EPISODE 943

[INTRODUCTION]

[00:00:00] JM: Every company has a software supply chain. A company builds its products from custom code, paid APIs, paid proprietary binaries and open source software libraries. As the types of software available have increased, the management of the software supply chain has become complex. Large software companies have always needed to ensure the security of their software.

With the growing variety of open source licenses, these companies also have to deal with an increased set of legal complexity. If an open source project is used in a way that violates an open source license, the company is subject to legal risk.

Fossa is a company that focuses on automating the management of open source compliance and security. Kevin Wang is the CEO of Fossa and he returns to the show to discuss the modern issues of software licensing and his experience building a company. It was a great pleasure to talk to Kevin, and he's quite an interesting person to discuss open source licensing, and businesses, and cloud providers with. So it was a great conversation.

[SPONSOR MESSAGE]

[00:01:17] JM: Email has been around for longer than I've been alive, but there's been surprisingly little innovation in inbox management. SaneBox is a new way of looking at your inbox that puts features like snoozing, and one-click unsubscribe, and follow-up reminders as first-class citizens.

If you are overwhelmed by your inbox and you're almost ready to declare email bankruptcy, tryout SaneBox. In the onboarding process, SaneBox analyzes your emails and helps you sort them into categories. You can get a free 14-day trial and a $25 credit by going to sanebox.com/sed. That's S-A-N-E-B-O-X.com/sed.

These days, I spend more time in my inbox than I do in front of my coding environment, and back when I was programming a lot I would spend hours configuring my coding environment because I wanted to maximize productivity. If you spend as much time managing email as I do, it's crazy not to set yourself up for success with your inbox. Stop the craziness, get sane with SaneBox. Go to sanebox.com/sed and get a free 14-day trial as well as a $25 credit.

Thank you to SaneBox for being a sponsor of Software Engineering Daily.

[INTERVIEW]

[00:02:50] JM: Kevin Wang, welcome back Software Engineering Daily.

[00:02:52] KW: Thank you so much for having me back on the show, man.

[00:02:55] JM: Last time we talked some about the license changes to open source projects. Let's start with that subject. Give me an overview for why some software companies are changing the licenses on their open source projects.

[00:03:10] KW: Totally. More broadly, I think there is like a Renaissance era right now inside of licensing experimentation. So there's a lot of these open source infrastructure companies. We can think like databases like Redis and Neo4j that see these cloud providers, like Google, Amazon and Microsoft come out with competitive versions of the open source projects.

Some history is most of these open source projects are extremely brittle. All of the development is funded by one or two companies. So if a giant monopoly like a Google or an Amazon can come in and suck a lot of the air out of the market, they can seriously threaten the future of the open source project underneath.

In response, a bunch of these projects have changed their license to things that protect them from this cloud provider problem. So things like the SSPL, the Commons clause, and they basically have the effect of saying, “Hey! You can use all our stuff for whatever purposes, but not you cloud providers.”

[00:04:05] JM: Do you think this is a good business strategy?

[00:04:08] KW: I'm not sure. I think we're really early on in what building a great open source company looks like, and there's a lot of experimentation that's happening that I think is extremely positive. My read on this entire set a new licenses is it's a pretty reactionary step. It's not like the business models were sort of crafted from scratch with this plan saying, "Hey! At this date, we're going to suddenly change these licenses because of this cloud provider problem."

I think most open source companies have underestimated how dominant these cloud companies would be and how aggressive they would be spinning up competitive products with open source infrastructure companies. I think it's an important step in terms of the reaction from some of these companies if there were a way to run the experiment again. I'm not exactly sure this would be the business model, but we're still early days.

[00:05:02] JM: I am with you that the open source providers underestimated the cloud providers. Do you think the open source providers also underestimate their own core competency? Because if you are one of these database systems, don't you know so much about your own software that you can spot the adjacencies that you could expand into rather than changing your license?

[00:05:28] KW: We don't know. I think that's a fair argument. Again, when it comes to things like this business strategy, you're sticking the company on that bed. I think if you say, "Hey! We have an advantage because we have better talent. we understand this is a core competency way better than these cloud providers," that's a lot to stake on that bit of speculation when the reality of it is these cloud providers are extracting hundreds of millions of dollars out of the market at a much faster rate at much greater scale than all these open source companies can.

At the same time, it's also hard to say that you really do have a better core competency because these cloud providers are so good at cloud. They're so good spinning up these extremely scalable things. They have so much baked in compliance, and security, brand power or all these other stuff on the same platform. If creating amazing cloud infrastructures is a core competency that you're under, I'm not exactly sure you can be in Amazon, Microsoft or Google.

[00:06:28] JM: What has been the reaction from AWS to these companies changing their licenses?

[00:06:38] KW: I don't think I've actually seeing that much outward activity. I think Redis was forked by Amazon, and there are a couple of forks that were getting maintained. I think Amazon is having new proprietary development efforts that are compatible with some of the existing open source APIs. But I haven't really seen a reaction where Amazon says, "Hey! Let's go communicate or figure out a better way to interact with these open source companies that don't make them feel like this competitive threat."

I think the stance from Amazon seems to be pretty commercial, which is what we'd expect. It's like pretty capitalists. It's like, "Hey, you're going to do this thing. Okay. We're going to do that thing." I think we're seeing a pretty capitalists reaction to another pretty capitalist reaction. I think Google is taking a much more collaborative approach and so is Microsoft.

I think a lot of these open source providers now that they see this predatory behavior coming from Amazon are finding other avenues to gain access to these cloud platforms, and Google has done a bunch of these partnerships. Microsoft has done a bunch of these pretty deep commercial partnerships with a couple of vendors. In some ways, that could be a one way where a cloud provider can get a competitive edge over another.

[00:07:47] JM: It's quite a game of diplomacy.

