EPISODE 710

[INTRODUCTION]

[00:00:00] JM: Open source projects benefit from the network effects of a large audience of developers, a popular open source project will be contributed to and used by thousands of developers who are continuously testing, deploying and improving the software. The open source movement has created massive communities and a thriving collaborative economy. Infrastructure software companies are increasingly built within an open source business model. Databases, queuing systems, orchestrators, operating systems and search engines have been started as freely available open source projects and leveraged into billion-dollar businesses.

In previous shows, we have talked about business strategy, go-to-market tactics and licensing of infrastructure software. There remains plenty of room for more open source infrastructure companies. We still need better databases and distributed systems management, but over time, open source will move up the stack. From Netflix, to Uber, to Slack, to social networks, to payment systems, all software verticals will become open source or will at least have open source alternatives to the closed source options we have today, and that's because the benefits of making your software open source often outweigh the costs.

For many software business models, the competitive advantage is not found in the source code. It's in the data, or the network effects, or the sales strategy, or the brand. Therefore, it makes sense that someday the source code will be freely available democratizing the infrastructure concerns and letting the software businesses move up the value chain and become less operationally intensive at the bottom of the stack. Rather than asking why should we open source our code, these companies will be asking, “Why shouldn't we open source our code?”

Joseph Jacks is the founder of OSS Capital, a venture capital firm that invests exclusively in commercial open source software companies. Joe believes that over time open source eats everything. In today show, we talk about the future of open source businesses, the impact of licensing, cloud providers and cryptocurrencies. I enjoyed talking to Joe, as I have in several past episodes, and I hope you enjoy it as well.

[SPONSOR MESSAGE]

[00:02:30] 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]

[00:04:06] JM: Joseph Jacks, you are the founder of OSS Capital. Welcome back to Software Engineering Daily.

[00:04:10] JJ: Thank you for having me on again, Jeff.

[00:04:12] JM: We've been talking a lot lately about open source software and what you are doing with OSS Capital. The thesis of this new fund that you've built is that open source software eats everything. To some extent, this has already come true. We have lots of companies that are built on open source. Many of the infrastructure companies are selling

versions of open source software. What are the areas of software that have not yet been eaten by open source? What's still proprietary?

[00:04:42] JJ: That is a really good question. I think, broadly, much of software the stack, as far as the critical components of it, will be open source. By critical components, I mean things that are sort of data path oriented, and the kind of computer science context data path meaning where you're reading and writing data or where you're proxying data connections, bytes, packets between things or you're transformation or data processing, and the answer is really, really hard to distill down to something concise because you're sort of asking what this sort of possible universe of things that could be open. I think, eventually, kind of everything becomes open, but it's sort of a spectrum. It's not necessarily a binary, like literally everything down to embedded software, micro-controller sort of stuff all the way up to every last bit in application software. I think there are certain things that will take a lot longer to become open source software.

[00:05:48] JM: What would take the longest for open source software to eat? If we think about all of the range of pieces of software we could have, we've got our iPhones, we've got our MacBooks.

[00:06:01] JJ: I would say along a continuum, and I've started to blog a little bit about this. I've just gotten quite insanely busy, so I've kind of fallen off the wagon in blogging weekly as I promised, and I feel really bad about that. But I will pick it back up. I think consumer applications will probably take much longer than infrastructure software, middleware, SaaS applications, line of business SaaS applications, proprietary cloud computing, service APIs and their respective services. By consumer applications, I literally mean things like Uber, Snapchat, the icons on your iPhone, right? I think those will take the longest.

Because consumers are the sort of end users, non-technical, like everyday people are the consumers of these applications and they're least sensitive or sort of scrutinizing of how those software products applications are constructed and delivered and built, than say line of business enterprise applications that are sort of packaged, deployed, implemented, installed by more engineering-sensitive and oriented people. Then infrastructure middleware, sort of cloud distributed system stuff, is much, much closer to engineers being sensitive to how things are

implemented than they had highly regard and value transparency and looking at how things are implemented and how to optimize and improve things and so on.

[00:07:34] JM: If you take a company like Uber today, Uber's value is mostly in the network that it's built. Uber connects riders and drivers and there's a massive moat around the data in their database. So it's not the software itself, it's the data that's sitting in their database. Why doesn't Uber open source everything in their software stack today?

[00:07:58] JJ: I think that the reason is they don't really have a compelling business case for that, and they don't have a compelling business case for that because their end-users are just looking for a business outcome. They're not looking for something that they can customize and tailor it to their own specific view of the world. It's a generic platform for delivering an outcome, getting you from point A to point B more efficiently with less friction and so on. I think that's kind of the top level answer. I do you think eventually that changes though overtime.

[00:08:38] JM: Contrast that with Netflix. So Netflix open sources a variety of components of their software and they get to use that as producing leads for hiring. So some people contribute to Hystrix and that leads to that person getting hired by Netflix to come and work on Hystrix as an example. With Netflix, you could arguably say the same thing, “Why don't they just open source everything? What would be the problem with open sourcing? Why selectively open source different pieces of the infrastructure?”

