EPISODE 815

[INTRODUCTION]

[00:00:00] JM: Open source policy has become a business issue as well as a political one. Business is like Elastic, MongoDB, the company, and Redis Labs, have started to view the open source licenses of the projects they work on as a means for business defensibility against cloud providers offering similar services. It remains to be seen how viable this strategy will be for the commercial open source vendors.

Companies that do not directly sell commercial open source are also grappling with questions around open source licensing. Facebook has become a force in the open source world through projects like React and GraphQL. Facebook leads these projects, but Facebook is not monetizing them other than to the extent that they use the projects to build facebook.com. Facebook's incentives are aligned with the rest of the industry on the quality of GraphQL and React. Proper licensing can help Facebook keep those incentives in alignment.

Joel Marcey, Michael Cheng, and Kathy Kam from Facebook join me for a discussion of the state of open source licensing and how that impacts Facebook.

A few brief updates to Software Engineering Daily land. Podsheets is our open source set of tools for managing podcasts and podcast businesses. We have a new version of Software Daily, our app and ad-free subscription service. Actually, the app is getting worked on. The website has been rolled out at softwaredaily.com. We'd love your feedback. We'd love any bugs you're having with the apps. I know the apps have their bugs, but we're working to improve them.

Software Daily is also looking for help with Android engineering, QA, machine learning and more. The links for all of these are in the show notes. And the FindCollabs hackathon has ended. The winners of the FindColalbs hackathon will probably be announced by the time that this episode airs. We will also be announcing our next hackathon in a few weeks. So stay tuned. Again, the links to all of these details, from Podsheets, to FindCollabs, are in the show notes for this episode.

[SPONSOR MESSAGE]

[00:02:18] JM: DigitalOcean is a reliable, easy to use cloud provider. I've used DigitalOcean for years whenever I want to get an application off the ground quickly, and I've always loved the focus on user experience, the great documentation and the simple user interface. More and more people are finding out about DigitalOcean and realizing that DigitalOcean is perfect for their application workloads.

This year, DigitalOcean is making that even easier with new node types. A $15 flexible droplet that can mix and match different configurations of CPU and RAM to get the perfect amount of resources for your application. There are also CPU optimized droplets, perfect for highly active frontend servers or CICD workloads, and running on the cloud can get expensive, which is why DigitalOcean makes it easy to choose the right size instance. The prices on standard instances have gone down too. You can check out all their new deals by going to do.co/sedaily, and as a bonus to our listeners, you will get $100 in credit to use over 60 days. That's a lot of money to experiment with. You can make a hundred dollars go pretty far on DigitalOcean. You can use the credit for hosting, or infrastructure, and that includes load balancers, object storage. DigitalOcean Spaces is a great new product that provides object storage, of course, computation.

Get your free $100 credit at do.co/sedaily, and thanks to DigitalOcean for being a sponsor. The cofounder of DigitalOcean, Moisey Uretsky, was one of the first people I interviewed, and his interview was really inspirational for me. So I've always thought of DigitalOcean as a pretty inspirational company. So thank you, DigitalOcean.

[INTERVIEW]

[00:04:25] JM: Joel Marcey, Michael Cheng, and Kathy Kam, you are all work on open source at Facebook. Welcome to Software Engineering Daily.

[00:04:31] JMarcey: Thanks for having us, Jeff.

[00:04:33] KK: Thank you.

[00:04:33] MC: Thanks, Jeff.

[00:04:34] JM: So to start off, I'd like to set some context for how open source happens at Facebook. Facebook is a unique product. It's a massive social networking company. There's lots of decisions to make around internal tooling, and I think some internal tools that get built turn into open source projects. Other ones do not.

Explain how internal tools get built at Facebook and why some of them evolve into open source projects?

[00:05:05] JMarcey: So I think it's good to go and talk about sort of the history a little bit of how open source came to be at Facebook. So Facebook has always had open source in its culture ever since the beginning. 2004, Zuck's in her dorm room creating the first version of Facebook using LAMP; Linux, Apache, MySQL, PHP. So we've always had that sort of in our blood.

So what I always tell people, like we try to find reasons not to open source something, versus trying to find reasons to open source any project. So generally, if we feel like a project would be beneficial to the community and we feel like it would maybe start a new technology trend or improve existing technology, those are the types of reasons we would give to open source any given project.

