EPISODE 991

[INTRODUCTION]

[00:00:00] JM: Chat systems have been a part of software development for decades. Older systems like Pigeon and Yammer were surpassed by newer systems like HipChat, and when Slack was created, it quickly became a part of most software companies. But Slack does not fulfill the needs of every company.

Mattermost is an open source chat system. Mattermost can be configured to work within enterprises that have strong constraints around compliance and data governance. Whereas Slack is a SaaS product that requires users to send their data to the cloud servers managed by Slack, Mattermost allows the enterprise to design how data moves through services and where the databases are hosted.

Ian Tien is the CEO of Mattermost and he joins the show to talk about why many companies need their chat system to be hosted in a private cloud or on-premises. In a previous episode with CTO Corey Hulen, we discussed the engineering behind Mattermost. In today's episode, we explore the management and strategy of the business as well as some additional engineering since Ian Tien's background is as a software engineer and computer scientist.

[SPONSOR MESSAGE]

[00:01:14] JM: Today's show is brought to you by Heroku, which has been my most frequently used cloud provider since I started as a software engineer. Heroku allows me to build and deploy my apps quickly without friction. Heroku's focus has always been on the developer experience, and working with data on the platform brings that same great experience. Heroku knows that you need fast access to data and insights so you can bring the most compelling and relevant apps to market.

Heroku's fully managed Postgres, Redis and Kafka data services help you get started faster and be more productive. Whether you're working with Postgres, or Apache Kafka, or Redis, and that means you can focus on building data-driven apps, not data infrastructure.

Visit softwareengineeringdaily.com/herokudata to learn about Heroku's managed data services. We build our own site, softwaredaily.com on Heroku, and as we scale, we will eventually need access to data services. I'm looking forward to taking advantage of Heroku's managed data services because I'm confident that they will be as easy to use as Heroku's core deployment and application management systems.

Visit softwareengineeringdaily.com/herokudata to find out more, and thanks to Heroku for being a sponsor of Software Engineering Daily.

[INTERVIEW]

[00:02:47] JM: Ian Tien, welcome to Software Engineering Daily.

[00:02:49] IT: Yeah, thanks for having me.

[00:02:50] JM: Internal chat has been a product that almost every software company has needed for the last 20 years, but it was only around 2014 that the chat products like Slack and Mattermost started to appear in prominence. Why did it take until 2014 for that product category to exist?

[00:03:12] IT: Yeah, that's a great question. I don't know. I think what happened around that time you described is that you had a generational shift of human beings that were spending a lot of time on Facebook and social media and communicating through digital channels just became a lot more natural. What you saw was a rapid decline in the actual use of email and a generation of people that really relied on messaging, especially mobile messaging. For a lot of millennials today, they think of email as password reset and where I get letters from my parents.

I think that's what's really driven something that has always kind of been there, which is whether it's IRC or it's been instant messaging. It's really taken something that has always kind of existed and been sort of like a secondary system and made the primary just because of how people have been growing up, and that's a one-way door, because it's really built on the experience of another generation of users.

[00:04:12] JM: HipChat was the first product to market in this category. Why didn't HipChat take the entire market?

[00:04:21] IT: Having wonderful friends who've worked very, very hard o that product, I would say that –

[00:04:25] JM: Great product by the way. I used it.

[00:04:27] IT: We did as well, it's actually part of the formation story of Mattermost, is that we were such heavy HipChat users when it got bought by a larger company and really took a backseat product-wise. It was a very noticeable decline in product quality. We would lose data. It would have bugs. It would have outages. It would be usable at points and we were so frustrated that we actually tried to leave that product and it wouldn't let us export. We had 26 gigs of data and they really weren't designed for that, because we were a video game company.

When we stopped paying our subscription, they pay walled us for our own information. We couldn't actually access what we put in there. We didn't think that was the model that we really wanted to engage with. What we did was we took some of the software that was in our video games. We're a video game company before that are about 10 million hours of usage and we created our own platform to do this messaging. We open sourced it and that kind of turned its own thing.

But I think that HipChat was really popular. It was on this wonderful track. People didn't understand how powerful it could be with integrations and with the right sort of workflows. I think that what you'll find in these categories, it's going to continue to evolve and there's going to be many, many products like this out there.

The biggest one is Discord actually, right? If you look at the number of users on Discord versus Slack, Discord is actually much larger and they're kind of the same thing, Mattermost, Discord, and Slack, and HipChat all have a very similar sort of channel-based user interface. I think that system, it's kind of done. There's a way to do it now. Now that question is, "Well, do you want to use a close source SaaS service or do you want to use an open source service that you have access, and where in the market is that going to go?'

We think where it's going to go is, "Hey, you've got operating systems. They're all open source now. You've got databases that are all open source now. You've got virtualization infrastructure. The leading ones are open source right now." We think collaboration is going to go the same way.

[00:06:20] JM: We interviewed your cofounder, Corey, and he talked about some of the ways that Uber uses Mattermost, their fork of Mattermost, and that is a highly customized implementation of a chat system for their specific workflows. What are the features around chat system that can be customized for specific workflows?

