireless carriers sort of dictate what mobile browsing is going to be, I think is something very important and we need more focus on.

Similarly with Sunbird, I think calendaring is something which is going to have a lot of tie-ins both to mobile browsing and to personal information management and how sort of ubiquitous computing ends up, getting implemented. And there again, there is a fairly open standard called iCal but that standard has become extraordinarily fractured and it seems that everybody says they are iCal compliant, and yet if you need to get two iCal compliant vendors, for example, Sunbird and Apple iCal, trying to talk to each other, there's problems. And its because each is supporting a portion of the standard. So again there, having more people look at that problem and especially more industry people like Lotus, people -- I know that Sun is doing a lot of work right now with Calendar, and Oracle has been as well. But I think vendors like IBM need to get in there as well and Apple and really work out how can we make it so that these products -- there's never going to be a one winner for Calendar -- but how we can make it so that they can add value to each other by being interoperable at which point the differentiation

becomes which is providing the best user experience for a task. Let the data be free, (inaudible) and just focus on how you are actually going to add value to the user instead of by locking them in. So I think we should be pushing those messages a little more, but again, not to the consumer audience, to the audience who wants to help and has the skill to help control the destiny of that.

OR: So for the consumer audience, you think maybe its just too much noise?

MB: I think, right now, the products aren't ready, is the basic and sad truth. And I hope neither Dan Mosedale nor Doug Turner will hunt me down and kill me for saying that. But they are both, I mean, they're also both dot releases right now, neither of them hit a 1.0. Calendar is raising towards the 1.0, and it's the one that I have been interacting with the most aside from Firefox right now and I think they got some really exciting stuff going on there. So, if you haven't interviewed Dan Mosedale or listened to the interview yet, you should, he is a very interesting guy.

OR: Okay, thanks. So, I know that you've only been with Mozilla for roughly a year or so but you may have some insight into this. I understand during the early development of Firefox, the CVS access was restricted to sort of a limited to group of people and….

MB: That's even true now.

OR: It is, yeah, so was going to have you discuss how the access has sort of shifed or why that was and why you think that was beneficial to the production of Firefox.

MB: There's a discussion going on in Mozilla.dev.planning right now started by Mike Connor about changing the way CVS access works. And again, so here we are stepping up my expertise a little bit, but what I will do is a sort of give you my observations, which is, the more people that have CVS access means the more sort of higher level fights you could have. CVS access almost represents the last control, it is the gate to actually making a change to the code and the gatekeepers historically have been the people who then control what's in the code. As the project grows, the benefit of them being a gatekeeper turns into a detriment because all of a sudden they also become overburdened and they become a

bottleneck in the process. And so, from that side there's an argument to increase the number of people who have CVS access. From the downside, one thing that's really important to our process is being able to have stable builds at the end of the day. So, we need to know if we give you the CVS access and you, what's called break the tree, and as a result the builds stop working, we need to know that you can then fix the problem even if wasn't your patch. And even if you check in someone else's bad code, you need to know how to fix it so that as soon as possible we are not stopping all of our internal testing, we are not stopping the work of other developers who then have to wait for the tree to be fixed before they can make sure their code won't break the tree. And just for that reason I think limiting CVS access to a trusted group is very important, and also making sure we don't get, you know, too many commits that clobber on top of each other.

And in terms of how it restricts the variety or creativeness of the people working on the code, I have very rarely seen somebody refuse CVS commit as the way of stomping on a design decision or stomping on a change to a code. It generally gets stomped on way earlier than that. And generally by the time it gets to CVS commit, the module owner, or whomever has actually already agreed to do something. From what I understand early on in the Firefox development process CVS commit was very limited simply because they wanted to move as quickly as possible, and there was a core group of them, and they didn't want to have to have check-ins that they didn't understand, and they were all co-located, they were sort of -- It was almost like you had five people in the room, and at that point CVS commit is just limited to those five people so that you always know what's going on, and you don't want to give it somebody outside the room because then you need to understand where the check-ins coming from. That's really no longer the case with Firefox so that model has changed a little bit, and that's probably also why. Maybe, I guess.