So React is an example of something that came out, we've developed internally for a specific reason, but we decided that, "Hey, this is kind of a new framework that actually could shake the industry," and it did.

So we definitely look for reasons to not – We try to find reasons not to open source something and we generally don't come up with a lot of those. So we generally try to open source as many projects as we possibly can.

[00:06:21] JM: And the reason that, or one reason that large tech companies like Facebook would open source something is because if you have something that is critical infrastructure or it's very useful within Facebook and you believe that this could be critical infrastructure for other

companies, then by open sourcing it, you're getting at ahead of the curve and you're saying, "Here's something we use internally, and we're going to kind of manifest this community around this project," and therefore there is a sense of self-interest within it. It's a positive-sum action, but it can gain a lot from Facebook's point of view.

I think it actually makes further sense, because in many ways, Facebook or a company like Facebook, is ahead of the curve in terms of problems that the rest of the industry may see in 5 to 10 years. What are some of the engineering problems that Facebook has been ahead of the curve on maybe seeing things in engineering communications, or continuous delivery, or problems with JavaScript? What kinds of things has Facebook seen before the rest of the industry?

[00:07:33] KK: I think one of the things we see a lot, I mean, at our scale, we have one of the largest kind of mobile app there is. So we think a lot about scaling performance to ensure that the Facebook experience is great. So we actually see a lot of kind of issues in the different platforms, iOS and Android, that we see before most people do. We have learnings and we investigate on how to mitigate some of these, and that's something we would love to share to the broader ecosystem.

[00:08:04] JMarcey: Yeah, and you could imagine like if we – There's probably a bunch of open source projects that we could consider using for many different problems that we have. But a lot of those don't – We find maybe sometimes a lot of those don't scale to what we need as a company with billions of users and people using all of our apps. So we can either like fork those projects and try to work that way, or we can try to come up with ways to determine is there a better way to accomplish something given the scale that we actually need for a business. So it's a lot of – We have a big user base, a lot of apps, and what's out there that can help us accomplish our goals. If there's nothing out there, what can we do to actually create some new technology to help us.

[00:08:47] MC: Yeah. I think we've traditionally seen open source as a way to share our learnings and share our sort of most difficult trials and tribulations that we've had to deal with as a company. So in the beginning stages of our growth, it was scaled as Joel and Kathy mentioned. So sharing our frameworks, sharing how we've solved those problems has – In our

sense, from our perspective, a rising tide lifts all boats, and that has made, to your earlier question, of what we decide to open source and what we don't have to decide open source. We're not looking to sell infrastructure as a stack, as a service. We don't have a public cloud offering. So it's a lot easier.

Those decisions as to – Because we're not looking to make money off of infrastructure in the same way that some other companies might consider. So those decisions are a lot easier for us, right? Then going into the future as our company pivots towards privacy, as our company pivots to solving the more challenging issues of the day, we expect that our engineers will also be asked to create novel solutions to those and hopefully open source could be one of the ways that we share our solutions to some of the more difficult problems that I think everyone in the ecosystem is facing.

[00:10:03] JM: Indeed. It will be interesting to watch the open source projects that come out of that pivot. Kathy, you were working on externally-facing developer tools at Google for almost 5 years before joining Facebook. How does Google's culture around building external tools compare to that of Facebook?

[00:10:23] KK: I think the two company is very similar in many ways. They kind of have top engineers working on difficult problems around scale, and open source is kind of equally part of the DNA of both companies. So coming here, I feel right at home and really enjoyed the good collaboration here at Facebook.

[00:10:42] MC: Michael, you work on open source licensing policy at Facebook, which is not exactly what is a conversation I've had on Software Engineering Daily in detail. So maybe we could do another show in the future. It'd be fun to go deep on that subject. Describe what you do on an average day.

[00:11:03] MC: Yeah. So I think there're a couple of big buckets. The first big bucket is making sure our engineers know how to use open source for internal development of our projects and platforms. So that's basically the entire universe of knowledge with respect to that bucket is, "Am I TBSD okay?" and you have to think a little harder about some of your license. Apache is probably okay. For internal development, a lot of even copyleft licenses are probably okay given

that we don't ship as much software, like in a distributed format, like in app, or we don't have as many apps and don't have as many like PC or Mac applications as some other companies do. So that's the first bucket, making sure we're using open source correctly and properly in a way that complies with the license requirements and also helps us move fast.