[00:06:48] IT: Yeah, that's a great question. I mean, if you want to get really specific and technical, it's really like what do you open up at the API level? With Mattermost, the extreme case is you fork Mattermost to do the certain things. Overtime, we are about how do we architect the system where it's really not necessary to fork, right? Where we've been able to put like our web applications, our desktop applications. How do we separate them modularly? That's one. You can sort of like modify rather than fork.

The other one, and here's different mechanisms to do that. Allowing the API support with people needs. What people really want in some cases, they want a canvas. I want to run arbitrary JavaScript. Well, in a multitenant system, the answer is no. But in a single tenant system where you can have a plug-in architecture that says, "Okay. You're going to sign up to do this because you're the admin. You're going to allow this," then you can get really, really powerful. Yeah, you can and install buttons and have certain patterns, but if you want to create your own micro-application within a custom message type inside a platform, that's where we think the whole market wants to go, right?

I think the more developers we talk to, they're like, "Okay, what messaging platform are you using?" They always have, "I don't like the sound effect. I don't like the way that it threads messages. I don't like this or the other piece."

What open source is, it's a platform for innovation. It's openness, it's transparency, it's experimentation, it's iteration. We believe that long-term, these challenges and these needs these sort of like micro- markets are really going to be served by an open platform. We're talking about APIs. You've got custom message types. You can input your own sort of server JavaScript applications within them. You've got bots that can work on the platform with different types of security and permissions and controls. You've got lots of customization you can do with our own frontend's.

All the Mattermost experience, the desktop app, the web app are going through same APIs as those APIs are open. If somebody can fork our frontends and say, “I want to do the web app a different way.” People can actually build brand-new ones. One of the most popular examples there, like someone's –We've got to Matterhorn, which is a terminal client for Mattermost written in Haskell. People have created the pigeon interface, a frontend to Mattermost. It's just kind of a couple of small examples.

[00:09:05] JM: Pigeon, that old chat client.

[00:09:07] IT: Yeah, and they were just like, “Hey! We have users that need Pigeon,” and like, “This archaic backend is going to be end of life. Can we use Mattermost as a frontend?” Someone just, “Hey, I think you can,” and they wrote it, and they documented it and there's install instruction. If you want to web search Mattermost Pigeon, you'll be able to find one of these sort of extreme cases of, “Yeah, let's customize.”

[00:09:30] JM: You got any more niche examples of the – you just mentioned, the Pigeon interface and somebody writing a terminal that you can embed into Mattermost? What else do you got?

[00:09:41] IT: There's hundreds. You can go to GitHub and search the word Mattermost and you'll just find all the open source projects, hundreds and hundreds of them doing very different things. Creative, there's this application where someone can feed their cat remotely using Mattermost. You'll see a Mattermost hackathon. We have a Mattermost hackathon running right now. There's a lot of creativity that comes out of there. Like the last one we ran, you can just search a month or two ago, Mattermost hackathon. Just search for those. There's like two hours of demos of just all these creative things people are doing.

[00:10:12] JM: Is open source important to that kind of customizability? Couldn't you just have API hooks or chat bot systems?

[00:10:20] IT: Open source if fundamental to building a developer community that is creative and energized in ways that only open source can create that. It's transparency, right? It's like, "Oh! Your API doesn't support that. Too bad. I guess someone built this thing." But it's transparency not on the codebase saying that like, "Hey, this is how much code we have to change for that API to do what I need." It's also open source in our ticketing system and our pull requests that say, "Hey, here's what's coming down the road." We have people who are like, "Oh, I can't use Mattermost until this feature is in. We're not going to talk to you guys."

Then out of the blue, they come back. It's like, "We saw the ticket we were maundering being like closed. Therefore we know that this feature that we've been waiting for forever is going to be like coming in the next like month, in the next release cycle." They're that far ahead. That level of transparency is I think fundamental to what I think is it's table stakes right now to the best developers our there contributing.

[00:11:18] JM: The market leaders in this category today of enterprise chat, you've got Slack, Mattermost and Microsoft teams. How does the market segment around these different products?

[00:11:30] IT: I think there's sort of three segments in enterprise messaging. You've got the first mover, which is going to be Slack. It's sort of the best known. It's got a big brand. You have another category, which is incumbents, and you have Microsoft teams are the best known one. There's other sort of legacy communication providers that are sort of throwing their hat in the ring.

The third segment we think about is open source, and Mattermost is really defining that segment and leading it in a very special way with we've raised 70 million dollars of funding. We're really earning the trust of enterprises. We're building a very large community, over a thousand contributors. We're really aligning to the open source DevOps community out there. Hey, if I want to do collaboration and I believe my enterprise is going to be focused on software innovation as one of the critical priorities of its strategy, then I want to think about DevOps and open source and how do I make that more real-time and have faster cycles, and that messaging platform is part of that initiative.

[00:12:32] JM: From your experience working at Microsoft, you used to work there, what's the core competency that Microsoft can bring to this product category? Is there something that's particularly well-equipped to doing to compete?

[00:12:49] IT: Being ex-Microsoft, our super power was really pre-integration. Pre-integration is that, "Hey, if I need to solve a problem in SharePoint, well I've got the SQL server team on the database end. I've got the Windows server team. I've got like the browser team, if anyone uses Microsoft browsers," in theory.