[00:07:49] KW: Yes. There is a lot of stuff going on for sure.

[00:07:53] JM: If you are Amazon and one of the open source companies changes their license, is there some kind of recourse? Can you pay them some subsidy? Can you pay them like a percentage? If Amazon Kafka service wants to run KSQL, can they pay some percentage of – I don't know. Is there an agreement that they could reach?

[00:08:20] KW: Well, totally. I mean, whenever you put a more restrictive license out, the message that it sends doesn't say buzz off. It that you have to talk to us now, and if we want to figure out some sort of commercial arrangement, we actually have to figure it out together. I

think the motivation by some of these open source companies was to spark a dialogue with some of the most powerful cloud providers and actually figure out a model where both the open source project and the future of it could benefit and the cloud providers could still benefit from those changes.

I'm actually not aware of any agreement that's been reached by Amazon so far. I know Microsoft and Google have come to a couple of commercial partnerships, but it seems like Amazon is still chugging along with some of their internal development efforts. Would've you heard?

[00:09:02] JM: I have not heard anything about Kafka at least. I mean, the only thing I've heard is Amazon forking Elasticsearch and making a new distribution called open distro or something, Elasticsearch open distro.

[00:09:15] KW: Same with Redis as well. An important thing to remember is that when you change a license, that doesn't apply to the history of the project. That only says changes moving forward. If a license change happens in 2018, every version before that date, the 2017, 2016, that version of the software is still under the old permissive license or whatever open license that you had in the past.

Now the competition or the surface for the competition doesn't happen throughout the entire history of the technology. You can't take it away from open source. Instead, it's just that new improvements to the project are now proprietary and it's sort of your developers versus the other developers. But you start off on the exact same level playing field.

There's an argument that some of these changes already came a little too late, because so much value has already been produced and the core technology is already out there. Now you're no longer fighting a war over the technology, but you're fighting a war over the channel.

[00:10:16] JM: That's why I think they're playing with fire.

[00:10:18] KW: Yup.

[00:10:18] JM: It's dangerous. I mean, you could totally blow up your community doing this.

[00:10:23] KW: Yeah. Well, what other choice do some of these companies have? I don't know.

[00:10:28] JM: Figure out a new business line.

[00:10:31] KW: I don't know how feasible that is. The entire history of commercial open source projects, like some of the most successful businesses, tend to have a very heavy core open source project but a really, really small and simple business model that's natural around them. The business models that have gotten away too complex are the ones that's much hard to scale and grow.

Redis is an awesome example. They both produce Redis as the open source project and they do hosted Redis with a couple of extra enterprise features on top of it. It's just simple. You make money by hosting the thing that they're really good at making. Same for a lot of these other infrastructure providers. They have some sort of pretty thin layer on top of it and it's not too big of a logical jump to innovate on a secondary business model.

I think as the discussion about how to create a commercial open source company matures, we're seeing a lot of patterns where the right way to identify one is to find a really natural way to build the business on top of the core project that you have. The challenge is if there is some sort of cloud provider or a massive monopoly out there that can just eat up the core project and then do that simple step to the business in a such better way than you can, that's a huge problem. That kind of breaks one of the most prevalent patterns we have today about building a successful commercial open source company.

[00:11:50] JM: But commercial open source is like 10 years old.

[00:11:54] KW: Totally.

[00:11:54] JM: It's very immature. This whole criticism is built around a company that started as an online bookstore.

[00:12:05] KW: Right. That's a good point. I mean, it's a –

[00:12:08] JM: If an online bookstore can find its way to being a cloud provider, certainly a database company can find its way to a second business line.

[00:12:20] KW: Oh! I'm with you in the fact that I feel very optimistic about the value that's been generated by these commercial open source companies. My only point is that I think it's a reasonable step to say like, "Hey! Let's put us from guardrails to protect the business in the interim," and then it takes time to change strategy. It takes time to change course and the innovation on these business models are pretty long tail. Supporting these open source projects for instance, paying for the development takes huge amounts of resources.

So you can't simply take all the developers that are contributing and building on to Redis, which is mostly one company, or Neo4j, or Elasticsearch, or any of these other technologies and say, "All right. Stop doing this, and you're going to completely work on something new.

[00:13:03] JM: I mean, I don't know what their balance sheets look like. They could be profitable. I have no idea how much money they're making.

[00:13:09] KW: I think in many of these businesses, they're pretty successful. I don't think that there's that much existential risk baked into some of the most popular brand names right now. I think there is a huge graveyard of smaller companies where the air has just been sucked out of the market by some of these cloud providers. So it makes previous businesses just no longer viable. That happens, right?

Markets become saturated. There are vendors that become dominant and monopolistic and people have to innovate and find new business models. I think that's totally fine. I think many of these companies will survive, but I think it's also reasonable to say, "Hey! We've had this strategy and this whole business line that's been possible because we've previously built this technology and put it on the wild." If the world and the landscape changes and that business model is no longer viable, then a license strategy might not be viable anymore.

[00:14:04] JM: It's jumping to conclusions so quickly. It's like you said, it's very reactionary.

[00:14:09] KW: Totally.

[00:14:10] JM: Again, I don't know what their balance sheet looks like. I did an interview with the DataStax founder. What's his name? The CTO. I don't know. He's done a ton of work on DataStax. Actually, they did I think an open core thing. They have an open core-like situation. I was talking to him and I was like, "Is this really a good strategy?" and he said, "Absolutely." He said, "We have seen the benefits to our bottom line of making of–"I don't know the specifics of their license change, but they – In their defense, in these open core companies defense, it's not mutually exclusive. You could build a protective license and also search for a second business line. Amazon is a tough competitor. I'm not like passing judgment. I think it's just risky.