The second bucket has to do with helping Kathy and Joel in the program alpha side. So open sourcing anything is kind of like giving something away. It's kind of like giving a piece of property away, and it just requires figuring out from a consensus perspective, making sure that everyone is aligned, or that this is the sort of right thing to do. So a lot of times I will help Kathy and Joel figure out what kind of other sort of legal issues might come about as a result of that, and also sort of making sure that all of the stakeholders are looped in. So that's probably the second big bucket.

The third kind of thing that my team also works on is we also work with other companies to ensure that all of us, we're all thinking about open source. We're all thinking about open source in a sort of thoughtful and responsible manner from the perspective of the community and from the perspective of collective action and making sure that we're all doing the right things by the community.

So one of the things that that involves is often kind of speaking out and taking positions on controversial topics that have legal implications. I spoke at FOSDEM earlier this year about some of the strange licenses that have come out, some of the controversies around Mongo and other companies relicensing what are originally open source projects to other licenses that people haven't seen before. So I think, increasingly, we are looking to become more active in the community and making sure that we are doing our part to ensure that the ecosystem is safe and that the values and the principles of open source continue to be reflected in the day-to-day actions of what people are doing.

can trust. Whether you are a new company building your first product, like me, or an established company that wants additional engineering help, G2i has the talent that you need to accomplish your goals.

Go to softwareengineeringdaily.com/g2i to learn more about what G2i has to offer. We've also done several shows with the people who run G2i, Gabe Greenberg, and the rest of his team. These are engineers who know about the React ecosystem, about the mobile ecosystem, about GraphQL, React Native. They know their stuff and they run a great organization.

In my personal experience, G2i has linked me up with experienced engineers that can fit my budget, and the G2i staff are friendly and easy to work with. They know how product development works. They can help you find the perfect engineer for your stack, and you can go to softwareengineeringdaily.com/g2i to learn more about G2i.

Thank you to G2i for being a great supporter of Software Engineering Daily both as listeners and also as people who have contributed code that have helped me out in my projects. So if you want to get some additional help for your engineering projects, go to softwareengineeringdaily.com/g2i.

[INTERVIEW CONTINUED]

[00:15:53] JM: Let's talk about that in a little more detail. The licensing changes that some companies have made as a defense to cloud providers productizing the same open source project that a company like a MongoDB or an Elastic is based on. So this is kind of an interesting topic, because it comes down to the roots of open source as well as the sustainability of open source. Can you just give me a lay of the land for how you see the changes to the licensing world in open source today and maybe your subjective views on what is right and what is wrong?

[00:16:36] MC: Yeah, that's a great question, Jeff. I would start off by saying that I actually don't like to start off the conversation with this idea of right or wrong. I think open source means very, very different things to all of us. What I've seen folks do, especially around the discussion of

MongoDB's license, which is called the SSPL, is they really gone off into their corners and started like throwing sort of [inaudible 00:17:02] at each other.

If open source means different things to all of us based on our personal narratives, then it's really difficult for us to agree on what the definition of open source is, and I think, ultimately at the end of the day, everyone who's having this debate no matter how sort of frustrated they are with one another, we have more in common with one another than with somebody who just wants to keep all software closed and doesn't want to give any away. Those people I think are very, very different from us. I think it's important to start off by reminding ourselves that we started off on this path together.

But that being said, I think there's a question – So this all started with the rise of open core, which is this idea that part of the software will be closed sourced and part of the software will be open source. The part of the software that is open source that the companies will use to get adaption, to get users, to get contributions, and then companies will then make money off of the part that's closed source.

How I see the problem, and these are my own views and of course not those of my employer or the company, is that when folks started out this business model, they made a decision as to what proportion of the stack they want to make open source and what proportion of the stack they want to make close source, and they thought that that proportion would lead to dollar signs, or lead to like a successful revenue model.

That may not necessarily – As industries change, as industries are disrupted, that equation actually changes overtime. So I think that Mongo and others have figured out is that they actually want to – The things that they thought they wanted to open source, they may not, and they will actually want to close source that.

So taking that to its natural conclusion, I think at the end of the day, a lot of this relicensing sort of controversy has to do with folks making an initial business decision about what they think will make them money and then deciding to change their mind and then changing the license. I think that's what it ultimately comes down to.

When you look at Mongo – So Mongo specifically, they blamed big cloud providers for their current predicament, and big cloud providers who take their code offered as a service deep within their stack and then don't contribute anything back in return. If you actually go and take a look at the top committers and the top contributors to the Mongo stack on GitHub, almost all of them are Mongo employees.