[00:09:09] JJ: I think the answer to this right now is pretty speculative, and again it’s like my deep conviction and I could be wrong about this, but I really believe eventually Netflix does open source everything. The reason they don't now is kind of going back to what you're saying, the value of the platform and services in the network effect that has accrued around delivering a super differentiated and compelling user experience and outcome, which is you don't need to the basically go to a movie theater or purchase a DVD or go through this crazy complex selection process to find the next interesting movie to watch. You just click a couple buttons and you have this menu of recommendations and you can gain access to thousands of titles instantaneously and watch them on demand. It’s a user experience kind of service that doesn't justify kind of an open source delivery model, because the primary end-user of that service is not a software engineer or someone who wants to kind of deeply customize the composition of

that platform or that service to their specific needs from an implementation perspective. That's my overall take.

Going back to Netflix OSS, I think the reasons obviously that Netflix OSS exists is to drive greater engineering recruiting kind of activity and obviously maybe contribute back to open source ecosystem and build a bunch of interesting and reusable technology that shows Netflix is on a cutting-edge or innovating a lot in that area. Sort of tended to somewhat fall by the wayside, and for a variety of reasons, Netflix OSS tooling has somewhat stagnated recently and I think mostly because it is highly Java dependent and coupled to their particular infrastructure primitives. We've seen a lot of things kind of emerge and replace a lot of Netflix OSS specific projects that have sort of language neutral runtime, neutral designs that have turned out to be more compelling I think to a broader set of engineers.

But I think it's actually kind of a mistake on that comment that you make around just improving recruiting overall as the main reason to do open source sort of offices or kind of shared programs like the Netflix OSS program. It goes much further than that. I think open source as a strategy for large companies that are really important big technology companies like Netflix and Facebook and Twitter and Apples on so on are actually really, really strategic to those companies, but they haven't quite figured out why they're strategic or how they could be strategic.

[00:11:52] JM: A couple of other things to add here. First of all, I think the reason that I hear cited the most about why a company does not open source software piece X is actually that it would be difficult to support. So you would have people who are coming in. They're filing tickets or they're commenting on something, and if you're Netflix, you're like, "Go away. I just want to – I'm working on some internal team at Netflix and now I've got random people on GitHub that are commenting on issues that are in my internal infrastructure. Go away. You don't have enough context to understand why the issue has been raised in this manner." That's not really a commentary on the model of open sourcing. That's more a commentary on the communication medium around open source. Talking to Henry Zho or talking to these other people who have issues with kind of the open source management process, whether that's a product of GitHub or that's a product of online commenting systems that we have today, but just the kind of is not a fundamental issue with the open sourcing. It's more an issue of the community side of things.

The other issue that I hear sometimes is like, “Well, this will open up a security vulnerability,” but that again seems kind of dubious, because a lot of these – You could gradually open source or infrastructure. You could make the default to open source infrastructure, but do it in a very gradual manner, and you already have some security reviews around your infrastructure. So security seems like less of an issue, especially at a company like Netflix. I mean, I guess there are vulnerabilities around user data, for example.

Yeah, I think on the gains, the potential gains you have around hiring or also just knowledge sharing. The idea that these companies are like doing write ups of their infrastructure. They're doing blog posts. They want to come on Software Engineering Daily and talk about their infrastructure. Well, you could just open source it and make it like an archaeological dig. Make it like a business case study and more people might take an interest in these sorts of things.

[00:13:57] JJ: Yeah. I mean, I think all these things are true, definitely. Going back to your question about what gets eaten the furthest out of the last. I don't think it's fair to ask the last, because software is constantly evolving and we’re going to have more and more software, not less. So it's just going to be this continuous evolutionary continuum.

I think what’s clear empirically based on this dataset that we have, the spreadsheet, the commercial open source software company index, the cozy index, is that middleware infrastructure software is sort of the category that open source software has shown to grow and I guess "eat the most” over the last decade. This consist something like databases and message queues and operating systems obviously and some application level things, frameworks, middleware overall, and that's data integration, application integration.

What we expect, because that's a pretty huge part of the software stack. IN particular to enterprise IT and just the technology industry overall, this is a multitrillion dollar space. Infrastructure software will continue to shift towards open source based implementation approaches as well as distribution and go-to-market and everything else, and that will probably continue for the next decade or so.

What we think is going to happen after that though is the application layer will start to shift towards open source models. So we sort of codify that as the sort of “SaaS or line of business software as a service category, so things like Workday, and Salesforce, Concur, like all of those line of business, ERP, CRM, HCM, three letter acronyms for application level software in companies will also start to shift towards open source.

I don't think that's going to happen really on mass in a meaningful way until we have at least 5 to 10 times more software engineers producing open source software on earth relative to what we have now, just like 20 to 30 million maybe depending on how much we trust the GitHub data to be representative of that number. I think it’s probably accurate.

So let’s say 20, 30 million software engineers today, 10X that 300 million. I’d say we need at least 10X that. So it gets around 300 million to start to see sufficient empathy concentrate in the software creation process globally such that software engineers can actually implement line of business, enterprise application, functionality with a high degree of precision and functional empathy to actually replace the current approach, which is that you have product managers or business people come up with a great application idea and they go and start a SaaS company.

If you just look at the slew of SaaS companies today, most of them were not started by software engineers and most of them are started by product people and there’s a reason for that. That’s I think a huge category and that’s maybe another trillion dollar category. SaaS line of business applications that will shift to open source probably take at least 7 to 10 years for that really shift heavily over.