I mean, we have seen – This always get said about the Hadoop ecosystem, that the Hadoop ecosystem really suffered because there were two companies that were competing over the same open source project. That's where we're headed for.

[00:15:16] KW: Well, I think, one, the reality is I can't think of an open source project that's not Linux that hasn't been really dominated by the contributions of one or two commercial entities. I think, one, the reality is –

[00:15:30] JM: Kubernetes.

[00:15:31] KW: That's true. Yeah, Kubernetes is another one. We're talking about like a couple out of sample size, like thousands and thousands.

[00:15:38] JM: Bitcoin.

[00:15:38] KW: I think if we're to articulate the world of commercial open source projects, there is this kind of notion that this huge groundswell community effort and the reason why you have to have these extremely open governance models around is because you have to protect the interests of all these developers in the community that have staked their livelihoods on basically committing to this project.

The reality is that if you truly inspect the list of the top thousand commercial open source projects, it's like one or two key contributors, and they're all companies that have had to find some sort of business model or one governing structure or some foundation that's had to find one sort of commercial model around raising money to fund the development and future of these projects.

That hope, that picture of that groundswell community effort is just not a reality for many and many of the most popular open-source projects out there. I think if we need to find new ways to be able to support them, I think that's fine. I'm with you in the sense were many of these licenses that we see today are just the first step rough drafts in trying to experiment with this stuff.

I worked on the Commons Clause, which was the first of the licenses of this entire family. That was deliberately written and built to be kind of a temporary first step just to get the discussion started on all of these things. Maybe we'll find different models and maybe the answer is to partner with some of these providers, and the providers have a competitive interest to be able to eat more of these partnerships than the other ones. Who knows? But I think the experimentation is very good.

[00:17:13] JM: Yeah. All right. Well, let's move on from licenses for now. I don't want to go too deep down there. Explain what your company does. Explain the Fossa is. Why are you an expert in open source?

[00:17:24] KW: Sure. My name is Kevin, I am the CEO of a company called Fossa and we basically help big companies manage all of the open source and third party software they use. We built these code analysis tools. They integrate with your development environment and then we can automatically track and manage all these open source components.

A huge part of what we do and a big solution we provide is license compliance. If you have a big company using tens of thousands of open source licenses, we can completely automate the process of figuring out your obligations and complying with them at scale. So we deal with these issues all the time. Tons of the most popular open source projects and developers on the world use our tools for this purpose. We have a long history inside this community. We get called very

often when any issue around open source licensing pops up, and that's how we got looped into this whole cloud provider story. Yeah.

[00:18:15] JM: Did Amazon call you?

[00:18:18] KW: I think we talked to a couple of people from Amazon. We got a lot of calls from commercial open source vendors on what we do. Also, one of my close partners, Heather Meeker, whose sort of like an OG licensing celebrity. She's a lawyer who's worked on a lot of these early licenses. Was my partner for writing the Commons Clause and also kind of working with a lot of these commercial open source companies to figure out some sort of solution.

[SPONSOR MESSAGE]

[00:18:50] JM: If you are a SaaS or software vendor looking to modernize your application distribution to gain more enterprise adoption, checkout replicated.com. Replicated provides tools to deliver your Kubernetes-based application to enterprise customers as a modern on-prem private instance. That means your customers will be able to install and update your application just about anywhere.

Bare metal servers in a cloud VPC, GovCloud and their own Kubernetes cluster, vSphere. This is a secure way the your customers can use your application without ever having to send data outside of their control. Instead of your customer sending their data to you, you send your application to your customer.

Now, this might sound difficult and maybe you're not used to it because you're a SaaS vendor. You're a software vendor, but Replicated promises that recent advancements from tools like Kubernetes make it far easier than before, and the Replicated tools can help vendors operationalize and scale this process.

The Replicated tools are already trusted by noteworthy customers like HashiCorp, CircleCI, Sneak and many others. As a result, over 45 of the Fortune 100 already have an application deployed via Replicated in their infrastructure. That's a strong sign of adaption.

Go to replicated.com for a 30-day trial of the full Replicated platform. You can also listen to an interview with Grant Miller, the CEO of Replicated, that we did a while ago.

Thank you to Replicated for being a sponsor of Software Engineering Daily, and you can check it out for yourself at replicated.com and get a free 30-day trial.

[INTERVIEW CONTINUED]

[00:20:58] JM: Some of your customers, you have like Uber and Zendesk. These are people who run Fossa in some kind of continuous delivery workflow or on some regular basis, some Cron job. It iterates through all their code and does static analysis to make sure that they are in compliance.

[00:21:17] KW: Totally. The broader narrative is it can't be a good software company right now without being graded open source, and the reality is all modern technology now is built with open source. Uber is built with it. Zendesk is built with it. A million and one software companies out there.

What we do is we basically come in, we install our code analysis tools into the CI process, into the code host, into whatever area where there's active code development going on and then we can see every single time a developer brings in a piece of third-party code from a build tool, from copy and pasting and checking it directly into the code and figuring out what the licenses on top of it. If there are any security vulnerabilities, quality issues, anything else that would be relevant to the health of the business. Then we can help them automate license policies, automate vulnerability management. All of these things are just really critical to the workflow.

[00:22:09] JM: I don't know how much you can talk about this, but I know Zendesk is one your case studies. How did the open source challenges of Uber compared to Zendesk?

[00:22:22] KW: I think a lot of these challenges from company to company are pretty similar. I'd say the really hard part of this problem is everybody has a completely different way of building software. They might all use something like CICD, but they use different tools. Their ways of

bringing in open source are different. They have different build tools, different ways of organizing the code.

So the ecosystem is extremely fragmented. It can create a ton of value by making all that fragmentation easy with one simple tool that that understands all of these different ways that you can write code and being able to structure it all into something that's sensible into a one single process.