But you have pre-integration and you have a lot of leverage because you have transparency over your entire stack, right? Then you could innovate with the entire stack together. That's Microsoft's historical advantage. The disruption that open source can create is very material because it offers a same thing. It offers transparency through the entire stack and it means that if I want to build something, I can actually go through the entire stack, my platform, and I can work to align all those together.

The funny thing is when you're at Microsoft, it's kind of like the same amount of effort. It's not the same, but it's like you've got to go to those other teams and you got to convince SQL server to do it this way and you have to have all the meetings. The open source world is the same thing. You have to be able to align all these pieces together.

But when you align things in the open source world, you'd like at cloud native, you look at a lot of the things that are doing it large scale. It's extremely powerful because it changed the whole world. It's that same concept if you've got stacks and transparency and you want to influence and Microsoft super power is that pre-integration, and open source, it's the same.

[SPONSOR MESSAGE]

[00:14:16] JM: Looking for a job is painful, and if you are in software and you have the skillset needed to get a job in technology, it can sometimes seem very strange that it takes so long to find a job that's a good fit for you.

Vettery is an online hiring marketplace to connect highly-qualified workers with top companies. Vettery keeps the quality of workers and companies on the platform high, because Vettery vets both workers and companies access is exclusive and you can apply to find a job through Vetter by going to vetter.com/sedaily. That's V-E-T-T-E-R-Y.com/sedaily.

Once you're accepted to Vettery, you have access to a modern hiring process. You can set preferences for location, experience level, salary requirements and other parameters so that you only get job opportunities that appeal to you.

No more of those recruiters sending you blind messages that say they are looking for a Java rockstar with 35 years of experience who's willing to relocate to Antarctica. We all know that there is a better way to find a job. So check out vettery.com/sedaily and get a $300 sign-up bonus if you accept a job through Vettery.

Vettery is changing the way people get hired and the way that people hire. So check outvettery.com/sedaily and get a $300 at bonus if you accept a job through Vettery. That's V-E-T-T-E-R-Y.com/sedaily.

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

[INTERVIEW CONTINUED]

[00:16:05] JM: The position of Mattermost is not solely around the customizability. There's also the matter of data ownership, similar to the situation that you described with HipChat earlier. There's the idea that a privacy-conscious organization would want total control of their own data on their own servers. I recognized that there are companies in the market that have a desire for that. There are companies that want to control all their own data, whether it's data that's part of their application infrastructure or part of their chat communications.

But taking a step back, how rational is that? If we're talking about who to trust with your data, wouldn't an enterprise be better off trusting a large SaaS company or a large cloud provider? Why do these enterprises want to take matters into their own hands?

[00:17:04] IT: Yeah, it's a great question. It's not that these enterprises are trusting their data. They're trusting their customer's data and they're trusting information that really shouldn't be shared publicly. Security vulnerabilities are a great example. If I'm an enterprise and I have a security vulnerability that's going to affect – You see this all the time. It's going to affect all my products, right? Well, if that security vulnerability got disclosed outside of my company, then I've really put a lot of my customers are risk. Where am I okay sharing that information?

Now, imagine that you're trusting a company with your data, right? But if they're in turn trusting other companies with your data and it sort of passes down the system, then it's the weakest link in the chain. That's the issue. To have trust, trust is how do I expose myself to vulnerability based on the positive expectations of someone else, right? When you're a customer and you work with an enterprise, what you say is like, "I trust you and here's what you owe me in terms of that trust." It's really hard for that company to say, "Oh! In that contract, you know what? I really have these other SaaS provides and you have to agree to make it okay for me to use all those SaaS providers for my stuff." The customer basically says, "No," right? Because they're an enterprise too.

That's where the rubber hits the road, is that people who take those customer agreements very seriously have to honor that customer's data, or honor the risk that could be created when sort of secretes, when vulnerabilities, when confidential business information gets leaked out. You'll find with Mattermost, it's arranged with those people who really care about these agreements, right?

It goes from large financial services institutions, to government entities that are tasked with huge amount of responsibility, to actually some of the bodies that are governing these privacy laws. They're using Mattermost. It also goes to animation studios, right? People who have contractual agreements about really critical IP that they're producing, right? Like animated movie, if spoilers come out, if information comes out and it gets disclosed, there's a tremendous amount of economic lawsuit that could happen there. It's really about other people's secrets and how do you ensure that you're managing those properly.

[00:19:25] JM: It's not necessarily about some law or compliance issue, it's more these companies just don't want their data escaping the control of their supply chain, their data supply chain.

[00:19:43] IT: Yeah. Basically it's both.

[00:19:46] JM: There are customers that are like, "We don't want to use a SaaS service because it doesn't fit out compliance schema."

[00:19:55] IT: Yeah, it's really about like what you're really doing is you're entering a relationship – I think in the SMB space you're kind of like, "I swipe my credit card. I click the terms of service. I'm done." Right? In the enterprise space they're like, "What is the market cap of that company? Show me its balance sheet before I put my data in that thing. You got to prove to me that this is real."