I think to answer your initial question though, what gets eaten the last? At least as far as I can see and as far out as I can see, being very speculative here, but I think some of this is potentially grounded in certain areas. Consumer application sort of could potentially come next after that, which is like all the things that normal everyday people use on a daily basis to operate and run their lives. If you open up your iPhone on your Android device and you click through all those different icons and all the different consumer applications that we use, many of them are 10, 20, 30, 40, a hundred billion dollar companies, they’re proprietary software companies. I think eventually those also shifts to open source based distribution models.

I mean, we have – Interestingly, we have one example in the SaaS category of very high growth and exciting company alternative to a proprietary model, which is Mattermost. Mattermost is an open source based Slack and it's a very compelling exciting project and company. This is just an early sign of what is essentially a new kind of emerging category of line of business enterprise SaaS, which is this communications real-time sort of email alternative.

[00:18:16] JM: I've got Mattermost coming on the show at some point in the near future. I'm looking forward to that, but this is an open source Slack alternative. Why does that work? Why is that a viable business?

[00:18:28] JJ: I think what actually matters, and it kind of goes back almost to your comment about Uber's primary value being concentrated in their network and in their data, which I agree with, Mattermost allows their customers and their users to disaggregate the control point of the data itself as well as the network around the deployment of Mattermost server-side components and overall deployments.

You could take Mattermost, which is from what I understand, Go, React and it's fully open source based implementation of something like Slack, but with lots of improvements and their own approach to things, and you can run that on your own server infrastructure. You can control where the data is persisted. You can control how it's persisted. You can control how it's secured and maintained and you consume it over an API as well through a cloud service. But it's basically fully open sourced data path component of the entire application itself. That's very compelling to a large number of people. In fact, in particular, open source projects, which is sort of ironic. But you have large open source communities and ecosystems that have considered moving off of Slack on to Mattermost, because they want to control – They want to have better access and control over the data that consists of all the communication exchanges between people in those different communities and those different instances of the application and a lot of other reasons as well. I mean, you can sort of –

[00:19:55] JM: You can imagine Slack having – Slack hasn't really done much in the way of data science applications that have been built on top of Slack, but Slack has a wealth of data that they can mine and give you layer 2 services on top of. Today, Slack is pretty cheap for the enterprise that want to purchase it. But if they start offering you really, really compelling services

that are built on top of your data and they've already got all your data, then you're going to start realize, "Oh my God! There's nothing I can do, but pay Slack for this amazing service that they're now offering me based off of their data advantage." Now, that's not going to show up for a while. But the companies that are hopping on Mattermost today might be seen that coming.

[00:20:43] JJ: I think that there's space for both models to succeed at a very, very large scale. I don't tend to look at this as an either/or. I think it's more of an abundance sort of philosophy, and you're absolutely right. There's a huge data advantage that Slack has, but there's also a huge open extensibility and kind of giving control back to the user sort of advantage that Mattermost has in terms of an open source approach for how the application itself is implemented as well as the sort of data control lifecycle shifted back to the user.

What's to stop Mattermost though from having lots of concentrated inside and data to build generic models on that could serve a large space by hosting Mattermost and building a cloud service. I mean, they can absolutely do that. I believe they might already be doing that. There's really nothing to stop them from that approach.

I think both companies are very early on in their lifecycle. So we're going to see a ton of innovation on both sides. Again, I'll say a closed approach is viable. An open approach is viable. Both companies can probably be hugely successful. It's like kind of no one even really knows how large that sort of market is, but it's probably in the hundreds of billions of dollars over the next decade.

[SPONSOR MESSAGE]

[00:22:08] JM: Data holds an incredible amount of value, but extracting value from data is difficult, especially for non-technical non-analyst users. As software builders, you have a unique opportunity to unlock the value of data to users through your product or service. Jaspersoft offers embeddable reports, dashboards and data visualizations that developers love.

Give users intuitive access to data in the ideal place for them to take action within your application. To check out Jaspersoft, go to softwareengineeringdaily.com/jaspersoft and find out how easy it is to embed reporting and analytics into your application.

Jaspersoft is great for admin dashboards or for helping your customers make data-driven decisions within your product, because it is not just your company that wants analytics. It's also your customers that want analytics.

Jaspersoft is made by TIBCO, the software company with two decades of experience in analytics and event processing. In a recent episode of Software Engineering Daily, we discussed the past, present and future of TIBCO as well as the development of Jaspersoft. In the meantime, check out Jaspersoft for yourself at softwareengineeringdaily.com/jaspersoft.

Thanks to Jaspersoft from being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:23:40] JM: We very quickly gone down something of a rabbit hole into outliers from the typical open source business models that you see. So the typical open source business models that you see are built around infrastructure software companies, and you alluded to this earlier, I think the cozy index. But anyway, this spreadsheet, it's at oss.cash. That's the URL, but it links to a Google spreadsheet where you've laid out 38 companies. These are open source business models, or companies that are – What's your definition of an open source business model?

[00:24:15] JJ: There is no open source business model. There are companies that implement business models based on open source projects that they fundamentally depend on. That is very subtle and hard for the industry to gain consensus around, because it is subtle. Industry consensus typically happens when things are very easily articulated without definition fragmentation. With open source, we have continuously seen a lack of agreement on almost every concept and framing, because this is an extremely subtle and nuanced area of the technology world.