I'd say across most of the companies that we run into, we always see a ton of complexity being like 20 or 30 different build systems ways of using open source, languages, current processes. Big companies also like to grow through acquisition. A lot of times we'll see a new team that's been acquired, and that team has a different set of tools and a different set of ways of using open source. The value comes when you can bring all of that stuff together into one interface for, say, your licensing team. One way of enforcing policy is in one kind of control plane or behavior.

[00:23:30] JM: What kind of control plane would I do? Would I like set flags or something? If this kind of thing comes up, set a warning or halt my software. turn off all my software if this happens. What am I doing?

[00:23:43] KW: Totally. One of the most critical pieces of our tool is basically this whole flexible policy engine. At a high-level as a business user you can say, "All right. These licenses are good. These licenses are bad. If I see these licenses on these teams or if these certain obligations are getting triggered or something gets statically linked or whatever, I can either stop the build process. Meaning, get feedback immediately to the developer and say, "Hey! You can't bring this change in," or I can send flags somewhere inside of my organization for review, or I can basically control how all of that stuff flows."

The key part is that this was built to all work in real-time. A lot of times we see the way that companies deal with this is some sort of internal fire troll. You find something out, you hold a meeting with the lawyer and the engineer. they try to figure out what's going and do a bunch of back-and-forth. If you do this during the release process, you halter release, like the huge

painful fire drill. We can turn that into basically this operationalized process that runs. It's just part of the everyday software development.

[00:24:44] JM: We talked about this in some detail in our last show. What have you been working on since then?

[00:24:50] KW: Since when? Since the last show?

[00:24:52] JM: Since the last time we talked. I guess that was like 9 to 12 months ago?

[00:24:56] KW: On yeah! Totally. We had a lot in news. We announced our series A, finally. We had actually raised that a little while back, but we've finally announced it a couple of weeks ago. It was 8.5 million in new cash led by Bain and Costanoa and Norwest also participated in it. Since then, we've significantly grown our customer base and grown sort of the size of the customers that we work with.

The entire identity of the company now is really focused on how to solve these open source problems inside of a big company. So we believe we've built the first platform that works really well for the big enterprise and can manage all of the complexities involved when you have thousands of engineers and hundreds of thousands of open source components getting used at scale. Since then, we've onboarded some really, really large customers. You mention two of them, Uber and Zendesk. We also have Risen Media. Twitter is one of our customers. Just big software engineering organizations.

[00:25:50] JM: What kind of engineering problems has that led to for you as you get these gigantic customers? How has that changed your engineering requirements?

[00:25:59] KW: Totally. Our engineering team has been forced to grow up very, very rapidly, and our headcount overall as an organization has really, really grown. We're targeting about 50 people by the end the year and hopefully significantly more than that early into next year. In terms of our own internal engineering process, we have two very different kinds of customers. One customer likes to take our tools and running all on-prem. The other customer likes to use our own cloud products, and we have to support both their channels.

Just the maturity level on how requirements come in from our customers, how we work with them, how we test it and then how we release it both on on-premise channel versus a cloud channel has really had to level up in a whole variety of ways, and we have a lot more people involved in that process now. We have a customer-facing team, sales engineers, customer success managers that collect and parse requirements. Communicate with a customer. Deliver these solutions with them. There's just so much more that's going on and it's been a real growing experience for the team.

[00:26:56] JM: The process of going from a SaaS product to an on-prem product is something we've done a little bit coverage of lately. What was your process for taking a hosted SaaS product and creating an on-prem product? Is that hard?

[00:27:12] KW: Yes. I think it's absolutely hard when you have to maintain two different ways of doing the software, and we actually saw this coming. The most common piece of advice I think you'd get as a very early stage company is you pick one. Either do on-prem, or do SaaS, but don't try to do both. We tried to do both and we're still doing both.

In the earliest version of the product, we actually started with on-prem because we knew that the majority of the constraints that we're going to see that we're going to be really hard to work around was actually going to stem from the on-prem architecture and the amount of cost, and supporting customers was mostly going to come from the on-prem side of the house. In SaaS world, we have way more flexibility. We have way more control, way more room to screw up.

[00:27:57] JM: Bold.

[00:27:58] KW: Yeah. We actually architected the initial version of the software knowing that eventually we'd target SaaS, but we have to instrument it really well for on-prem. It was actually on-prem first, yet we also architected the application layer like a SaaS product.

The actual infrastructure was built like an on-prem product, but the application was built like a SaaS product to support multi-tenancy and all these other things. Then when we made the transition over to SaaS, we already had everything containerized. We had these units of

scalability, because we know our customers would have scale us on-prem and the application was assigned to be multitenant from day one.

So the transition for us was actually pretty easy. I think the supportability is a challenge that's fixed for every single company out there, where you have different customers. They want to run in an air gapped environments off their own Oss. Containerization is made easier for us. But when we start deploying, even then, like Kubernetes was super young. Docker wasn't stable on many versions of many operating systems out there. So they were our fair share of nightmares.

[00:29:04] JM: That's pretty unconventional. You bit off the hardest part of your end state first instead of going with the easier SaaS product.

[00:29:14] KW: Maybe it was naivety on our front to pursue that.

[00:29:17] JM: No way. No way!

[00:29:20] KW: We want to be ambitious, and ultimately the thing that drove us towards that decision was market opportunity. We felt like, "Look, there's tons of people who want this stuff in SaaS. There're tons of people who want this stuff on on-prem." If we were to be too prescriptive to early on about that, it might've just been a bad move for the business.

I think we believe that largely it'd be true. I think our customers come in roughly half in SaaS, half in on on-prem. That's true not only in count, but also by dollar revenue. We've been surprised by seeing such a healthy blend between the two release channels. By doing that early on, it's forced to get really good at the quality control processes of both of those channels and it hasn't been as operationally difficult as we expected. Meaning, not that on-prem is an operationally difficult or SaaS is an operationally difficult, but the cost of maintaining the two hasn't really costed us additional focus points in one area or another.