OR: And does this differ, if you know, from any other Mozilla products? Are they run sort of similarly?

MB: Yeah there I really I can't – I don't know.

OR: What do you think Mozilla maybe in particular Firefox, or, I am sorry -- Why do you think they are unable to attract so many users, in particular Firefox?

MB: Selfishly, I will say that it's because of the area which I am now responsible for, which is that Firefox simply did things better than the competitors where things is defined as the set of things that the majority of people want to do when they're browsing. I think even hard core browsers, the type of people who ask for what I'd classify as a hard core features, so the ability to turn on or off the rendering of a certain tag in the rendering engine, liked Firefox for the fact that they could easily modify it to do exactly what they wanted. But in the majority of cases Firefox adheres to the principle of get out of my way and let me browse. And it focuses on the content. You see this a lot in Safari as well. Safari has the least amount of what's called chrome or user-interface layer that belongs to the browser, and it really tries to let the user focus on the web, which is what they are trying to get to.

I think there's a certain amount of people who are attracted just to the idea of it being an alternative to Microsoft. There is a security model argument there which a lot of people think was one of the reasons why we became successful, and it may well be; I don't think its necessary as compelling an argument. I honestly don't think the majority of users are aware or as concerned with fishing and spyware, they find these things annoying but where that actually manifests itself is that Firefox had fewer popups and Firefox said eventually your system will work better, and people started noticing those correlations, people started noticing that it was more convenient in a lot of ways than other browsers. We try to focus really on the simplicity of the install experience, we try to focus on the simplicity of managing your information. And then another reason why I think we attract a lot of users was that we got a very rabid fan base and we let the marketing be up to the community, and they just really pushed the message out there and grew our audience that way.

OR: So I am shifting gears slightly, how would you maybe list Mozilla priorities?

MB: Ah, the Mozilla priority question. You must get such fun answers to this one. Shifting is how I'd rate Mozilla's priorities. Our

organizational mission is to promote choice and innovation on the Internet, and I think that is still our priority. And the way we express that priority is we try to figure out how do feel with our current resources and our current position and our current strengths and weaknesses, we can best accomplish that task. And adding thrust to the Firefox movement seems to still be the best way to do that. We have a certain -- depending on stats – anywhere between 10 and 15 percent of North American browsers working on Firefox. That's a dangerous number. It's enough to get comfortable with, its enough to say yeah we are now a competitor. Its also enough that it can go away fairly quickly. And you can look at it as, you know, yay that's 10 percent Microsoft and other companies don't have, or you could look at it as its only 10 percent they need to get back. So I think we need to shore that number up and I think we need to make Firefox even better and make it a more convincing value proposition for users and I think that gets us a lot of things towards our original mission goal in that we make sure that there is no one controlling stake for web browsing. We make sure that there is a choice that users have and that there is an active sort of community and movement of innovation and development about web technologies. And that it's always moving forward, and its never a matter of well, we've gotten all of the markets so we can ease off on that product for a while until we go somewhere else. So I think for that reason our number one priority overwhelmingly is still Firefox.

I think that we need to be very strategic about the way we invest money in some projects, in terms of what value -- not we as a corporation can get out in terms of profits and dollar and cents -- but what sort of value our parent foundation can get out of us adding thrust to the other projects. And so far I haven't seen the value for investment proposition coming out of any of the other ones that are sort of staying around at the lower levels. We are seeing more and more email users use web based platform for e-mail, which is sad because I like Thunderbird but it means that Thunderbird probably needs to recreate itself a little bit to make sure its still relevant and important for users. And now Scott McGregor is going to come headhunting after me, but that's okay. Similarly with Calendar, I think the Calendar is very interesting to subset of users but its not global enough yet, I don't think that there is a way that we can change the computing world through Calendar the way that it currently exists. I think there's some really good potential in there. Similarly with the Minimo. And so I believe

those are the projects which we are investing in, but until we see how they are going to change the Internet and that world, the primary focus really still needs to stay Firefox. Because there's a lot more things we can do there. We are not bounded by opportunity or number of things we want to do, we're bounded simply by time and bandwith to do them all.