Different open source business models or commercial open source software company business models are – For example, Opncore is very common business model approach for commercializing an open source project. But it's not a binary sort of you're Opncore or you're not. It's sort of how do you implement Opncore. We've written a little bit about this. There's also

support, so support SLAs. Someone issues a ticket. You're contractually bounded to responding to that ticket within a certain SLA timeframe. You have indemnity and support warranties and so on for the software. There's training. So people need to be trained on at a given piece of software. You charge them for a training program or training and engagement and you go and train them on how to use a software. There's obviously consulting, where you need to go and implement and customize and configure and deploy and install and maintain that software's implementation in different environments and pre-production and postproduction environments and so on.

There's hardware-based distribution models, which are also sort of business models for commercializing open source software. In the spreadsheet that you mentioned though, Jeff, there's actually a column that sort of roughly lays out the different business models that those companies implement or use, and what we've seen is Opncore sort of the most common one. There's actually quite a lack of industry agreement though on what to call that. I just prefer calling it Opncore, because I think there's decent traces of consistency there. A lot of companies prefer to avoid calling that approach Opncore, because in many circles there's sort of a negative connotation associated with Opncore. But yeah, there are definitely a number of commercial open source software company business models.

[00:26:31] JM: Right. The point is your fund, OSS Capital, you're dedicated to investing in open source companies, but you do acknowledge that this is a hard definition to grapple with what in open source company is. You have the definition of an open source company as if a company heavily relies on or builds upon an open source project as the fundamental building block justifying its core product existence, it is definitionally a commercial OSS company. It's a definition of a commercial open source company.

So you have Google that relies on Linux, which you know you could say, "Okay, Google relies fully on Linux. Maybe Google is in open source company." You could say, "AWS productizes proprietary – Or they make proprietary versions of open source packages like Kafka, or Redis, or Elasticsearch. Maybe Amazon is an open source company." There's a wide gradient of things, but you are trying to focus on the things that are leaning towards the Opncore model. Is that right?

[00:27:38] JJ: Well, you said a couple things there, Jeff. Let's just double-click on the definition, because I think you're absolutely right. There is almost zero industry agreement or consensus around what it really means to be a commercial open source software company, and this is the main motivation for maintaining the spreadsheet, because we actually have a lot of patterns and similarities in those companies.

I'll comment on your Google relies on Linux question first. So Google relying on Linux does not define Google as the commercial open source software company, because Google's primary fundamental product's existence is proprietary software implementing very deeply nested complex structures of page rank and their Google search application and something like seven or so other billion user applications, which are all proprietary in terms of their core data paths and the core business logic representing and making up those applications.

They're absolutely not a commercial open source software company in that sense. You could look at our definition. You could go, "Well, let's stretch this out," and say that if every company depends on an open source programming language to implement their business logic, aren't they then therefore definitionally a commercial open source software companie because they depend on that open source or those open source programming languages to like build their products?

I would say definitely not. I mean, every company uses commercial – Every company uses open source tools and projects to build their software products. That's just sort of the world we live in. What we're trying to sort of parameterize, I guess, or narrowly define, is a set of companies in the industry that fundamentally depend on a given open source project or a small number of them. Like in the case of HashiCorp, they depend on I think five or six, Console, Terraform, Vagrant, Vault, Packer and just a very small handful of projects to justify their existence. If those projects didn't exist, the company wouldn't exist. The revenue wouldn't exist. The business model wouldn't exist. All of those downstream effects wouldn't exist. The company fundamentally depends on those projects.

That's as crisp as we are today. I think we're going to get a lot more refined in this definition and improve it overtime, because what you mentioned earlier is 100% correct. We don't have industry agreement and it's one of our big goals to try and get industry agreement in this area.

[00:30:17] JM: The commons clause came out of Redis Labs, and that uses licensing as a way to litigate against cloud providers. So there's this issue where Redis comes out. Redis Labs is started to be a business based around Redis, and the creators of the open source Redis, objects storage, memory-based system can make money. So there's this open source project and then the Redis Labs, which is the company built around the open source project. So it allows the founders of the open source project to work on the technology and also make money. But then AWS steps in and says, "Well, we're also can offer Redis a service. AWS has great distribution channels, because people are already running their EC2 instances on AWS. So they might as well run their Redis on AWS as well." This puts Redis Labs in a tough spot, because they say, "Well, not as many people are buying our Redis as we would like, because they're just going to AWS." So Redis Labs comes out with the commons clause, or I guess you could say Redis comes out with the commons clause. How viable is the use of the licensing as a way to protect the business model of an open source project?

[00:31:42] JJ: So really complex area, and I, instead of commenting directly on commons clause, which [inaudible 00:31:48] from our team actually authored by request of a few folks in the community and the ecosystem, including Redis and in a few other folks. I'll comments on the more abstract thing, which I tends to think is a more contrarian way of looking at this. I think that it will just provide some perspective on my own views here.

I think it is really, really bad, and in fact completely vain for a company behind an open source project or the creators of the project to try and prevent others, meaning cloud providers or other vendors, from capturing value around that open source project. I think that completely violates the ethos, the overall trend of point and effect of open source software almost irrespective of the base open source license, whether it's Apache, GPL, MIT, BST and so on.

Why do I say that? Well, I say that because of a secondary deep belief I have, which is that open source software always creates and generates orders of magnitude more value than any constituent can capture. So if you look at those constituents as being cloud providers, commercial open source offer company vendors or even the creators of the project, none of those constituents can capture a material fraction of the value that open source software

creates in the world, which I think is many, many trillions of dollars in value. It's really hard to quantify that, not just economically, but in many other areas.