[00:30:18] JM: But it's definitely not naïve. I mean, what I like about it is you were deterministic. You took the approach of saying, "This is going to work. This is going to be a good product, and we're so confident it's going to be a useful product. We're going to do the hardest part first,

because if we think about it over the long-term, doing that is going to net us less work than doing the easy part first."

[00:30:43] KW: At the beginning it didn't feel like that much of it, but it just felt like, "Okay. Both kinds of people want it. So we just have to build both." It felt like just a regular requirement, and I think maybe some of the pain experienced by many organizations is doing this early on is way easier than doing it when you already have a really established engineering staff and set of processes and existing architecture in place.

I think we got lucky in a lot of areas. The majority of our decision-making from the early days of the business up until today were all just driven by market. We consider what the market wanted and then we were their slaves. We do whatever they needed. Whatever was possible. So we never really over optimized too much for envisioning the future where we could say no to one set of customers or another. It was still optimizing for the whole discovery process of what the market wanted.

I think the market wants both. There're still a lot of people out there that need on-premise software especially for something we do, which touches your code and touches compliance and security related issues. It was good bet for us. I think we're one of a few companies that would actually say that.

[00:31:52] JM: Is there a specific engineering problem within Fossa that has been particularly acute and difficult to deal with?

[00:32:03] KW: I think there are a lot of really exciting engineering projects that we have. I would say the majority of the most hairy ones come from the growth of the company in a short amount of time. But in terms of the scope of like what we need to build, we think of our infrastructure in three ways; data discovery and dashboards.

On the data side, part of what makes our system work is we have to build like the Google for open source. Any time there's a new open source component, we have to find it, analyze it for licensing and security issues and then build this gigantic real time database of every open source package out there. At the same time we have to be able look at a piece of code.

Regardless of the environment, weather there's a build running, not a build running, regardless of whether the person running it even knows how this codebase works, we have to look at it and figure out what the third-party code is inside of it.

Both of those are really, really difficult technical challenges and also they have also very broad domain. I can give million one languages and million one ways you can deploy an open source package. I think supporting all that domain is definitely a lot of work.

[00:33:09] JM: Has there been anything where the business has shifted in the sense that you've seen opportunities because of customer requirements that you didn't anticipate?

[00:33:24] KW: Yeah. I think that's the eventual future of all enterprise software companies.

[00:33:29] JM: Except for the open source database providers.

[00:33:31] KW: Of course. I mean, I think in the early days, it pays off to have really big vision and strong perspective of what the future looks like. But if you're an enterprise software company, eventually you're going to just really interact with the market in a very, very tactical level that you might not have in something like a consumer company.

But like you close a contract, that contract have very specific requests and requirements inside of it. Then as you go to more broad customers and you have more people on the ground deploying these pieces of software and interacting with the customers, you get your product roadmap read out to you by how you've gone to market.

Overtime, our customers have just given a somewhat incredible feedback that's basically helped a lot with just the directionality of what the real opportunity looks like. Why? I'd say this is one of the natures of enterprise software. You build a ton of what your customers need and you realize all your other customers need this thing tuned and eventually it becomes this large platform that has all of these stuff that came from the aggregation of interacting with the market.

Once in a while you sort of pack it all together, clean it all up so feel like one whole cohesive story again and then you grow your customer base again. You do that over and over until you achieve market dominance.

[00:34:45] JM: And expand into adjacencies, like find new markets, right?

[00:34:50] KW: Totally.

[00:34:51] JM: Do you have any ideas? What will those adjacencies be?

[00:34:55] KW: I mean, for us, the vision for the entire platform we're building is basically a platform to help enterprises maximize its use of open source. The product surface area is extremely broad. Right now we do compliance stuff. We do security stuff. We do some code quality stuff, but like all of the ways that a big enterprise company can interact and get value out of open source is on the table for us.

Many of these expansions for us is normal. We sit in the engineering and development process. We have an immense amount of data about the third-party software and people use and we have this kind of proprietary dataset of what like proprietary companies use over the rest of the open source ecosystem.

From that, there's a huge opportunity to build a million one features. We have a million one ideas and we have only so much time. So we build basically what our customers want the most. But just in the position that we're in today and the kinds of customers that we're getting and the kinds of requests that come in through the door, already, the scope of the product has become a lot broader than it used to be.

[00:35:53] JM: What are the frictions in dealing with open source? They're totally not in the scope of what you're dealing with today. What are some untapped opportunities?

[00:36:01] KW: Yeah. Well, I think there's a whole universe of – There's a reality now, that if your building software, you don't own 80% of the code anymore inside of any app. At that stuff is coming in from third-parties. There's a question that all of these systems that we built to ensure

high-quality software at a timely rate when most of the code was proprietary, whether that model would still work out when most of the code is from third-parties and sourced from other places.

There's a whole interesting domain of managing software quality at scale when most of your stuff comes in third-parties that I don't think anybody's really figured out yet. We're still in the early days of taking a look at this and we're laser-focused on what we do today really, really well, which is things like compliance and security for open source dependencies.

There is this really interesting universe I think of code quality issues out there. The traditional models that we've seen that you'd apply to things like compliance and security issues breakdown very fast in the world of code quality because it's something so much more subjective. So we've seen a couple of efforts in the space to try to quantify it. I don't think anybody's really gotten it right so far, but I think if you're able to find a solution there, you can create a lot of value very quickly.

[00:37:11] JM: When you started Fossa, I'm sure you had some imagination for how the business would evolve over the – How long have you been? Three years? Four years?

[00:37:22] KW: Yeah. I have been working on the company for about four years now. High school career.