MB: Okay, in talking about sort of open source generally and broadly, how might you define a successful open source project?

OR: One that gets used. And not even necessarily used by end users, but even if an open source project gets reused in other bits of code, I think that's successful. One that is active and maintained, I think shows a sign of success. So there's a lot of open source projects out there that are sort of stillborn. There's a lot of other ones that are very, very active, and can be reused, and can end up making it easier for – and here's where I'll start sounding like Larry Lessig -- easier for other people looking to do similar things to build upon the creativity of others. We all stand on the shoulders of giants. I mean I'm talking to you using a microphone that was invented based on design, based on design, based on design you know back in time. And I think we are all better for the ability to create off of the creation of others, so I think that the basic metric of success for an open source project is that it gets reused. Some of them aren't going to be sexy, right. I mean Firefox is a little sexy and it's a web browser that brings you know the cool new web apps, it's hip, it's web 2.0. Some of the open source projects that I think are most important are really foundational, the open source bits on which the majority of the Internet runs, the Apache web server, Linux itself. I think those projects are more successful than people give them credit for. And I think Linux has a lot of ego problems because it hasn't succeeded on the desktop but I think that what Linux needs to continue to be proud of is that they run an awful lot of the Internet. And there is a huge value proposition in terms of making sure that they can run Enterprise level Internet, and Enterprise services and servers, and that frees other people up to be more creative and eventually to create a desktop on Linux that will have resonance with users, so I would even classify that as a success. I am a pretty generous person though, so I think the harder question is, what classifies an unsuccessful opens source project. And that would be one where there wasn't wide need for the aspect and so it isn't really well used, and so you can spend a

lot of time investing in open source mechanisms when really a sort of smaller private proprietary thing were necessary. That said, once its done, you can release the source to the open, so I guess it also depends on how you define an open project, but at the core I would say that any open source project that ends up getting reused is successful, either by users or other developers.

OR: Have you worked on any commercial software projects?

MB: Yep I used to – before I came to Mozilla, I worked for IBM Canada, and I started working on VisualAge for Java 3.5 point something or other Enterprise Java Toolkit and I was eventually shifted to doing user interface evaluation and design work on what became Eclipse 1.0. So that was a sort of different time for IBM where they went from a commercial package to an open source package. And I think Eclipse has been very successful. I'd also like to amend my previous answer to say, an open source project is successful once it starts moving in directions that its founders couldn't predict. And so Eclipse has done that, right so, I don't think IBM ever would have predicted that BEA would become a member of the Eclipse board, or even that all these other vendors would sort of jump on board of Eclipse and say, you know what, a sort of open source IDE for developers that allows them to just get used to a development environment then plug in the tools they need to develop their software, it's a really good idea. And Eclipse has started to move in lots of ways that I don't think IBM ever predicted that it would move but that is a really good marker of success of an open source project because it means that all of a sudden it is growing organically instead of just being a mechanism through which you can get people to contribute to your code. How socialist of me.

Commercial packages, so after I worked on Eclipse I started working on WebSphere Integration Developer -- currently available in 6.0, contact your IBM salesman now -- and it was based on an open source product, right so it was an Eclipse based tool, but it itself was proprietary and closed, and so have worked on sort of more traditional software development models. And I am not sure that they necessarily moved any faster than open source mechanisms, which is often a critic of open source, that it doesn't move fast enough. And from what I've seen, open source moves at least as fast as the projects that I had been working on that were

closed source before. There's perhaps more chaos around the control of the product, and I think due to the way -- and my experience again is with enterprise software so due to the way certain types of software like enterprise software marketed -- that degree of control is actually the bigger blocker, right. So, when you are selling enterprise software there is a list of things which you need to have in the box, as decided by a CIO of some company whether or not they need it that seems to actually be a totally different distinction, just a matter of they want these check boxes to be filled. And, in a lot of cases, the people working on the software, don't understand why a lot of the check boxes need to be filled, don't understand how it is useful to users, they just know that they need to do it in order to sell the software. You don't get as much of that in open source. And when you do get that in open source, those are the hardest things to get it done, because nobody really wants to work on them, and that's where…

OR: And where are the instances where that does happen in open source?