Mongo actually doesn't want – My sense is that Mongo doesn't necessarily want that many contributions, because it wants to be able to pick which features go in the close stack and which features go in the open stack. So I'm not sure how much cloud providers are actually to blame for wrong doing or if they have done anything wrong at all. I think it really does just come down to changing dynamics, changing landscapes with the rise of public cloud and the ability or inability to adapt to it.

[00:20:05] JM: It's such a great summary of the landscape right now. I guess when I said right or wrong, I share your lack of wanting to deliver a value judgment over this in terms of a morality kind of thing, because I don't really think that's the terms that we need to discuss this in. I think even from a strategic point of view, this is kind of a dubious decision. Joel and Kathy really like your perspective on this, because the way I see it, as somebody who has developed software, and I haven't spent that much time in the software industry. Well, I guess, five or six years. That's probably more than a lot of people listening, but less than a lot of people listening as well. I have a sense of the ethos of open source, and this doesn't really jive with me, if you will. It's almost like kind of a religious sensibility that you develop for what open source is even just from interacting with open source software.

The thing is like what I suspect is that a lot of people who join these companies who are the best engineers at places like Elastic or Mongo, they also have a certain sensibility for what open source is. When somebody, a higher up, changes the strategy, and therefore changes the license, and therefore kind of dilutes the purity of the open source branding of the company, is really dangerous for your kind of engineering, I guess, excitement for working at that company.

I don't know. What's your perspective, Joel and Kathy?

[00:21:39] JMarcey: So I kind of look at this maybe at a more idealistic or simplistic view than maybe Michael does or some other folks, is that folks that come to Facebook or any other big company, that they just want to work on open source. They want to share their contributions to the world and they really don't want licenses to get in the way of that.

So I feel like sometimes that these are arguments are – I don't know, maybe it's the wrong word. Maybe more theoretical than like practical for the folks that are actually doing the work of contributing to these projects.

I know at Facebook, we're trying to be more cognizant on the type of licenses that we put on our projects, or an example we know, like MIT and BSD are very recognized and popularized to use across all open source projects, and I think that a really good way to go. I don't think that engineers in general want to get into the minutia of worrying about, "Can someone use my code legally or not?" They just want people to actually use it, or, "Can I use someone else's piece of code?" or "Can I get inspired by their code to help my project out?" I think it's more like they want to have that sort of sharing mechanism across the projects. So engineers I think definitely look at it a lot in different ways than maybe folks in the legal department at various companies.

[00:23:06] KK: And I think this is where our team comes in to play. We are kind of the interface between engineers and the legal department to help guide and provide guidance. Like Joel said, the engineers just want to collaborate and get the work done. They honestly don't know all the details until it's bad for them. Until it blocks them from being able to do something, they don't really care or even really want to know the details.

So the key here is that having a team, like our team, which is the open source policy office team, where we handhold and guide these projects and making sure that their goals is met with the correct legal kind of contract and then ensuring everyone can collaborate and just focus on the technology.

[00:23:49] JMarcey: I want to just add too, since Michael is here, he's been very invaluable in trying to change that type of culture that exist at the company about allowing engineers to have the freedom to contribute to open source, to bring in open source and those sorts of things. So I

feel like at our company in particular, we're moving definitely in the right direction when it comes to how engineers perceive open source.

[00:24:18] MC: Thank you, Joel.

[00:24:19] JMarcey: Yeah, sure.

[00:24:20] JM: I'd really like to do a show at some point about kind of the strategic wisdom of these decision, because the way – I've talked to some people who have knowledge of these companies, and I think they really just feel boxed in by the cloud providers. I'm just surprised at that, because – Come on! You're an innovate software company. Can't you figure out an innovative piece of software to build to get you out of this mess rather than change the license? It's just kind of – I don't know. But that's for a different show.

[00:24:54] MC: I can say, Jeff, that – I can't give you too many details about this, but there are discussions with the cloud providers going on right now. There are projects spinning up where, let's just say for Mongo specifically, the cloud providers – Again, I don't know – I'm not sure if there's sufficient leverage enough to get them to do this, but the plan is to have them expose their Mongo version of the service to Mongo to allow Mongo to extract revenue from within the cloud stack.

[00:25:27] JM: That's kind of a cool compromise.

