EPISODE 990

[INTRODUCTION]

[00:00:00] JM: The word DevOps has a different definition depending on who you ask. For some people, DevOps is about the process of managing and releasing code. It can involve container management and service orchestration. DevOps can involve infrastructure as code and safer configuration management. In addition to a set of technologies, DevOps can be seen as a management concept that describes agile practices and breaking down communication barriers between different teams.

One thing that most software companies have decided is that whatever DevOps is, we want we want it. We want to release more software. We want to do it faster and we want to do it safer. We want streamlined communication between management and engineering. We want a full understanding of the value chain of software.

Despite the elusiveness of a single description for what DevOps is, GitLab can credibly describe itself is a tool that satisfies the DevOps needs of many enterprises. GitLab started as an open source version control management system based on Git, and it has expanded into products that include continuous integration, security, issue tracking and monitoring. The trajectory of GitLab into such a large, flexible platform is something that few people anticipated.

The best explanation for how it happened is that the downstream result of an engineer within GitLab deciding that the code hosting product needed to have a continuous integration product bundled with it created this tightly coupled, unified workflow that enterprises actually liked. The reason that they like it is because, today, there are many enterprises that are trying to make a big set of changes to their software development practices. Everyone is trying to do “digital transformation”. The world is consolidating around Git for version control and Kubernetes for container management.

Almost every enterprise is figuring out a “cloud strategy”. Every team wants to have continuous integration and they wanted to have some security products paired into that continuous

integration workflow and they want to have a release workflow that works in a popular, vaguely defined set of practices known as dev sec ops, or perhaps a GitOps.

With so many changes coming to enterprises, it turns out that many of these enterprises just want some sane defaults. When GitLab came to market with a bundled CI and code hosting product, the company discovered that the customers were very happy to have integrated tools that worked well out-of-the-box, and this was in stark contrast to the years of end-by-end tooling integration that an enterprise would have to stitch together and make a broad range of carefully selected tools.

Sid Sijbrandij is the CEO of GitLab and he joins the show for a conversation about how GitLab arrived at its product development strategy. In a previous episode, Sid discussed some of the core features and history of GitLab.

Today's show expands on many of the subjects that we explored in the previous interview and we also had a spirited discussion of the modern nature of work and how GitLab's unique culture and fully remote team have evolved as the company has scaled.

We have partnered with SafeGraph for the SafeGraph Data Hackathon Challenge. We're giving away $4,000 in cash prizes as well as Software Engineering Daily and SafeGraph swag. We've got Software Engineering Daily hats and t-shirts and other kinds of swag, socks. SafeGraph is a geospatial data company which curates a dataset of more than 6 million points of interest. It's a lot of location data and there's a lot you can do with that location data. You can build apps and data science projects with that data. If you've been looking for a creative opportunity to explore large datasets with the potential to win $4,000 in cash prizes, this is a great opportunity. The hackathon is hosted on FindCollabs. It's going until February 24th. So you got plenty of time to create a project and start hacking on some data science, and you just go to findcollabs.com to sign up.

I remember when I was in college, I was hacking with the Yelp API, and the Yelp API does have some interesting location data, but it's nothing compared to what SafeGraph has. SafeGraph is a data as a service company, so I think it's a pretty interesting company concept, the idea that you can get data as a service, and this is a great opportunity to try out the offerings of the

company and perhaps win some money or Software Engineering Daily swag. Check it out at findcollabs.com.

[SPONSOR MESSAGE]

[00:05:03] 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.

[INTERVIEW]

[00:06:53] JM: Sid, welcome back to Software Engineering Daily.

[00:06:55] SS: Yeah. Thanks for having me.

[00:06:56] JM: GitLab has highlighted something called the tool chain crisis. Describe the tool chain crisis as you see it.

[00:07:03] SS: I think what happened is that overtime, we got more and more tools in the DevOps process. It started with just some version control, some CI. But overtime, there are so many best practices we're implementing through tools. For example, you can see this in progressive delivery. We have feature flags that are happening. We have sometimes Chaos Monkeys. We got incremental rollouts. We got automated rollbacks. All these things are supported by tools. What happened is that every practice kind of got its own tool. Now what's happening is if you want to go from planning something to getting it out there, you have to go to 10, 20 tools to do so in it. Between all these tools or handoffs. There's no visibility.

Some data is in one tool. Some is in another. People don't have equal access. I mean, some tools, but not other tools. That's slowing people down. These handoffs are taking longer, and that's the crisis. We have too many tools. Every customer we meet, every user we meet is like, "Yeah! DevOps tools are great and we're using all the best ones." But our problem isn't the individual tools. They can do what we need. Our problem is integrating all of that. Our problem is transferring people between teams. Our problem is visibility. That's the tool chain crisis.

[00:08:19] JM: There are a number of case studies of GitLab usage within large enterprises. Goldman Sachs is an example. A company like Goldman Sachs, they've had software infrastructure that they've been building on for decades. It's like an archaeological dig. They have all these different strata of old tools and these things are all plugged into mission-critical workflows.