MB: In open source, there's certain instances around, an example of something which -- I shouldn't say nobody, because we've got some really good people working on it, but accessibility, is a great example of the type of work that a lot of developers just don't want to do. And it's not a matter of them being horribly bigoted or mean people, it's a matter of them not finding that work personally exciting. But then you get people like Mark Pilgrim and Aaron Leventhal for whom that sort of work is extraordinarily exciting, and they both happen to work for IBM and IBM happens to pay them to work on our code base, but they have sort of taken on that little direction and they whip people into making sure that our code is accessible. And without them we would not have anywhere near the value proposition we do, because our code would not be accessible and US government, for example, wouldn't be able to use our code. So, that is an example of a certain thing which we have to have but, we really need to have dedicated people towards doing it, and I could see that without the contributions of Mark Pilgrim and Aaron Leventhal that would be a harder sell to get somebody involved and working on it.

OR: So in what other ways would you say, either in management or kind of in day-to-day activities, does the proprietary software differ from open source?

MB: There's way more management in proprietary software and that has its benefits and its detrements. So with Firefox 2, we tried to start reinjecting some classical product management into the process and we hit a whole brick wall of resistance. A great example was at one point we named something the product requirements document. We tried -- we didn't use a typical PRD template for it -- we tried to make it more like the Mozilla way of doing things. But listing our requirements and prioritizing them and then trying to explain where these priorities came from and have people help us set those priorities. We weren't doing this in a closed off fashion, we were doing it within the eyes of the community, but people really rejected the name product requirements documents. They just they wanted it to be called, just here are the requirements, here are priorities, de-emphasize the product aspect of it. Similarly, within each sort of focal area, we tried to sort of attach a, here is a development lead, and here is a product lead who will help you get the research and data you need to make the right development decisions, and that came with a lot of rejection and anger as well. Going back to the, we have module owners why do we need product mangers as well? And I think there's some benefits that product management brings into things. So product management can take a lot of lifting off a developer and let a developer focus on what they enjoy and what their preferences are for working while taking on product management tasks which are equally important, such as doing competitive analysis, going out, and finding out legal requirements, talking with partners, finding out where we can interact and what sort of concerns we should be taking as requirements that are non-technical requirements. But figuring out how to make that work while not slowing up our developers and while not blocking on that, is something which we continue to try and evolve and which we really need to figure out. Another, and it's a major difference between proprietary software and open source software. In proprietary software…

MB: Alright, so small interuption there, we were talking about the differences between proprietary and open source models. So what I was saying was the one of the things that a proprietary model is you have a manager who tells you as a developer what to do. And

again benefits and advantages: benefits is, you know what you are responsible for, you know what your deadlines are, you know what you are supposed to do. Disadvantages are that you might not be personally passionate about doing that. The open source model is far more self-directed, where you understand there are requirements and you sign up for doing one of those things, and then you take on that responsibility. There is a certain degree of personal choice out of there, and there is a certain degree of casting about, wondering well who am I, what am I responsible for, what's my role in the organization; I have been guilty of belly aching about that more than once. But one of the things that is nice is that once you've signed up for something, you get this sort of feeling of personal responsibility for it. And as a result I think that people end up being a lot more passionate about what it is they are doing then just simply, I have had these ten bugs assigned to me and so I must do them, for this is how I earn my paycheck. I don't know many people who work for Mozilla, or who work for a company that lets them work for Mozilla, who define themselves by this is what I do to earn a paycheck. A lot of us tend to work on Mozilla because we want to work on Mozilla, and it's kind of bonus that we are getting paid for it. I know that I was describing to a friend of mine that one of the differences for me between working on Mozilla and working on IBM is that IBM was a job and I was very passionate about my job, I really enjoyed my job, and I enjoyed the people I worked with, but when I came home I wasn't thinking about what I was doing in my job. Whereas now in Mozilla when I'm not doing anything else I'm generally doing stuff for Mozilla. And it's not because I am overworked or because I have too many -- or because I feel like I'm always behind, it's because I always want to make sure that I'm moving that project forward.