[00:25:28] MC: Yeah. I think that's the most logical and natural compromise that we think, but it's kind of frustrated by, like we talked about, these initial claims, the initial missiles that these companies threw at public cloud providers, which frankly as I talked to before, I'm not sure were entirely fair, right? So that kind of muddies the water a little bit when you throw these claims of unfairness and all these stuff around, right?

[00:25:54] JM: Oh, yeah.

[00:25:56] MC: And to bring them on to the table, that's not like a great place to start a conversation. Of course, as a practical matter, what leverage do these companies have at this

point? So it's market disruption. It's a lot of market disruption that's happening right now, and I'm hoping that we can come to some kind of solution that doesn't involve everyone going proprietary, right?

[00:26:14] JM: Yeah. Just to the point of – I really shouldn't go down the rabbit hole on this one, because I want to talk about Facebook. But just to the point of what leverage do they have. I mean, what leverage did Amazon have when they started AWS? It's like a new – It was a totally new innovative business that Amazon stood up essentially from scratch and has provided massive value to the entire world. Not just the software world. The entire world has benefited from AWS. Some of these open source businesses, these commercial open source businesses, have the gull to basically assert a moral superiority over big, bad Amazon. It's just kind of a – This is not a good look.

Okay. Anyway, Facebook. So one question I have, one fundamental question I have around a company like Facebook, or this also extends to a company like Netflix. Basically, Facebook or Netflix, their market dominance is not necessarily due to like the software that they're running. Facebook has a really great newsfeed product. It has a really great video playing product, so does Netflix, etc.

But, really, the value of the company is kind of in the operational cadence and the network effects, etc. Kind of it's in the database. The value is in the database, rather than in the code base. I mean, arguably, you could say there's even more of a moat by virtue of the codebase, but you can make the equivalent argument that maybe we should open source everything except for the database layer, like the actual contents of the database.

To what extent – You have that argument on one end that's kind of like the Linux argument, versus kind of the polar opposite end, which is like – I don't know, maybe the Oracle argument. Today, Facebook sits somewhere in the middle. What's the natural – I guess, how extreme can you get in terms of your open source of Facebook?

[00:28:25] JMarcey: If I understand your question, I actually think Facebook might be towards that initial end you were talking about, like why don't you just sort of open source everything but the database? Obviously, we're not open sourcing data, user data, like that obviously. There's

certain key elements that every business has that they just can't open source, because it's just business proprietary things.

But I think in general, Facebook as a company, we do open source a lot of things from the top of the stack to the bottom of the stack. We open source database like RocksDB, MySQL. I mean, parts of MySQL, those sorts of things. We obviously open source UI, at the UI end. So I feel like we're at the point of not in the middle. We're more towards the end that you were speaking about why not open source everything. We don't do everything, but we do most things. I think we're going to continue to do that obviously with various caveats of things that we just – Or just aren't able to open source.

[00:29:35] MC: I think that's right, Joel. I think we probably open source as much as we can and as much as we feel. I think the bandwidth – So open source also, as you know, it's not something we just throw over the wall and forget about. Open source is also something we have to maintain, we have to sort of actively sort of nurture the community for and things like that. So I think when we think about open source, like there are certain things that – There are projects that have come across our desk which we've liked like, "Yeah, we could open source this, but is it really that interesting enough for the community to see all that? Do we also want to commit? If it's not that interesting to the community, do we want to commit manpower to it?"

[00:30:20] KK: I think this kind of highlights the two different types of open source we do at Facebook, and Joel and I have talked a lot about this. One type is where we are open sourcing as a reference so people can see the code. We do a lot of that in our AI and ML work where we have some papers and we really want people to see the code. But we're not necessarily interested in a community behind some of these smaller open source project.

Then there are the big ones. The big ones like Buck, and PyTorch, and React Native, and React, that we really want to build a community around and we want to put the resources in. So when an open source project come through our desk, that's one of the first question we ask, is what are your goals in open sourcing and whether we have the kind of manpower to achieve this goal. If we don't, how do we mitigate it or whether open sourcing is really the right thing to do.

[00:31:13] JM: Yeah, this is alluding to the fact that close source software is sometimes not just about, “Oh, this is my proprietary mote.” It’s sometimes about, “Look, we want to build software at a reasonable pace, and if we open source everything, there’s going to be so many security vulnerabilities that are going to be exposed. There’s like backdoors,” and like things that people would just not even think of if – Vulnerabilities would not even think of unless the project was open source.