So I'll go back to cloud providers capturing the value and the sort of overall stance that Redis and now MongoDb and others are actually taking around adopting commons clause or something like it, which is to prevent and restrict and in fact make it possible to litigate those cloud providers, which I think is just a very, very bad idea.

I think the positive side of a cloud provider building a differentiated service and capturing value around an open source project is this open source project now has distribution into millions of customers that it didn't have before, because the cloud providers elevating it as a first-class service. In addition to that, the cloud provider is potentially contributing back to the open source project and making it better by pushing the scale further across those millions of customers. In fact, we're starting to see this. Amazon, in fact, are actually doing more open source contribution. The natives ever done in the past, and I think people like Adrian Cockcroft and the team have demonstrated a real plan and demonstrate real energy towards contributing things back to Kubernetes and many other projects, and they realized that that's necessary. Maybe they haven't done it at the level of other cloud providers, but I think the point is even if they didn't, we shouldn't be griping over or giving them stress about that.

I think what matters is that the sort of value that open source creates is sort of a pie, and that pie continues to get bigger and bigger and bigger and it's just this enormous sort of value creation pie. But the different constituents in the ecosystem sort of squabble over how much of the pie, how much of the slice of pie are you capturing relative to my slice of the pie and so on? It's just really, really ridiculous in my opinion. The economics are fundamentally different with open source compared to closed source. Open source is just – It's creating, it's generating trillions and trillions of dollars of value. If you can go and capture 1% or 2% of it and build a multibillion-dollar platform company or a $10 or $20 billion platform company, you should be happy in my opinion.

If you're a cloud provider and you can build a slew of services fundamentally based on open source projects and those services generate tens of billions in revenue, but then the commercial open source software company behind the those open source projects maybe perhaps can also

generate a few billion in revenue, which I think is absolutely the case, or even close to what the cloud providers can capture and generate in revenue.

I think everyone should just be happy. I think everyone should say, “Great. Open source is moving forward. There's more open source in the world. There's more value being delivered back to end-users, which is really where the vast majority of the value is being captured and delivered,” and that is just a good net overall for the world. We should not try and come up with commercial license restrictions around who can capture more value. What we should do is come up with sort of a creative Commons or a commercial open source license standardization and frameworks that allow people to reduce a lot of wasted energy in constructing agreements for commercializing open source software. I think that's a more nuanced kind of separate conversation where commons clause could potentially be a good building block or a good stepping stone in a direction.

I personally think that Redis Labs really, in my opinion, did a poor job representing the intentions and the purpose behind something like commons clause, mostly because it's actually a net positive for Redis, that Redis itself is getting deployed and distributed and scaled in lots of different environments, particular cloud provider environments, that only make the software better, only bring it into more end user environments and so on.

[00:37:08] JM: How does that make – I mean, with AWS, they take Redis, they productize it, they don't contribute changes at least as far as I know. I guess I haven’t looked at the specifics. This is what I've heard from I believe people that have been writing about this. But AWS doesn't contribute changes back to the software. They just productize it. They maybe make changes internally, but they don't release those back in the open source world. How does that help the overall Redis community if AWS is just distributing a proprietary version of Redis?

[00:37:42] JJ: If AWS is exclusively forking Redis and distributing a fully proprietary version of Redis and not pushing any of the changes, improvements, bug fixes, and so on back upstream to Redis core, I think that there's a philosophical or maybe a moral argument you could make that they’re not being good open source stewards. I would say there's a really good argument for that for sure.

However, looking at the fundamentals of open source licensing and the point behind open source software, no one should ever restrict or force or expect to cloud providers, or frankly any company for taking an open source project and doing that. Why? Because that's just the whole point in open source. It's free to access. It's free to augment. It's free to modified with varying restrictions depending on the base license of course. But I'll actually cite – I've cited this a few times now. I think it's really, really brilliant comment and 100% accurate and I agree with it fully, a tweet from Doug Cutting, who is the creator Apache Lucene and Apache Hadoop, two open source projects that actually created several multibillion-dollar companies, Cloudera, Hortonworks, MapR, Elastic and I believe a few others. Doug's tweet is paraphrased as basically saying it is insanity to expect a cloud provider, or let's say this is my paraphrase point, any company that is capturing value around an open source project to contribute proportionally back to the open source project relative to the benefit of the value that they capture. It's insanity, because it just sort of violates the whole point behind open source, which is you put it out there and you want to create value in the world.

If you want to capture value around that, there's a lot of different options, a lot of approaches and you have to work hard for that and you have to build differentiated user experiences and earn those customers. But you shouldn't force other people just because they're doing a good job capturing value. That's my take on it.

[00:39:45] JM: Yeah. It does seem in some ways like a question of norms, and – Well, okay. So maybe AWS is not contributing back to Redis their core bug fixes that they should be contributing back according to some certain set of norms, and maybe we can kind of say AWS, that's a point against you in the future. The what degree it matters that we're keeping score of how we're rating AWS is good graces. We can say, "Okay. That's a point against you."

I think what you're saying, so you're intuiting something also that by distributing Redis further, even though it's in a proprietary form, even though they don't give back to the Redis open source repository, there is something about the fact that they are just distributing Redis that is good for the Redis community, and I think that intuition is kind of correct, because even if I'm just using AWS Redis at my company and then I go and start a startup later on, I'm going to think of Redis as a tool in my toolbox and I'm going to go and use Redis later on, and maybe I use AWS Redis again, but maybe not. Maybe I use Redis Labs Redis.