OR: Do you consider open source software projects – and Mozilla since that's what you work on – as a public service? Have you ever thought about it that way?

MB: I consider any software development a public service. And ours has to, ours has the benefit of being free. I don't think that we are a service for which the public should be charged, necessarily. I don't see this as a service which the public has to have either, right, so the public never asked for us to give him the service. We have decided to make this service as public as possible because we see benefits out of that, so it's not a public service like I'd see sewage

or roads as a public service. It's a public service in that we really, that we are serving the public and we feel like we are doing this for the greater good of the Internet public. Trying to wrap this around a pithy thought but I don't think I considered it before, no. There is obviously a public benefit and a public good to open source projects but I think it would be dangerous for open source to start seeing itself as a public service. Simply because I think that they might feel like that without us, there is no other way of doing things. And what really open source is trying to do is demonstrate that without commercial software services, there is another way of doing things. So, the other model is well-established and has created fortunes and has also created a lot of progress, what we are trying to prove is that another model can work just as well.

OR: And what do you think, the success of Firefox might do for sort of open source as a whole?

MB: To be seen, I think that for open source as a whole, what it will do is it will prove that there are ways of mixing the models more successfully. So I think up until Firefox the model had been -- and I would even say up until arguably Mozilla 1.0, I think we shouldn't differentiate Mozilla 1.0 with Firefox in terms of the impact on open source projects. I think that Firefox ended up getting more users because it paid attention to user requirements a bit more than Mozilla, but I think that Mozilla was still a very different way of doing an open source project, and that it was a user-facing project, whereas most open source project have not been, historically. And I think that Mozilla really sort of took into account the idea of making itself relevant to the user, instead of relevant to a specific class of users, they tried to be relevant to the global user and they just, I think that with Mozilla 1.0, they just missed the market a little and then they refined it for Firefox. I think what it's going to prove is that mixing an open source model with a consumer application model can end up being very successful, and figuring out how to make -- how to return investment back into the community through revenue share or other mechanisms is a solvable problem. Getting other companies to participate and proving the value to those other companies and having them participate is a solved problem, now even. And I think that will have a great effect on the open source community, and I think you've seen that as well. I think that, you wouldn't see as

much industrial focus on open source if it weren't for the success of Mozilla 1.0, of projects like Apache, or projects even like Linux.

OR: And do you sort of think that open source techniques can be applied to other means of production like you mentioned marketing, do you see examples of open source methods in other areas in society, and do you – if you do, do you think they are all kind of part of the same movement that's happening right now?

MB: I think so, yeah, so I'm going to even step out of software altogether and talk with public policy. A friend of mine, Dave Eaves who is a bit of a policy wonk, he's a mediator and negotiator and potentially a future Canadian Prime Minister, if he has his way. He has become very excited about open source since I started telling him about what I was doing. And he wants to see policy documents written in an open source sort of fashion. And again, I think that a lot of people mistake open source with everybody has an equal voice, and that's not what it means. What open source means is you are listening to the input of others and you are not making a presumption that the only people who have valuable input are the people who work for you or the people you work with, but rather that a very powerful and good opinion can come out of the strangest of places, and the only way that you can make sure you are actually getting access to all of that information and knowledge is by opening up the process. And that the benefits you have from doing things in a closed environment are outweighed by the benefits you have of just some random guy who is interested in what you are interested in and happens to be killer smart and just happens to be in different place than you, can give you his knowledge and you can benefit from that knowledge.