Also, it’s sometimes to the point of how much manpower of human power, I should say, do we need to put behind this open source project. You think about something like Google Borg. I don’t know the exact details of why they didn’t open source Borg, but it had something to do with the fact that Borg was so tightly coupled to internal Google systems, that if they were to just open source Borg, it’s kind of like – To make that actually useful, they would have to open source everything in Google, because it’s so tightly coupled to everything.

[00:32:22] JMarcey: Yeah. We actually run into some other issues with things that we open source. For example, like Kathy mentioned, Buck, right? We open sourced Buck as a build tool that we think the community would be able to benefit from, but we also have internal uses of Buck that would not make any sense to open source. The community wouldn’t get any value from it.

So it’s a game of open source what you can and don’t open source what doesn’t need to be open sourced. But in general, if we feel that a technology or a project will have some value to the community, that we will try to open source all of it or as much of it as possible so that the community could benefit from it, but then doesn’t get confused as to why like some things aren’t working like they might work inside the company, because we have certain systems or internal uses of a project that can’t be utilized externally.

[SPONSOR MESSAGE]

[00:33:30] JM: Azure Container Service simplifies the deployment, management and operations of Kubernetes. Eliminate the complicated planning and deployment of fully-orchestrated containerized applications with Kubernetes. You can quickly provision clusters to be up and running in no time while simplifying your monitoring and cluster management through auto

upgrades and a built-in operations console. Avoid being locked into any one vendor or resource. You can continue to work with the tools that you already know, such as Helm and move applications to any Kubernetes deployment.

Integrate with your choice of container registry, including Azure container registry. Also, quickly and efficiently scale to maximize your resource utilization without having to take your applications offline. Isolate your application from infrastructure failures and transparently scale the underlying infrastructure to meet growing demands, all while increasing the security, reliability and availability of critical business workloads with Azure.

To learn more about Azure Container Service and other Azure services as well as receive a free e-book by Brendan Burns, go to aka.ms/sedaily. Brendan Burns is the creator of Kubernetes and his e-book is about some of the distributed systems design lessons that he has learned building Kubernetes. That e-book is available at aka.ms/sedaily.

[INTERVIEW CONTINUED]

[00:35:06] JM: What else goes into the process of making a piece of software at Facebook that's kind of internally coupled? What goes into the process of baking that well enough to make it exposed to the external community?

[00:35:23] JMarcey: Right. So most of the projects that start out at Facebook start out from a need that we have internally. Obviously, we want to make sure that the stuff that we're – The apps and the projects that we're shipping are being used – Have the appropriate quality and use from the projects that we have that they ship well. But I think we try to design our projects in such a way that we try to reduce the coupling as much as possible through various mechanisms on how we ship our code externally, through how we actually design configuration files and those sorts of things so that we can easily ship the code out and it could be used by the external community, but we also have ways internally to actually utilize it for optimizations that we need for our apps and those sorts of things.

[00:36:19] JM: Not to shift us back to the subject of controversy, but I do want to talk a little bit about this thing in 2017 with WordPress. So there was this instance in 2017 where WordPress

was evaluating what to use for its big refactor, and it was evaluating React JS, and then they were planning not to use it because of the patent risk. Then Facebook actually changed the license because of this kind of debate that rose up. Then WordPress actually changed course and used React, and I thought that was kind of a pivotal moment in Facebook's trajectory as a company moving towards open source. Can you describe what happened here?

[00:37:01] MC: Yeah. So I think there was – And this goes back to the point of the core of why licensing decisions really go back and what license to use and things like that really go back to the core of why companies do open source. So Facebook to a large extent does open source to sort of show leadership. As Joel first mentioned, because we feel like a lot of our additional technologies were built on the LAMP stack and we feel we have a debt to repay, and we want to do what's right by the community. Second of all, to also show leadership in that community.

So our goals are very community-driven, and there are many more degrees of separation between those community-driven goals and our revenue goals. So we don't think about it in terms of revenue. So we think about open source. We care a lot more about what the community wants or what the community thinks and what the community believes that a lot of other companies out there who have to weigh that against their bottom line at the end of each quarter, at the end of each half, at the end of each year.

So that's I think the preface, right? We came up with a license that we thought was actually beneficial without getting too much into the weeds. We thought was beneficial to the ecosystem. The community didn't agree. In the end, as we just mentioned, because we care a lot about what the community thinks, it didn't matter what we thought. We were going to relicense it, because the community – It started to become a blocker to adaption, and the whole reason we do open source is we try to sort of create – We try to reduce blockers, not increase them.