[00:41:02] JJ: There's also two ways of looking at the openness dynamic. There's wire protocol, query engine, API client-side compatibility, sort of openness and standardization, and then there's like the sort of data path part of it. If Amazon is forking Redis and changing a lot of the core run time parts about how Redis and memory catching side of things works or a key value or the core database itself and yet the interface into using the hosted Redis offering is the identical experience and protocol and client tooling and so on that you would use for upstream Redis, that is absolutely going to benefit. Like you say, skills reuse, knowledge propagation and the overall continued adoption of Redis itself in the industry, irrespective of how it is hosted and commercialized in different platforms.

You're absolutely right. I totally agree with that, and there's actually an interesting conversation I had with Kelsey Hightower about this, where as long as you actually have open interfaces, or open endpoint access standardization, or client-side standardization particularly for data services, you really don't need to worry about the core data path runtime component of things. I actually kind of disagree with that. I think the whole stack should be open source and there's ways to commercialize and build certain proprietary tooling where you can actually license software without needing to sacrifice openness on either end of either the interface boundary or the data path boundary. But that's a much deeper conversation. Yeah, to your point, yeah there're two sides to the openness there.

[SPONSOR MESSAGE]

[00:42:49] JM: Managed cloud services save developers time and effort. Why would you build your own logging platform, or CMS, or authentication service yourself when a managed tool or API can solve the problem for you? But how do you find the right services to integrate? How do you learn to stich them together? How do you manage credentials within your teams or your products?

Manifold makes your life easier by providing a single workflow to organize your services, connect your integrations and share them with your team. You can discover the best services for your projects in the manifold marketplace or bring your own and manage them all in one dashboard. With services covering authentication, messaging, monitoring, CMS and more,

Manifold will keep you on the cutting-edge so you can focus on building your project rather than focusing on problems that have already been solved. I'm a fan of Manifold because it pushes the developer to a higher level of abstraction, which I think can be really productive for allowing you to build and leverage your creativity faster.

Once you have the services that you need, you can delivery your configuration to any environment, you can deploy on any cloud, and Manifold is completely free to use. If you head over to manifold.co/sedaily, you will get a coupon code for $10, which you can use to try out any service on the Manifold marketplace.

Thanks to Manifold for being a sponsor of Software Engineering Daily, and check out manifold.co/sedaily. Get your $10 credit, shop around, look for cool services that you can use in your next product, or project. There is a lot of stuff there, and $10 can take you a long way to trying a lot of different services. Go to manifold.co/sedaily and shop around for tools to be creative.

Thanks again to Manifold.

[INTERVIEW CONTINUED]

[00:45:04] JM: You help start the Cube-Con, CloudNativeCon series of conferences and you been involved in the Kubernetes world for a long time. What kinds of business models are enabled because of the proliferation of Kubernetes?

[00:45:17] JJ: Oh! That's an interesting question. I never really thought of that before. New business models because of Kubernetes. I'm not sure new business models have emerged because of Kubernetes. I think Kubernetes has just allowed for better distributed systems to be built, and it is sort of like democratized. Brendan Burns likes to talk about how it's kind of democratize distribute systems knowledge, which I think is absolutely true. It's also just encouraged higher quality of kind of infrastructure for large or small web services and applications running on the internet. I don't know if it's particularly tied to business models that could be enabled. I have to think about that a little bit more.

[00:45:57] JM: We talked about this a little bit at Cube-Con, but there is very little overlap between the Kubernetes community and the cryptocurrency community, and I find this curious because these are two giant distributed systems communities. Kubernetes can be used to run private trusted clusters of compute nodes. Blockchains could be used to run open trustless networks and there's a gradient of trust that can exist between these two types of distributed systems; the private networks and the open public blockchain networks. It seems like these two communities should be cross pollinating. Why are they so disjoint?

[00:46:39] JJ: That's a really, really, really great question and something I thought a lot about. I'm glad you've asked this. Yeah. I think that there's very little collaboration between what I'll call the distributed systems community and the decentralized systems community where the distributed systems community consists of people sort of basing their knowledge and best practices time kind of around hard, well-understood computer science principles, like Paxos service-oriented consensus algorithms and sharding and replication of fault-tolerant computing and obviously all of the things that we see Kubernetes based on sort of leader election, leader follower kind of architectures. I think state reconciliation, state loops, control theory.

Then the decentralized systems community is sort of being based on like everything must be peer-to-peer. We'll just basically use variations of Merkel trees and implement new kinds of consensus algorithms problems across decentralized networks where decentralized networks can be defined also in a kind of maybe quantitatively lose way.

The best post I think that exists around quantifying a sort of spectrum of whether a system is truly decentralized or not as one by Balaji Srinivasan, who's the current CTO of Coinbase. It's a post he did with I believe one of the core technical folks over at 21, the company Coinbase acquired called Quantifying Decentralization. It's really great post. I had link to it in the blog, on the podcast here.

I think that both distributed systems and decentralized systems kind of go hand-in-hand. What I have looked at and observed anyway in a lot of these blockchain exchanges and kind of cryptocurrency startups as well as just even large token, like networks and things, is a lot of them actually run on Kubernetes, which is really interesting. So maybe not like inherently a lot of collaboration between those two communities. But, interestingly, this whole cryptocurrency

blockchain ecosystem uses a lot of distributed systems technology, in particular Kubernetes, to build their products and build their networks and their stacks and so on. So I think that eventually we'll start to see a lot more collaboration and a lot more sort of respect for both sides.