[00:37:27] JM: Amazing. How has the company diverged or done the same as what you envisioned at the beginning?

[00:37:36] KW: Totally. I think in many ways they're pretty deliberate about the business. We're deliberate about the funding strategy. Deliberate about what we we're building, about how we go to market. There were a lot of things that aren't really surprising. But this is my first company, and so there's a ton of stuff that I learned through the process about how to grow and operate and scale a business and what those things look like.

I think part of it is not being too prescriptive about the future. I think if you're in a good market with a good position, a lot of customers and you just kind of build the engine and run it, that engine produces a ton of directionality and a ton of activities and a ton of value on its own. A lot

of my focus has moved more from really trying to be careful, plan out and deliberate on the exact steps we take, need to take, to put to the future to just focusing on the right direction with the biggest opportunity and resourcing and building the biggest engine underneath it to take it over.

[00:38:33] JM: Well said. I assume you're not writing code these days.

[00:38:37] KW: I wish. That's a sad thing. I'm not anymore. I haven't touched code in probably about a year, and almost a year and a half I'd say. Very bittersweet thing for me, because it's one of favorite things to do, but it's not part of my role anymore.

[00:38:52] JM: Your day-to-day, is it hiring one-on-ones? All hands? What's the day-to-day like?

[00:39:01] KW: I do a lot of meetings. I do a lot of recruiting and hiring. I think finding the best possible talent is one of the most highest leveraged things that you could possibly do when building a company. We just done an offsite yesterday where we grabbed the whole team together and did an update with how the companies were doing, core metrics, all these other stuff, which is fantastic. I spent some time this week on that. I think day-to-day, a lot of it is just sharing ideas, talking to people and making sure that the business is performing really, really well.

[00:39:31] JM: How do you compete in these peak talent wars?

[00:39:36] KW: It's very difficult. I'd say hiring for us overall at the stage of the company, especially hiring technical talent, is probably easier than some of the other companies out there. We're building something by developers, for developers with all these interesting props inside of it and something good for the open source universe.

I think overall, we've had an easier time than some the other companies, but the reality is like, "Look. The things out there are going to just outcompete and outbid you." I think building a great developer brand has been awesome for us. A lot of folks that come in an interview with us already know about our products and know about what we do. At the same time, we started investing in other locales. Very soon we're going to have a Vancouver office, and the cost of

living there is much better adjusted. There's an amazing technical talent over there. It's also just a very beautiful city and it's in the same time zone as West Coast California.

I think overall, right now, startups have to be extremely open-minded with how they build their team and willing to be flexible in ways that very big companies aren't. If you can figure out how to you get that same sort of startup energy replicated in neither a remote team and a second office somewhere else, you'll have a much easier time.

I think the final thing was we knew recruiting was a muscle we had to build very early on the lifespan of the company, and so we hired a head of talent pretty early. I think it was sub – About 20 people or so is when we brought her on. Her name is Diana, and she's fantastic. There's so much groundwork that needed to get done just to build out the recruiting machine that she's put together, and now our hiring momentum is fantastic, it's because we made some of these investments really early on.

[00:41:17] JM: Tell me more about the organizational structure of the company.

[00:41:22] KW: Totally. So we got engineering products, and we've also got sales and marketing, and we've got GNA. We've got the kind of three branches of government that you totally expect.

[00:41:34] JM: GNA?

[00:41:35] KW: GNA. That's like general administrative, all the additional staff. It's awesome. We've been in a process of building our leadership team right now. We have an amazing head of sales. Amazing head of finance and functional leaders and tons of different sorts of areas of business. I think I focus a lot on the product and the team build and the hiring of the executive team right now. But it's constantly changing.

I'd say like our organizational structure today looks very different than like two months ago, which also look very different than what it looked like two months ago. I'd say that's still a living beast.

[SPONSOR MESSAGE]

[00:42:23] JM: Cox Automotive is the technology company behind Kelly Blue Book, autotrader.com and many other car sales and information platforms. Cox Automotive transforms the way that the world buys, sells and owns cars. They have the data and the user base to understand the future of car purchasing and ownership.

Cox automotive is looking for software engineers, data engineers, Scrum masters and a variety of other positions to help push the technology forward. If you want to innovate in the world of car buying, selling and ownership, check out cox autotech.com. That's C-O-X-A-U-T-O-T-E-C-H.com to find out more about career opportunities and what it's like working at Cox Automotive.

Cox Automotive isn't a car company. They're a technology company that's transforming the automotive industry.

Thanks to Cox Automotive, and if you want to support the show and check out the job opportunities at Cox Automotive, go to coxautotech.com.

[INTERVIEW CONTINUED]

[00:43:41] JM: Coming back to the engineering questions, there are a lot of engineering problems in this. I can imagine there's NLP problems, because you're scraping all these or polling all these open source repositories and you have to analyze them and you have to develop NLP systems for analyzing them. There're all these integration problems. You have to integrate with package managers and all kinds of other systems that are taking open source code and updating it and running it and so on.

These seem like large upkeep challenges. Do you dispatch specific engineers to manage those upkeep challenges or do you have like some system where your NLP is running, your package manager analysis system is running and it'll send you an alert if there's a problem?

[00:44:42] KW: Totally. You're right. We have a ton of data that we have to process in a very short amount of time, and especially if we're integrated in people's build systems, we have an

SLA with companies. The tolerance for downtime and the tolerance for long analysis times is very, very thin.

We've structured and architected the infrastructure so that the parts that have to move really fast are independent from the parts that have to be really slow, yet reliable. When somebody runs a CI build, a lot of the times we'll measure the performance of your system by how many times it can hit our cache of things that have already been analyzed versus how many times it actually has to go and analyze something new. That internal performance metric is very important to us just to help us make sure that every single time somebody wants some information about an open source package, we got it and we're already head of it.