So the uproar in the community got to a point where it started to get in the way of our central reason why before doing open source, which is why relicensing was a no brainer from our perspective. Since that time, we've also relicensed probably close to relicensing almost all of the projects away from BSD+ patents because, again, we're optimizing for sort of reducing blockers and not increasing them. In our opinion, we thought that that's what we were doing with the

BSD+ patents license. But again, it doesn't – At the end of the day it doesn't really matter. We're here to work with a community, and that's what required.

[00:39:17] JM: Kathy, some of your work at Google was on Android. If I recall correctly, Android was kind of a central component that is affected or potentially affected by this never ending court battle between Google and Oracle. Was your perspective on software licensing and open source, was that affected by your experience, I guess, witnessing that court case, or did you just not pay attention to it at all?

[00:39:49] KK: Frankly, I was on the engineering side and I did not pay as much attention to it. As I mentioned before –

[00:39:54] JM: That makes the two of us – Have never gotten deep on that.

[00:39:55] KK: As I mentioned, engineers just want to get kind of engineering work done and build the best kind of experience, and I kind of left that to my lawyer counterparts.

[00:40:06] MC: Yeah. I mean, me and my friends, lawyer nerds, we did show up at the trial and we did show at the trial to kind of heckle one side. I'm sure you can guess which side.

[00:40:17] JM: Yeah. I mean – Okay. I guess we don't have to go down that rabbit hole. I don't know. Michael, do you think that's – Has that had an impact on how the world has seen open source or how the software world has shaped its opinions of open source?

[00:40:31] MC: The outcomes and the implications of the case are extraordinarily frightening to me, and it is – Because it's such a long running saga, it's not something that I'm worried about on a day-to-day basis, but it is a cloud that hangs over open source. So to your question, I don't think it's affected, because it's been there for so long. I don't think it actively affects what we do or how the ecosystem works. But in terms of worrying about on the periphery, on the edges, when we're talking about corner cases about what kind of bad actors could out of the woodwork. This is definitely top on our list and making sure that in wondering whether the ecosystem is as safe as it could be.

It's things like this that keep the more conservative companies from fully embracing open source, because they're worried about like, "Oh! What about this or what about that?" and that's the biggest – A lot of the biggest challenges that I have when I'm talking to my counterparts at other companies is trying to put their fears at rest, and it's a little bit harder to do that when you have scarier things like this that are out there.

[00:41:40] JM: I mean, what are some other cases? I can think of like open SSL, maybe. I mean, that was kind of a frightening moment.

[00:41:47] MC: You mean from a security vulnerability perspective.

[00:41:49] JM: I guess that's a poorly formed question. Yeah, from security vulnerability perspective or like more broadly, like critiques of the open source world, or the vulnerabilities of the open source world, or what keeps you up at night to the extent that open source software policy keeps you up at night.

[00:42:06] MC: I think there's so little case law that – I'm actually heartened by how little case law there is, because if you think about someone having to go and litigate at open source – One of the core questions of open source from scratch. That's a very, very expensive case, because somebody would have to go get experts on weighing on all of these fundamental questions that no one has thought about and that are not actually enshrined in law anywhere. They're mostly reflected in norms of day-to-day practice.

So the fact that there's so little law kind of – On one hand it contributes the uncertainty, but on the other hand it also contributes to the certainty, because maybe no one wants – People are so afraid of litigating it, that they just want to leave it alone.

[00:42:52] JM: A few final questions. This is kind of a far-flung question. I don't know how much either of you have thought about this. I'm heartened by the Facebook leadership's seriousness in which it approaches the future of cryptocurrencies and opportunities to use blockchain technology, because I obviously see a lot of opportunities, and I don't know if either of you have listened to the Jack Dorsey podcast tour that he's been going on. He's very optimistic about how crypto could affect social networking. I think it could be quite a good solution to a lot of the

problems both in social networks and potentially in open source. Do either of you have any speculations or visions for how cryptocurrency could affect open source policy?

[00:43:38] KK: I think this is a relatively new space, and that we're still looking to it. There's a lot of opportunity, as you have mentioned, but there's also other considerations to think about. So I don't think there's really – The industry is still developing.