You know what the funny thing is? I think the people that run SaaS services are the ones that trust SaaS services the least, because you realize – Just some examples is like, "Hey! Oh! Username, password." You'll see these thing, and if you've done M&A on small SaaS companies, it's like, "Oh, yes. We salt and we hash the passwords." This company that was in one acquisition that I've seen, yeah, they salted and hash the passwords and then they also stored the passwords in clear text.

[00:20:40] JM: Just being redundant.

[00:20:42] IT: Yeah, just in case. I mean, people who build these SaaS services, it's not by anyone's fault, right? Look at what happened to Facebook, right? They had logging. They had keystroke logging, and, "Whoops! Guess what? That turns into a clear text password store." It's not malicious. It's by accident.

Here’s the thing with SaaS services. SaaS services are made by human beings. It’s like Conway’s Law. You’re shipping the org structures. Guess what? Human beings are fallible. Right? Yes, SaaS services, there's people behind them, which are really trusting in a SaaS services, which is trusting the engineering team that built it.

[00:21:18] JM: Again, you are trusting an engineering team whose existence is based on their ability to keep things secure. I mean, they’re as existentially threatened by a data risk as you would be if you're a bank that chooses to run your own chat service, right?

[00:21:41] IT: The differences with open source, you can see all the source codes. It’s like, “I’m going to trust the source code,” right? There’s a bunch of people who wrote it, but I’m going to trust the source code and I’m going to trust all the people who’ve been paid to try and break the source code,” right?

At GitLab, they spent $1 million on HackerOne trying to break and find security issues inside the product, and anyone can expect that source code, right? That source code people trust. How much trust do we put in a Silicon Valley company? The average tenure of a Silicon Valley engineer is what? Like two years.

By the time they start their job, they’re already looking for their next job. How much trust are you going to put – I mean, there's many wonderful services, but like think about from an enterprise point of view. Am I going to trust a Silicon Valley company where the engineers are turning over consistently or am I going to trust an open source project where millions of dollars are being spent and thousands of thousands of eyeballs just like mine in this enterprise?

You run the system through black deck. You run them through a lot of – You check libraries. You can do all that in a complete open environment. The arguments from 20 years ago that, honestly, like Microsoft is putting out there. I worked there, right? You would create this marketing collateral and you say, “Why Windows server is much more secure than Linux?” and all these arguments. Today, Azure runs more Linux than it does Windows server.

[00:22:58] JM: Right. That said, don't security vulnerability usually come from misconfigurations and operator mistakes rather than bugs in the source code?

[00:23:10] IT: I think that every system has vulnerabilities and it's really about what's sort of combination of those bot and vulnerabilities are open. It's about what are the PaaS that you can find into them? We live in a world where after meltdown inspector, people don't even want to run their virtual systems on the same chips. It's really about vulnerabilities. I'd say that it's really about the vulnerabilities.

[00:23:31] JM: I admit there are definitely security risks to both the code and the configuration of your databases or whatever. We've done more shows on Slack than Mattermost, and thing I remember about Slack is that they had trouble scaling up the application to where they could have a really high number of concurrent users. There was some hard limitations for some time. There were times when you could not have more than 80,000 users in a Slack channel. They eventually overcame that. I'm assuming that you've encountered similar bottlenecks to building really, really large chat rooms.

I just use that as a jumping off point into a conversation about engineering. What are the core engineering challenges to supporting a chat system that has tens of thousands of concurrent users?

[00:24:23] IT: Oh, it's a great question. Tens of thousands of concurrent users. You'll hit all sorts. You definitely have the scaling issue. You have tradeoffs between, "Hey, there's a really cool feature," and "Hey, I really want to scale this room." You'll have to make choices between some features have to turn off after a thousand users, right? Then you also want to think about – There's sort of the technical scale, and then like, "Hey, you've got to write pump."

One thing that we didn't expect in some of our largest deployments is that when you test the number of messages you can send, people can only type so fast, right? So then you've got one message and it broadcast to like tens of thousands of users. That's okay, but we only expect so many. What happens is someone makes a post and people love it and they start doing emoji reactions. When they do emoji reactions, they go really fast and they go in parallel, and which you actually have is a DDoS attack created by emoji reactions, because you never anticipated that number of – That many sends that quickly, right? Each one of those sends. So, you batch it. You do different things. Then that's the engineering challenges.

Then you’ve got your UX challenges, which is, I mentioned someone with 20,000 people in the room. The list is too long and there’re too many identical names. How do I handle that? When I’ve got tens of thousands of people in the system, my sidebar goes nuts, because there are so many channels. How do I think about these people that want to push the envelope and they say, “Hey, I want to be able to group these channels. I want to have this sorted in a certain way.”

I think there’s sort of next level things when you work in large, large organizations and there are many design choices you want to make. Here’s the thing, people want that custom. People want to tailor that experience to what their users need and how their workflows happen. That I think is where the market is going. It’s really like, “Hey, I’ve got this system. It works, but there’s all these pain points that come at scale.”

If you don’t set things up correctly, you’ve got engineers who are constantly pinging by things. They’re like, “Oh my God! There’s like tens of thousands of people here. All the people have the same question and they’re going to like swarm this one engineer with one question,” right? Because like, “Oh! I can really reach that person.” Then they get blasted. I think that categories are, “Hey, what are the technical challenges and then what are the UX challenges?”