There're a lot of things that we do to maintain the overall integrity and output of the product and the things that are really important to customers. Internally, we have like many companies in an on-call system. We have very senior sales engineering staff that interacts with our on-premise customers. We have a ton of instrumentation throughout the products so we know exactly it's working, what's not working, where computer time is getting spent throughout everything? The systems have some pretty hard lines inside of them. These systems are very well-architected for the specific goals that they actually have.

[00:46:11] JM: I'd like to know a little bit more about your infrastructure. You mentioned this, a caching system and there's obviously some kind of data pipeline that's running. Let's talk about the data infrastructure. We'll talk about data infrastructure and then we'll talk a little bit about the runtime. The data infrastructure, this system that's just enough scraping the repositories and pulling the package managers in order to be able to have some data to do analysis over, and then you have to do the actual analysis. Is all this just – Can you just scripts or like Airflow, DAGS, or do you need to use anything particularly complicated like Hadoop or something?

[00:46:52] KW: We run a lot of tools. The way the you can imagine, it's kind of like a distributed task queue, and the tools that we run to actually analyze the source code, analyze it for issues, licenses, all these other things are all tools that have different kinds of implementations. The license scanning part is implemented in one way. The security scanning part is implemented in another way. So we basically have to down the code and then inside of a container we run all these tools inside of it. Take the output and upload it back.

We have basically this huge queue, this huge pool of containers that we spin up and down to process these different revisions and versions of codebases and then we send all that stuff back.

[00:47:28] JM: You're talking about customers or you're talking about the open source repository?

[00:47:32] KW: This is basically for – It's actually for both. So there's sort of like a part of our infrastructure that handles proprietary code and helps customers figure out what is inside of a given unit of proprietary code. Then there's also separate part of the system that is also like a big distributed task queue that analyzes open source and third-party code. Then when we're done chunking through all these stuff in real-time, we send it all up to a single Postgres instance or a Postgres cluster. We leverage the hell out of Postgres. Postgres is such an incredible piece of software. We've really gotten our mileage out of it.

[00:48:11] JM: Right. It sounds like none of the individual compute tasks require so much data that you would have to use Spark or something.

[00:48:21] KW: Generally, no. I think many open source projects are relatively small. When I say relatively small, I mean, it's rare to find open source project that's in the gigabytes in terms of code size. We're dealing with high-volume of projects, but probably relatively low levels of actual code size. We deal with proprietary code that looks very different. Proprietary code can be gigs upon gigs, especially in large enterprises, you can have these huge monolithic codebases that a single developer can't even load on its machine and build.

So they have to load like kind of like a partial chain set, chains of chain set. Send that chain set through like a build train into some proprietary production build infrastructure and then you get feedback like a few hours later. Building a system that can handle that level of scale and code looks very different than building a system that can handle a bunch of high-volume open source packages.

[00:49:18] JM: Had there been any management dynamics that have been difficult for you to master?

[00:49:23] KW: I think people are way harder than computers. I think that's one of the things I miss about programming. Programming is very easy I think compared to people. Totally. I mean, the past couple of years have been an amazing learning experience for me personally on how to build and grow a team, and this is not only my first company, but I dropped out of college to work on this thing. I'm a noob on so many different levels, but this has been an amazing experience for me. I'm really proud of the team that we had built today.

I would say the people at Fossa make it really, really easy for me to play my role, because we just have an amazing group of self-motivated, super positive, solution-oriented people in the office that are constantly having a huge amount of appetite to solve problems and also have a huge emotional maturity to deal with all the ups and downs inside of a startup.

We've got a great leadership team today. We've got a great overall staff inside the company. We have an amazing energy and the business is performing exceptionally. Times are really good right now. There are times where managing company is harder than others. But overall, it's never been I think a situation where I felt ever distrustful of any of the motivations on the team. Distrustful or sort of negative on any of the intentions from anybody, and the people have just been an immense pleasure to work with and learn from.

[00:50:46] JM: Had there been any – This is more of a company management question. Have there been any SaaS tools, like newer SaaS tools that have surprised you and how useful they are?

[00:50:56] KW: Oh my God! Yeah. I'm very bullish on the realm of customer success. I think the world of CRM and new business acquisition tools has really dominated and gotten a huge amount of attention from the technology industry. The world of business retention tools is not only I think more complex in a lot of ways, but also just has gotten less attention and less investment from the tech community than the former. The reality isn't in any healthy business, eventually your recurring business is going to be providing more value to the company than the new business.

This is usually a much later stage, and then where we're at or where many companies are at. But it's very common for a company to get out the gates, close a ton of new business. A year later or two years later, hit a bunch turn with the initial customers and then just die or get sent back immensely.

We've tried to really get ahead of this energy by investing in customer success tooling and instrumentation and headcount and teams very early on to lifecycle the company. We've hired an amazing director of customer successor early on. They brought in a tool called Client Success, which is produced by a pretty small company. It's still a startup, but it's an awesome tool. It's got these pulse checks, graphs. It gives you this amazing visibility into how every single company and client is doing in your customer stack. You get sorted by dollar ARR.

You can see every single email and communication sent to them all on this one dashboard. It's great. It really shines for an enterprise software company like us where there communication with customers can be really distributed. It can be done through the support person on our team, engineer, a sales engineer in-person, our customer success people, our product people. Being able to see that all-in-one place coalesce gives you a really cohesive story of the entire end-to-end customer experience. Now you have this like incredible air traffic control dashboard of how your customers are doing.

This is something top of mind to us. It's enabled us to be extremely customer-focused, and as an agenda of every single weekly or biweekly product meeting, we pull the customer success dashboard. See all of what the customer needs. Have all [inaudible 00:53:12] tickets integrated with it. It's just awesome. It's called Client Success, by the way. I've raved about them already on Twitter. They're an awesome startup. They're going to do great. I highly recommend checking them out.