Do I see this as sort of a shifting attitude in all sorts of industry? Maybe. I think certain -- I think the sort of variable in that equation that needs to be controlled is, to what degree of control do you -- what degree of control do you hand over to the open source audience, where do you put your gatekeepers, and how do you respond to that audience to whom you say you are listening. I think the whole idea of participatory design is becoming bigger and bigger and you're seeing that more in marketing, you're seeing more where people realize that the best way to market a product is not to just tell somebody this is why Tim Horton's coffee is great, but rather to make Tim Horton's part of their day and make them

part of Tim Horton's and have Tim Horton's listen to what they want, and respond to those user needs. And I think that where technology is landing us is a point where the exchange of information is now so cheap and so facile that you can start doing these things more easily and you can as, again, as Tim Horton, or as a politician you can put up, you know, here is the bill, or here's the new product that I am looking to give out, what do you guys think of it. And you can get people telling you their opinion before they even try it. And how much you listen to that is obviously your concern, but that sort of involvement by the marketplace and the development of things which are eventually destined for them, is just an obvious benefit to me, and all that's different now is that we are coming up with ways of making it happen.

OR: So, last question, and is this extremely broad…

MB: Because I have been answering so narrowly to date.

OR: What do you think is the future of Open Source?

MB: Flying cars…ah…future of open source, boy, I don't know. It's really, really hard for me to predict that. Only because to me it's so new and wowy and amazing. I will be very interested to hear the opinions of a lot of my colleagues on this. I think the future of open source is that we are going to see more and more requirements for platforms to end up being open "source-ish". And we're going to see – and the requirements are not going to come from government, they aren't going to come from any sort of regulator, it's going to come from the marketplace. Using an example that I just read in the book, which I would quote if I remember the title, so I'm just going to state this now, this is not my terminology or idea, creative commons by attribution maybe, who knows. It used to be that you could do Trojan horse platform plays where you, you know, you give somebody what seems like a great gift and it turns out that you have a whole bunch of guise inside which make sure that you lock somebody into their platform.

But nowadays people have seen that so much, and they are so wary of it, and they are so wary of creating another IBM circa 1980 or Microsoft circa 1990 – and I think Microsoft is changing, which is why I said Microsoft circa 1990. But they are so worried of creating these sort of proprietary platforms that the market is

extraordinarily wary of it. And I saw this when I was working on Enterprise software -- they didn't want to locked in. So standards-base was really important to them, so that if they want to change their vendor and take their data with them, they could. I see a lot of industry trying to rebel against this, specifically the media industry. They're really trying to rebel against that idea of the users data is the users, but I think that's going to be where open source really starts to gain traction in offering this idea of -- by making sure the data formats aren't proprietary and by making sure that the platform isn't proprietary and that somebody can move from vendor to vendor is going to become more and more of a market requirement. And so that's where you are going to see, from an industry side, a lot more investment in open source, because you buy interoperability through open source. Not necessarily easy interoperability, don't get me wrong there because I as a proprietary vendor could open source my data format and then do say, okay anybody else can deal with that and I am not going to make it easy for them to. But at least that would be there, so you can start that interoperability.

And then from there I kind of see open source -- I think Dave is right, I think you are going to see a lot more open source in public policy, and in non-software aspects. You can see a lot more sort of people being able to file bugs on products, like Tim Horton's coffee or their BlackBerry, simply because I think more and more industry will realize that their most passionate resource are the people who like their products, and want to make their products better. And it would be foolish for them not to listen to those voices, because those are the people who are going to then buy those better products, and its like a sort of a guaranteed feedback loop. And I don't know how many eons we need of business moguls rewriting the same maxim, listen to your customers and give them what they want, but it seems like every business book is basically focused around that principle. And then they talk about different ways of listening and different ways of giving, and I think open source is just a very powerful way of listening to your customers, and a very powerful way of helping to give them what they want. How's that for prophecy?

OR: Great, thanks very much, Mike.

MB: You are quite welcome.