[00:26:29] JM: Explain how the Mattermost engineering team is structured.

[00:26:32] IT: The Mattermost engineering team. First of all, we’re rote-first first, right? All different time zones all around the world. When possible, we try to group them by time zone. It just makes collaboration a little bit smoother. Then there’s going to be sort of functional groups, right? Whether it’s on integrations or it’s on end users. There’s going to be sort of themes that people work around.

Then we’ve got product managers that help do the planning and the customer research and specifications and overseeing how everything works together. Generally, it’s very similar to like the two pizza teams sort of a pod structure, but really rolling up to sort of R&D leadership.

[00:27:06] JM: Since Mattermost has to run on-prem, you can’t use any off-the-shelf cloud tools. How has that affected the development process for the company?

[00:27:18] IT: When you say we can't use any cloud tools –

[00:27:20] JM: For example, you can't use EC2, or RDS, or Amazon Kenesis, or whatever. Most of the companies that I do interviews with are like built entirely on the cloud. I just thought it'd be an interesting opportunity to talk about how you build the infrastructure in a way that it is completely open source and deployable and scalable.

[00:27:41] IT: Oh, that's a great question. We actually have support for AWS, for Aurora, for S3, and we also have support for the open source alternatives, like MinIO [inaudible 00:27:53] S3. It's really about being customer-centered. Then self-hosted.

There's the word on-prem, but how we think about, it's the way you self-manage. If you want to self-manage on AWS, that self-management as well and people want to do that and they'll contribute code and then we're like, "Great. Now it runs on S3."

[00:28:12] JM: So you more like say, okay, you need a relational database here. You need a bucket storage here. You're welcome to use an open source thing or Amazon S3 or whatever.

[00:28:24] IT: Yeah, you can do – There's sort of like small – And it's really about talking to our customers always and saying, "Oh! You just want to run this really small? Great. You can just put it in a folder on that one note." It's like, "Oh! You actually need a network – You need a NAS. Okay. Great. You have a NAS." "Oh! Actually you probably need an S3." "Okay. Great." "Oh! You want to complete the data center? Okay. Well, then here's MinIO." It's about documenting everything and really understanding, "Okay. For this situation, this is what I need." Then having it available.

That's been a tremendous benefit for a lot of enterprises who say like, "You know what? I really want something as vendor neutral. I really want something that's going to be flexible, because every vendor right now, they don't have a cloud strategy. They have a multi-cloud strategy."

[00:29:07] JM: Right. What's the biggest technical problem that you've solved in the last year?

[00:29:11] IT: Biggest technical problem we solved in the last year. I don't know actually. I think that'd be – Because I only see the solutions. I don't see how deep the problems are. I think one of the ones that kind of comes to mind is just mobility, right? How do you satisfy all these security requirements around mobile? You do it in a way that's very performant. It's just sort of like a lot of things you got to get right to have mobile to be secure and to be thoughtful and to be usable by the end customer. I wouldn't say that's fully solved.

I think that's definitely a work in progress, but I think we've made probably more progress on enabling enterprises to create really private mobile experiences that connect with their DevOps infrastructure than anything else we've seen on the market.

[00:29:54] JM: Has Kubernetes significantly affected how you run Mattermost?

[00:29:58] IT: Absolutely. You can Google Kubernetes Mattermost and you'll find our Helm charts and how we set things up. It just makes things a lot simpler and it's kind of where the market is going. A few years ago we're like, "Oh my gosh! How do we support this and this and this and this and this and Puppet and Chef and this?" There's a lot of open source projects out there that do support Mattermost there, but now because things are converging a little bit, there's a clear direction, and you can do a lot more when you have clarity on the platform and the direction.

[00:30:25] JM: We're at GitLab Commit, which is the GitLab Conference. What's the connection between GitLab and Mattermost? How do you see these products comparing in terms of their customer-base and their go-to-market?

[00:30:40] IT: Yeah. Inside of GitLab, you actually have Mattermost ready to go. Our binary is in there and if you just search for GitLab Mattermost on the web, you'll find the install instructions. Say, here's my endpoint and you say, "GitLab reconfigure," and, "Boom!" You got Mattermost running. All the core features of Slack, Slack keyboard shortcuts, can work with GitLab pieces. I think that's one. Actually, in one system it's the same.

I think when GitLab talks about concurrent DevOps and the ability to say have one application across the entire DevOps stack, Mattermost really takes the power of GitLab and it makes it real-time. Now I've got this across my mobile devices. I've got aggregated information through the GitLab bot. I've got the ability to go to mobile. I've even got the ability to go to my smart watch, and I can get notifications there. I can reply with a voice to text, and it really takes things to the next level.

[00:31:32] JM: I'd like to get your perspective on going to market. Can you start by just telling me the first significant sale that you had with Mattermost and how the early sale process got defined and how that eventually snowballed into what kind of go-to-market strategy you have today?

[00:31:55] IT: Yeah. I got them. I'm trying to figure out how to say this. I would say – Our first significant sale was with a manufacturer in Silicon Valley. Just when we're starting, we just started our commercial version and this manufacturer like, "Hey, we really want this. But we're going to pen test you." They ran it and they had their red teams on it and they found seven issues in a few days. We're ex-Microsoft, so we fix them pretty quick and they're shocked because like open source projects don't do that.