I'll say one other thing here which is kind of maybe an emergent property of how the decentralized systems community has developed from the merchants of the Bitcoin whitepaper about nine years ago to now, we started to sort of see people vector into the community driven by economic interests and sort of digital money and antagonistic stance towards our current financial system. Whereas the distributed systems community has been much more grounded around kind of computer science and technology and building more robust software and things like that. Very, very different sort of entry point, philosophical and intellectual entry points in terms of the people that make up those communities. I think that's probably part of why we've seen a lot of not necessarily polarization, but like a lack of collaboration.

[00:49:38] JM: Well, it's possible. I'm basically asking the question, "Hey, we're at a Kubernetes conference. Kubernetes is based around computers. Computers require electrical engineering. Where's the electrical engineers? Why aren't there electrical engineers at this conference?" Eventually, there is some point at which there is not enough porousness between two communities that either you just don't get as many electrical engineers at a Kubernetes conference.

[00:50:04] JJ: I think you've asked something that's much more directly correlated, which is decentralized systems build and run on top of distributed systems, and there're so many similar goals and outcomes that both of those communities are trying to achieve in terms of building very robust, scalable platforms for large – Millions of users or large amounts of data that I think they should be communicating and collaborating a lot more, and they're not.

[00:50:32] JM: Cryptocurrency protocols are open source. Are you investing in them?

[00:50:37] JJ: I think for OSS Capital, we're really focused on business models and companies that we really understand well. So if you look at the sort of representative list of companies in the commercial open source software company index, companies like Elastic and Red Hat, GitHub, GitLab, MuleSoft, MongoDb, Cloudera, Redis Labs, HashiCorp, many of these

companies, they tend to be sort of B2B, B2C enterprise software companies, really, at the end of the day. Cryptocurrency protocols are a very different animal and sort of fall outside of our circle of competence , if you will. We're just really focusing on what we understand really well.

[00:51:18] JM: Is an open source social network a viable business?

[00:51:21] JJ: I think it could be. I think it could be. It depends on how you define social network. In fact, our first investment that we made and we've actually already announced this, the team announced their round on the 1st October, actually, which is just a few weeks ago as a company in New York, but they're also a distributed team. It's called the Dev Community, Dev.2, and they are the folks behind the practical Dev on Twitter, which is like a really, really awesome Twitter handle. I'm sure a lot of your followers and listeners know it well. Actually, sort of looked at all the activity and the engagement they got from the practical Dev handle on Twitter and all the knowledge and content and engagement there and they sort of felt that building a social network around that would make a lot of sense. So they built Dev.2 and actually open sourced the poll request workflow for generating content on the site and sort of an open source based social network. It's very, very exciting. Lots of really interesting activity around the project as well as the community itself. So if that's any indication, I think you can definitely say that there's a lot of potential for large scale social networks being built fundamentally on core open source based distribution model.

[00:52:36] JM: I agree with you. I've had been on the show several times. I really like Ben. I like the practical Dev. I think he's going to have an interesting time figuring out how to do monetization, because the codebase is open source. The open source idea of it is I think one of the great appeals of it. That's the whole OSS Capital, like you're putting money into an open source project. But the thing is if you put – Like if Facebook core open source, people would see all the places where, "Oh! There's advertising and tracking, and it's like that's – I don't want that in my code." So if you have an open source community, if you monetized by advertising, the open source people might say, "Oh! There's your ad insertion code. I don't want that in my community." Is there some problem here where if the users see the injection of monetization related code, they will balk at it or do you think the users are sophisticated enough to say, "That's okay. I'm okay with that."

[00:53:41] JJ: I think the latter. I think the latter, but it's a really delicate balancing act and you have to be super careful with learning, maintaining and in fact growing trust with your community and with the people who have made a given open source project successful as well as contribute to a product or sort of an application user experience based on that open source project. I think Ben and team are super, super empathetic and sensitive to the trust that they've earned and the trust that they're continuing to grow in their community.

So there's definitely lots approaches in different ways that commercialization could play out there, but I think if you look at how like, for example, GitLab, has been really transparent about the approaches for commercialization and sort of their open and close kind of dynamic, what aspects are close source? What aspects are open and the sort of open core model being very transparent about that? Being sort of direct and clear and detailed about all those features down to roadmap level visibility, I think end-users and consumers of those services ultimately realize businesses have to be built. People have to be paid, and that dynamic over time, it's just something that emerges over running communities really well and earning high levels of trust.

[00:55:02] JM: I think what you should do is you just make it like – Think of it like an newspaper. It's like an open source newspaper site where you can pay to opt out of advertising and tracking if you want, but if you don't pay, you have the alternative of using it like any other media site. I go to BuzzFeed all the time. I know BuzzFeed is tracking me and serving me ads, and I'm okay with it. I make that bargain, and even if BuzzFeed were open source, I wouldn't talk trash about them. I would just say, "Yeah, that's the other bargain I've made."

I find that the Patreon model, the subscription model, that seems to be like a less explored side of things. Do you think the Patreon model for open source, do you think that could be a venture scale return business model in the future? Do you think that there will be a model of just people that support something like an open source community?