[00:53:24] JM: I will check that out. But I would imagine, Fossa is not one of these tools that churns a lot, but I guess it requires some upkeep from the user.

[00:53:36] KW: I would hope so too. I would imagine it so. I mean, our numbers are looking good on that. Generally, your customers are very happy, and once we're in there, we're

constantly providing value over the part of their process. We fairly very rarely run into the situation where a customer integrates us and says, “Okay. I don't see the continued value of this product.”

It's a pretty stable tool. It’s a pretty stable product overall and it plays a pretty critical function I think in a lot of the companies that we integrate with. I think a huge priority for our team is minimizing the amount of days it takes to actually get us distributed to the point we’re in that posture in the company. The focus of a lot of our customer success work has been like hours within signing the purchase order and inking the agreement. How quickly can we mobilize all our resource to our customers deploy it and get the tooling, providing value to them?

[00:54:30] JM: What's the – I mean, I know you’re a creative business thinker. What's the coolest idea for a software business that you have been unable to pursue because you're too busy running your open source company?

[00:54:42] KW: I was just thinking about this client’s – Honestly [inaudible 00:54:45]. Seriously, I saw this tool and I’m like – I have a lot of ideas. I actually think that many of them are not venture scalable businesses or maybe not businesses that are things that I would go and try to hire hundreds of people for and go create this massive starship. One of them is actually this sort of like customer success stack.

Eventually, at some point, I think it’d would be really cool to build a really deeply integrated and really sophisticated customer success platform.

[00:55:17] JM: AWS for customer success.

[00:55:20] KW: I sort of think about like Auth0 for customer success. I think the ways that you actually have to do it is it can't just be kind of like a high-level relationship dashboard. You have to have some sort of deep roots or integrations into the actual product. If we think about the problems at like a product inside of enterprise might have towards adaption, there’s a lot of really, really similar people and process related things. You need a team to adapt it. You need to keep them updated. You have to push their agenda.

A lot of times when you get into a big company, it's hard to create a huge map of who the key stakeholders actually are. What full adaption actually looks like? What success looks like in that team? There're all this just like recurring related work that comes with every single account, and every single software product usually has to not only get their internal teams really sophisticated, but build these tools directly into their product help enable their teams to get full penetration.

I'll give you a couple of examples on this. Every single company has to integrate with an SSO provider. For you to even think about how much coverage you have, you have to know, “All right. Inside of one big company with 10,000 employees. So we sell to the engineering team.” How big is that engineering team actually? How many teams are there? Which team should we target first to get the most amount of adaption? What do those teams actually care about? We need to find an SSO provider or some sort of employee directory.

Then at that point, you have to figure out how to invite them and how to onboard them or build some sort of onboarding workflows into the team. This exists with any process side of any single company, not even from a vendor's perspective, but like say I want to roll out – I work for like, say, a huge company like a GE or a Ford or something and I want to roll out a big process or some tool I developed internally. I got to do the same kind of advocacy there.

I think what it would look like is it would look like a bunch of these kind of like SDKs or sort of bespoke tools that you can actually build in to the product that provide connectors for all the tools that you'd see in the enterprise; Slack, messaging tools, like GitHub, GitLab, if you care about data for many of those tools, user directories, ELDAP, etc., and then building a layer on top of that where then you can now build onboarding workflows. You can build some sort of outreach campaign and run these campaigns internally inside of enterprises to get better adaption or outcomes for a tool or something.

I think that's what the platform looks like. It's sort of like a step a little bit lower level than something like a client success, something like this overall dashboard. But it isn't so technical that you're like writing some language or something that deep in the stack, but it’s something that can be instrumental with an existing product or initiative and it does a lot of the leg work just

to get people and process changes finished in order get some process, some tool something successful.

I want to build that. I don't actually know how successful that's going to be. I haven't thought that much about it. I faced the pain point. So I feel strongly about the value it would provide to me. I have kind of a feeling that if this business were actually turned to something scalable, it might not look like what I just described. It might look like something either lower level than what I just described or even like higher-level where it's not just vendors. It's for like – A business model of selling something to vendors to implement something, it's okay and customer success departments still don't have as much executive support as a sales department. So it's questionable how much value you can extract.

But it might look something like, "Oh hey! This is a general tool for all enterprises you can sell directly to have them get more value out of all of the people and process things they want to deploy." I don't know. I have no idea what this business looks like, but that's one thing I've been thinking about a lot.

[00:59:00] JM: Well, I'm definitely going to asking that same question the next time I interview you, because I'm sure you'll have another good answer. But just to tie it all together, see how easy it is to figure out an adjacency to your business?

[00:59:12] KW: Good. Good.

[00:59:13] JM: Kevin Wang, thanks for coming back on the show.

[00:59:15] KW: Thank you so much for having me. I appreciated it.

[END OF INTERVIEW]

[00:59:26] JM: If you want to extract value from your data, it can be difficult especially for nontechnical, non-analyst users. As software builders, you have this unique opportunity to unlock the value of your data to users through your product or your service.

Jaspersoft offers embeddable reports, dashboards and data visualizations that developers love. Give your users intuitive access to data in the ideal place for them to take action within your application. To check out a sample application with embedded analytics, go to softwareengineeringdaily.com/jaspersoft. You can 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's not just your company that wants analytics. It's also your customers.

In an upcoming episode of Software Engineering Daily, we will talk to TIBCO about visualizing data inside apps based on modern frontend libraries like React, Angular, and VueJS. In the meantime, check out Jaspersoft for yourself at softwareengineering.com/jaspersoft.

Thanks to TIBCO for being a sponsor of Software Engineering Daily.

[END]