We kind of fixed them immediately and they're like, "Oh! Well, okay. We'll buy something." The procurement process was argues and long and not great. It was supposed to be, but we actually have to champion that, like push us through it. We were sort of brand-new. We were engineers from Microsoft, not like sales people. We were able to kind of take this technology that they really wanted. We're able to add the key enterprise features that they needed, which was single sign on and a certain flavor of high availability and they're very willing to pay for that and they were sort on excelling in the process and pushing us along.

What we found is that pattern just kind of repeated itself, right? The enterprises come in. They're like – They have transparency. The engineering team see it. It's like, "We like what you do. But you got to just list the features."

One of the funny ones was like – Okay. There's pen testing. There's security stuff. There's single sign on. There's like, "We need custom emojis." That's kind of like one of these things – One of these things don't fit. It's like, "Well, why do you need custom emojis." They were embarrassed to say this, but like they have a build brick, they have a custom emoji with the head of their CEO. That's like the symbol for a build break and they just wanted to carry that on.

Yes, we added custom emojis and it’s actually fascinating. We actually now – In our CICD instance, we have like 4,000 custom emojis as like the load test and like every day we run them, because going back to scale discussion, this is what enterprises need, and this is what they don’t get out of the open source. It’s very sort of like okay to put in the enterprise version, because anyone who’s running it at that scale obviously is an enterprise and obviously is getting a benefit out of it.

Working with enterprises in the early days of our business has really been a partnership and they see the open source code. They’re appreciate of it. They understand we are a company and we do have a revenue model. What they really give us is feedback for where they want the product and the platform to go and we want to be one of their trusted vendors to build in that direction.

[SPONSOR MESSAGE]

[00:34:32] JM: Apache Cassandra is an open source distributed database that was first created to meet the scalability and availability needs of Facebook, Amazon and Google. In previous episodes of Software Engineering Daily we have covered Cassandra’s architecture and its benefits, and we’re happy to have Datastax, the largest contributed to the Cassandra project since day one as a sponsor of Software Engineering Daily.

Datastax provides Datastax enterprise, a powerful distribution of Cassandra created by the team that has contributed the most to Cassandra. Datastax enterprise enables teams to develop faster, scale further, achieve operational simplicity, ensure enterprise security and run mixed workloads that work with the latest graph, search and analytics technology all running across hybrid and multi-cloud infrastructure.

More than 400 companies including Cisco, Capital One, and eBay run Datastax to modernize their database infrastructure, improve scalability and security, and deliver on projects such as customer analytics, IoT and e-commerce. To learn more about Apache Cassandra and Datastax’s enterprise, go to datastax.com/sedaily. That's Datastax with an X, D-A-T-A-S-T-A-X, @datastax.com/sedaily.

Thank you to Datastax for being a sponsor of Software Engineering Daily. It's a great honor to have Datastax as a sponsor, and you can go to datastax.com/sedaily to learn more.

[INTERVIEW CONTINUED]

[00:36:11] JM: Do you have a sense for the enterprise sales market in terms of how much it's defi ned by the top-down versus the bottoms up sales process at least in this vertical? When you're selling Mattermost, is it a process of lots of developers within the company saying, "We want Mattermost." Or is it somebody, some senior executive that says, "We need a chat solution and we're just going to go with the one where we have complete control of our data."

[00:36:46] IT: That's a great question. There's probably two sort of like categories, right? One is sort of bottoms up where we get developers interested in us, and that leads to larger deployments and commercial relationship. There's another piece where they have a list of things – They have a list of requirements, and Mattermost happens to fi t it and it's unique in the market. That typically only comes when there is actually some still internal adaption.

I couldn't tell you off the top of my head like what the split is, but I would say like there're both. I'm saying the majority is definitely the bottoms up. One development team installs Mattermost, they really love it. Some friends, "Hey, can I bump a Mattermost team off you?" They're like, "Fine."

Then one day that initial admin goes in and he's like, "What? There's 50 Mattermost teams here? Who are all these people? What am I doing? I've turned into IT." That person will go to IT and be like, "Look. This thing needs to be centrally managed. There're obviously a lot of developers. I don't want to run this thing anymore. I loved it when I started it. I hopped up the way I want. Now, I need you to run it."

Then central IT looks at it and they're like, "Okay. Well, here's the Mattermost page. What is this Mattermost thing? Here is the pricing page. Oh! Okay, obviously the thing that – All the things that IT wants, which is single sign on, active directory integration, e-discovery, that's in a paid version." The developers got all the stuff they want. IT doesn't want to run the open source version because they want their life to be easy, so they buy the commercial version. It's like it's really easy to manage users that are going to be automatically turned on or turned off based on active directory. The single sign on works, their lives are easy. They have budget. That's kind of how the business – That's the majority of the business. It's not only sales-led, it's really bottoms up.

[00:38:20] JM: There's a perspective in the software business that the gigantic companies like AWS and Microsoft and Google and Apple are taking all of the opportunity. Do you feel like these companies are so big in any way that they're inhibiting your ability to build a strong business?