[00:56:02] JJ: Last I checked Patreon themselves, which doesn't directly answer your question, but Patreon, the company, has raised a little over $100 million in venture capital across series A, B and C. I think that that is really interesting innovation and business model side of sort of a digital kind of tip jar approach to subsidizing energy and time spent on various projects, many of which are open source projects on Patreon. I'm actually to few open source projects on Patreon,

where you could potentially build something venture scale returnable. I'm just not as clear about how something like Patreon explicitly apply to open source what that could look like. But I think it's possible. I don't have a strong enough opinion there.

[00:56:49] JM: Fair enough. That's far-flung. If you're a venture capitalist, venture capital can be highly competitive. If you want to have an edge you need to differentiate on some axis, you need to differentiate on maybe the capital, the amount of capital you give, the value-added services you give. Maybe it's your brand recognition. Maybe it's a set of LPs that can move and shake things in your favor if you're a company that gets pulled into the wing of a venture-capital brand. What is your differentiator when companies are talking to you and they're trying to raise money from you or you're trying to convince them to go with OSS Capital? What are your differentiators?

[00:57:34] JJ: This is a really great question and I'm always excited to answer this question, and it will be for many years to come, because I think we have something really unique and truly differentiated. There are two levels of value add and sort of value proposition to our founders and the portfolio companies that we serve as well as new founders that are evaluating us as a potential partner, and those two things are as follows: So we have a full-time firm and partnership where it consists of a few different folks, obviously myself, venture partners. We have portfolio partners, which is sort of a new construction, and we're all deep open source experts. That's all we do. In fact, that's all we've done really for many, many years. It's like sort of decades of experience across the partnership in different areas, in engineering and legal and finance, and community development, and we've been part of contributing to some of the largest, most successful open source projects in the world.

In terms of the overall secondary extended network that we have, which is the sort of second layer of the value proposition, we have a network of the founders of the largest, most successful commercial open source software companies of the last decade, many, if not most of the companies in the index that we talked about earlier, this commercial open source software company index. The value that we can provide to open source founders, commercial open source software company founders, is that we can sort of contextually provide the value and the insight and the wisdom of the experience from the large kind of commercial open source company founders of Elastic, which are also often times engineers and project creators to the

next generation of commercial open source company founders, because we have this network, which consists of an incentive structure. Many of those individuals are direct supporters and limited partners on our firm as well as just overall explicitly aligned with our model. We only support, we only invest in commercial open source software companies.

I think those two things are really where we concentrate our value proposition, and I'd say one final abstracting. As a firm, we're really razor focused. I mean, all we do is investing in commercial open source software companies. We don't really have a sort of a generic services arm. We're not claiming to provide any sort of generic business experience or product value, value proposition or any sort of vertical specific thing. We're specifically saying generically we will only support in investing in commercial open source software companies, and we have a network of founders of the last generation of these commercial open source software companies who have created tens of thousands of jobs, hundred billion plus in value, raised upwards of 10 billion in venture capital and many other things.

We hope that structure can add a lot of value to the founders who decide to work with us.

[01:00:29] JM: Last question; do you have any open source product ideas or business models that you would like to see people approach you with or that you would like to see people building?

[01:00:41] JJ: We've actually thought about a few things. I don't think that we're very opinionated or try and sort of encourage very specific companies get built. I mean, it's a huge undertaking and extremely risky to go and start a company in a contrived way.

[01:00:53] JM: But this is just a request for startups. It's not a contrivance.

[01:00:56] JJ: I would say you have to think about that one for a bit and I'd get back to you. Yeah, we have some really amazing, incredible people that we talk with constantly, and there's hundreds of folks who've engaged and reached out from our [inaudible 01:01:07] the month and we're super honored and excited by that. I think we're just really excited to see more business model innovation. I've said this a few times in other interviews and other venues. I think we're going to continue to support that. There needs to be a lot of innovation around how value is

captured with open source software and we're super supportive of awesome ideas that help move that forward.

[01:01:29] JM: Joseph Jacks, thanks for coming back on Software Engineering Daily.

[01:01:31] JJ: Thanks, Jeff.

[END OF INTERVIEW]

[01:01:35] JM: This podcast is brought to you by wix.com. Build your website quickly with Wix. Wix code unites design features with advanced code capabilities, so you can build data-driven websites and professional web apps very quickly. You can store and manage unlimited data, you can create hundreds of dynamic pages, you can add repeating layouts, make custom forms, call external APIs and take full control of your sites functionality using Wix Code APIs and your own JavaScript. You don't need HTML or CSS.

With Wix codes, built-in database and IDE, you've got one click deployment that instantly updates all the content on your site and everything is SEO friendly. What about security and hosting and maintenance? Wix has you covered, so you can spend more time focusing on yourself and your clients.

If you're not a developer, it's not a problem. There's plenty that you can do without writing a lot of code, although of course if you are a developer, then you can do much more. You can explore all the resources on the Wix Code's site to learn more about web development wherever you are in your developer career. You can discover video tutorials, articles, code snippets, API references and a lively forum where you can get advanced tips from Wix Code experts.

Check it out for yourself at wicks.com/sed. That's wix.com/sed. You can get 10% off your premium plan while developing a website quickly for the web. To get that 10% off the premium plan and support Software Engineering Daily, go to wix.com/sed and see what you can do with Wix Code today.

[END]