[00:43:53] JMarcey: Yeah, and just to give you an inside view, it's a pretty secretive project. So we don't have a lot of firsthand information to share with you about that.

[00:44:05] MC: That's exactly what I was going to say.

[00:44:07] JM: No problem. I mean, ideas. Just general ideas, like let's programmatically incentive people to work on open source software, for example, through some sort of currency.

[00:44:20] MC: Yeah, it's definitely possible. I think we've had a lot of discussions just like on a Friday night and a beer about all of the possibilities. I think –

[00:44:28] KK: Yeah, I don't think this is more than speculation on our part.

[00:44:31] JM: Totally.

[00:44:31] MC: Yeah, we're just speculating. Yeah.

[00:44:34] JM: Totally. Okay. I'll try to get you to speculate more in the future, because I think it's an interesting area of speculation. I agree with you. That's all we're at today as far as speculation.

[00:44:45] JMarcey: Yeah, one thing I'll just say to close that out, is that I think blockchain in general has a lot more application than just cryptocurrency. I think there're possibilities of entire industries that could be affected by it.

[00:44:58] JM: Yeah. I completely agree. Facebook recently started a podcast around Facebook open source. Joel, tell me about the Facebook open source podcast and what are your goals with it.

[00:45:11] JMarcey: Yeah. Thanks, Jeff. We just launched, I think, March 11 or about three, three and a half weeks ago. Basically, it's a podcast from the Facebook open source program office, where we talk about the people, the projects and the community around them of just how Facebook open source works.

So we launched our first four episodes talking about things from React, React Native and how we manage their community, GraphQL and our move to the Linux Foundation there. How we actually run our open source program office in general. There's also a podcast around a language we developed called Reason as well.

So you can learn about a lot of these projects. You can obviously subscribe at iTunes or Stitcher or Google Podcast, or you can go to the thediffpodcas.com and listen there and subscribe.

[00:46:03] JM: Awesome. Well, any closing thoughts for the listeners from either of you about open source software as it relates to Facebook?

[00:46:10] KK: I guess just a call to action and for folks who kind of review our open source projects. Take a look. We love to grow our community and love community contribution. So any feedback on that kind of area, we'd love to hear it directly from you. You can reach us via our Twitter at –

[00:46:28] JMarcey: @fbopensource.

[00:46:29] KK: As well as connecting to the projects directly in GitHub.

[00:46:33] JMarcey: Yeah, I'd like to add too. So my role at Facebook is what they call a developer advocate. That's similar to like developer evangelist, or different titles. So I see my role as basically being a storyteller, and I really want to work hard to like to try to come up with cohesive narratives around all the projects that we launch. As a developer advocate, my role is

to actually listen to the needs and wants of what the community is looking for in how they do day-to-day development.

If I can recommend Facebook open source projects to help them, that's great. If I can't, then I might recommend something else, or I would actually come back into Facebook and actually talk to folks internally and say, "Hey, you've done really well with a certain project, but hey, we have some different needs that the community is asking for. Can we actually do something here to help them out?" Yeah, definitely, we're very community-focused and we definitely listen to their feedback.

[00:47:28] JM: Okay, everyone. Well, really good talking to you. Really interesting conversation. Appreciate you all coming on the show.

[00:47:34] JMarcey: Yeah, thanks for having us, Jeff.

[00:47:36] MC: Thanks, Jeff.

[00:47:36] KK: Thank you.

[END OF INTERVIEW]

[00:47:41] JM: GoCD is a continuous delivery tool created by ThoughtWorks. It's open source and free to use, and GoCD has all the features you need for continuous delivery. Model your deployment pipelines without installing any plug-ins. Use the value stream map to visualize your end-to-end workflow, and if you use Kubernetes, GoCD is a natural fit to add continuous delivery to your project.

With GoCD running on Kubernetes, you define your build workflow and let GoCD provision and scale your infrastructure on-the-fly. GoCD agents use Kubernetes to scale as needed. Check out gocd.org/sedaily and learn about how you can get started. GoCD was built with the learnings of the ThoughtWorks engineering team who have talked about building the product in previous episodes of Software Engineering Daily, and it's great to see the continued progress on

GoCD with the new Kubernetes integrations. You can check it out for yourself at gocd.org/sedaily.

Thank you so much to ThoughtWorks for being a longtime sponsor of Software Engineering Daily. We are proud to have ThoughtWorks and GoCD as sponsors of the show.

[END]