[00:38:41] IT: That's a great question. Do the sort of mega corps inhibit like the smaller companies?

[00:38:47] JM: Or you in particular. This product category.

[00:38:50] IT: I think in terms of brand a little bit, the market is so large, right? I think all these large companies, they have very strong PR teams and marketing teams and they want to create the impression that like, "Yes, this is your only choice. Because this is the only thing you've ever heard of."

But like the reality is like Salesforce, which is like a giant, right? It's like CRM. It only owns like 40% of the market, which is shocking, like, "What? Salesforce is only 40% of the market." You'll see that with like Starbucks. That's not – When you look at the whole coffee market, it's not actually that big. I think that there's a ton of opportunity.

I think the thing with technology is that it's always changing, right? The winners in the last 20 years, in the last 10 years, in the last 5 years, it constantly shifts and things can change so fast. I think the arc of software is long, but it bends towards open source.

[00:39:41] JM: How do you size the market? Are there really that many enterprises that don't even have a chat solution yet?

[00:39:49] IT: Yeah. I think if you look at – I mean, if you look at Slack's S1, and they size the market about $30 billion. I mean, look at the analysts coverage, the market, it's around that ballpark. What it is, is well what's the fundamental platform for people where people are going to collaborate and they're going to build things together? It's really a massive market. Unifi ed communications is what? 40 billion and it's growing. That includes voice and video and meeting rooms and things like that.

Ultimately, the way that we collaborate is changing, right? Most people, email is like it's not the primary medium for a lot of people. What they need is sort of three things. They need voice video screen share. They need messaging and they need documents. They spend most of their lives especially internally in those sort of three things. What are those things look like in future?

I'll just say one more thing about where I think the big companies are going. Microsoft is pushing a lot of people to Office365 and people are leaving their self-hosted exchange deployments, and everything is going to cloud. The thing to watch out for in the next 3 to 5 years is all these enterprises go to cloud is how many of them, now that they're comfortable with cloud, will start going to GSuite? The reason why they're going to start considering GSuite is because GSuite owns the earlier generation, right?

People don't use Outlook.com for personal email anymore. They all use Gmail. When they have that experience and they have Google Docs going to schools and in their personal lives, it's going to be a lot easier to become a GSuite user when you go in the workforce, because if you haven't had the Office365 experience earlier on, and I know this because I worked at Microsoft and I was the head of HotMail and Outlook.com and it was really hard back then, because Gmail was so powerful and it was really changing things. That's just one example of how the collaboration market has noticeably changed, is that Office is no longer the familiar easy to use thing. It's this like, "Wait, I'm going to use – This Microsoft Office thing. I use Google Speadsheets. I use Google Docs. I use Google everything. How does this Office thing work?" That's like shocking to a lot of people, but it's true.

[00:41:52] JM: How do you think that applies to your business?

[00:41:54] IT: I think that change and behavior and that change in terms of market, it's just one example of how much change is going to happen.

[00:42:01] JM: Understood. The world has obviously moved in the direction of enterprise chat, but we haven't had any successful enterprise social networking. Why hasn't there been a successful enterprise social networking tool?

[00:42:13] IT: I guess the question is like should there be a successful enterprise social networking tool? What's the need that it solves?

[00:42:22] JM: I have no idea. Newsfeed. Do you want a newsfeed for your organization?

[00:42:29] IT: I haven't seen a product that's got that right yet.

[00:42:32] JM: But do I want it? Should I have it? Should I log in and have a digest of the company events today?

[00:42:40] IT: I think that's a need. I don't know if like technology is the best way to solve that versus an announcement versus leadership, and like the leadership of company being able to like articulate what is really important. Whether it's financial plans or lunch plans or what the R&D initiatives are. That seems like a human form of communication to me.

[00:43:00] JM: Since we're at GitLab Commit, GitLab is all about this rapid product expansion. They're doing a million things. Pretty interesting, the chronology of different software companies. I can't think of another company that has expanded in the way that GitLab is doing. If you were to think about the GitLab product development philosophy, if you applied to your own company to Mattermost, what would that look like? What would be the best adjacencies to expand to?

[00:43:27] IT: I think it's a response to something that we released in alpha a few months ago, and I would think it's slightly different. Iteration is super important, but it's not breadth. It's really about what does the customer need and what does the customer want? What are the use cases? What are they doing? What are they adding to Mattermost software that we're not addressing? What's the work they have to do? How do we make their lives easier? How do we make their lives out-of-the-box?

To response as an example of that, which is customers are building this themselves, right? PagerDuty is a wonderful product. It's a SaaS product, and per older conversation, when you have information that's not really yours, it's someone else's, there are a lot of PagerDuty customers that actually can't put anything other than a URL in PagerDuty, right? You got a page and you have a URL that's like, "There's something here," but they can't tell you what it is, because they can't take that information and put it in a third-party product.

We have customers that create their own PagerDuty within Mattermost. There's an instant, they have automation, and they automatically create a channel and they mention a certain number of people that need to come to that channel, they add like a bunch of reports. Then from there, they can use that command line to pull in different ports, pull in information and really sort of like solve the incident.

At the end of that, they have a postmortem. They say like, "Okay, here's what we did and here's where it was really efficient. Here is where we kind of get lost and here's what we can do better. We should probably pre-can some of these reports. We had to run these as well to manual." They had the postmortem. They write it up. Then they take all of it and they run an export function that throws into a PDF format that puts it into their like infosec compliance like system. They do that custom, because we didn't built that out-of-the-box for them. They want us to build that for them and they'll even want to buy it from us, because it's way easier for us to create it and us to maintain it and us to continually approve it than for them to stitch together these pieces.

I think that it's really about listening to your customer and iterating and just being able to deliver what they need and have that partnership. So like you solve their problems. They'll buy and they'll want to buy and they'll want to buy more.

[00:45:22] JM: The average company that is installing Mattermost and using it at scale. Just to revisit the customer desire, how does the – When the customers talk about what they like about

Mattermost, is it more about the total control of the data or is it more about the customizability of the tool?

[00:45:44] IT: That's a great question. Trust is a constraint. It's not an optimization. People don't want to – If you can trust me like 10 times more, it doesn't really matter. You just want to like – Trust is Boolean. If people want the trust, they want to have the trust as simply as possible, right? How can I put collaboration in a high-security on-prem environment that has tons of regulations in the simplest way possible and smoothly as possible?

Then the optimization is the workflows. It's like how many things can I connect to? How much time can you save me? How do I take hours in turn them into seconds? The best example is really connecting the legacy applications. I'm this company and I'm like, "I'm going to cloud." You're like, "Well, what do you mean you're going to cloud?" "We're going to put 10% of the company on cloud." It's like, "Okay. How far will you get?" It's like, "We're going to get to 50%. In like 10 years, we'll get to 50%."

Why is that? It's because this apple over here that runs on a mainframe me has $4 billion of revenue going through it and no one understands how it's going to work. It's never going to cloud. They're going to go on eBay and buy used parts to fix that mainframe until the end of time. They're not going to be able to make that shift, right? No one's career is going to last long enough.

You need a way to connect those private networks. What people are building, they're building RESTful APIs on these old COBOL solutions you have to tail that into. Then for Mattermost, you can actually connect through the command line, a /command. You can say like, "Hey, I want to pull this report. I want to pull this data." I think that's the way that – That is the user interface people want. They want an open source user interface that's really easy, that's connects our legacy apps, that supports all their security infrastructure. They don't have to trust it, because they can read all the source code.

That is like imagine you're in hell and someone brings you a glass of ice water. That's what it feels like working these really old rickety high-security legacy applications. How do you make it modernized?

Guess what, when you add Mattermost, you can't tell that it's an old legacy app, right? It's run a little slow, right? But you can't tell, because everything connects sort of in the same way. You pull the same reports and you have the same data. What you've done is you've modernized, right? Then with Mattermost, you can actually go to cloud and you can take this with you.

[00:47:52] JM: If you aren't building Mattermost today, what company would you be building?

[00:47:55] IT: Oh, that's a great question. I have no idea. I'm really happy building Mattermost.

[00:47:59] JM: A gaming company perhaps?

[00:48:01] IT: Gaming is like when you're dev, there are two things you want to do. You want to make video games and you want to make open source. I got that videogames checkbox. Videogames is really hard talking to games, because like especially running – It's like you have to work the holidays. It's going to be hundred-hour weeks. It's very real-time and like the forms aren't really fi lled with praise if you're in certain types of games. Yeah.

[00:48:23] JM: It's a hits business, right?

[00:48:24] JM: It's a hits-driven business as well. I learned a lot.

[00:48:27] JM: Last question. What did you learn from Andy Grove?

[00:48:30] IT: Oh! Andy. I learned a lot. What Andy had was he had immense ability to focus. He had this like just like iron will, right? It was just amazing, like when he was focused on something, how clear he could be. That's something I always like admire and aspire to.

[00:48:50] JM: Okay. Good answer. Ian, thanks for coming on Software Engineering Daily.

[00:48:52] IT: Yeah, thanks so much for having me.

[END OF INTERVIEW]

[00:49:02] JM: As businesses become more integrated with their software than ever before, it has become possible to understand the business more clearly through monitoring, logging and advanced data visibility. Sumo Logic is a continuous intelligence platform that builds tools for operations, security and cloud native infrastructure. The company has studied thousands of businesses to get an understanding of modern continuous intelligence and then compile that information into the continuous intelligence report, which is available at softwareengineeringdaily.com/sumologic.

The Sumo Logic continuous intelligence report contains statistics about the modern world of infrastructure. Here are some statistics I found particularly useful; 64% of the businesses in the survey were entirely on Amazon Web Services, which was vastly more than any other cloud provider, or multi-cloud, or on-prem deployment. That’s a lot of infrastructure on AWS.

Another factoid I found was that a typical enterprise uses 15 AWS services, and one in three enterprises uses AWS Lambda. It appears serverless is catching on. There are lots of other fascinating statistics in the continuous intelligence report, including information on database adaption, Kubernetes and web server popularity.

Go to softwareengineeringdaily.com/sumologic and download the continuous intelligence report today. Thank you to Sumo Logic for being a sponsor of Software Engineering Daily.

[END]