When they start integrating GitLab into their workflow, is it just an additional tool where the future that they foresee is a consolidated workflow around GitLab or are they literally throwing out tools today and rapidly adapting GitLab as that consolidated workflow? I guess I'm asking, they pushing this to happen soon or are they comfortable with this being a gradual process?

[00:09:12] SS: The latter. For their most important application, they brought the time that they needed down from two weeks to two hours. For the most legacy, the longest lived, the most complex apps, they're using GitLab to consolidate their tool chain. We meet CIOs and we tell

our story and they're like, "Finally, the last five vendors I had were all an additional purchase, an additional tool we'd have to do. You're coming here and you're telling me I can get rid of these 15 tools? That is perfect. That is what I want to do."

[00:09:48] JM: Overtime, time you've added more and more functionality to become an increasingly fully integrated tool. How do you avoid becoming like the AWS homepage where there's just so much stuff going on? It's hard to know exactly where to go or what your workflow should be.

[00:10:07] SS: We want to make sure there's convention over configuration. We have technologies like auto DevOps where you push your code and GitLab will do the right thing. You don't have to set all of these up. They should work out-of-the-box. You push your code, we make sure that it's tested, that the performances is checked, that the accessibility is checked, that the vulnerabilities are scanned against, that the monitoring is in place, that there's network security applied. All these things should work out-of-the-box. That's what we try to do, convention over configuration. If you want, you can dive deep, but you don't have to dive deep from the start. If you have a simple app, it should be simple to deploy it, which you should still immediately have the logging set up, for example.

[00:10:52] JM: Does GitLab grow within an enterprise organically or does it need some kind of change agent within the organization to lead the adaption of GitLab?

[00:11:04] SS: It always helps to have a change agent. There's amazing organic adaption. We talked about Goldman earlier, and they said, "Well, it's going to take six months to get to a thousand users." In two weeks, they were at 1,500 users. Within six months, they were at 5,000 users. There's strong kind of organic adaption. Developers like GitLab. If there's something they wouldn't like about it, they would fix it, change it, send it upstream and it would get solved for everyone. It is a tool the developers want to use. It really helps if there's also some commitment from the top of the organization.

What we frequently see is that individuals and teams love to use GitLab, but that the middle management of the company just spent the last five years selecting their preferred point solutions and building a platform around that. Sometimes it takes a proof of concept where that

platform, that DIY ops platform is compared against GitLab, and then they'll have to conclude what 100,000 companies can build together as a platform is better than what we can build by ourselves. But it sometimes takes the top of the organization to say, “Hey, let's compare the two.”

[00:12:16] JM: Why has security become closely integrated with the DevOps lifecycle?

[00:12:20] SS: I think security used to be an after fork. You built your code and then you made sure it was secure. You were done with your code and then let's do the security scan. But there’re now so many technologies to make code more secure. There’s static and dynamic analysis, dependency and container scanning, and there's so much work coming out of that that it's no longer enough to do it at the end of the cycle, because at the end of the cycle, when you're done, that is when you’ve invested the most money. Any delay you had at that point is amazingly expensive. The sooner you can find potential problems, the sooner you can address them, the more affordable that is. That's why security is shifting left and becoming an integral part of the dev sec ops lifecycle.

[00:13:07] JM: The strategy of your company since the early days has had a number of strategic shifts, and the most fervent shifts were in the earlier days before you had really figured out the monetization strategy. There is not an obvious path to going from an open source version control system to a DevOps platform.

I want to know more about how you think about the high-level strategy of company building. Just talking more abstractly than just GitLab, what are the concepts and strategy that have applied to GitLab that could apply to any other company in terms of how you see strategy?

[00:13:52] SS: Our mission is everyone can contribute, and the biggest decision in our company history has been to put the first tap towards that platform. It came from Kamil. Kamil made a better version of GitLab CI, and he posted it on the Internet and we’re like, “This is much better than what we’ve made.” For example, the run-up was written in Go instead of in Ruby, which has fewer dependencies, so it was easier for people to run.

We said, “We’ll make this official. F from now on, this is the official way to run your tests. Also, Kamil, do you want to join the company, because you're really good at this? We love to work even closer with you.” He joined, and at that time, GitLab was just version control, plus there was this project that Dimitri started, my cofounder. He never asked anybody. He just said, “I never want to upgrade Jenkins ever again in my life, and I'm going to make sure I write my own tests.”

He wrote his own CI platform, his own testing software, and two things happened. A few people found out about it and started using it, but also it kind of didn't scale for our purposes. W we needed to like invests more to make it better, and we did that. When Kamil came in, he said, “Look, we got this GitLab version control and this GitLabCI. we need to combine them into one.” Dimitri said, “You’re wrong. These are already perfectly integrated. They have single sign-on. They couldn't be integrated any better than they already are.”

I also explained how he was wrong. Everyone in our market had like separate offering, separate products and all our customers told us, “They wanted to be able to combine things.” He said, “Well, that might be right, but it's going to be more efficient if we do it in one application. It’s going to be so much better for the user.” Yet, we didn't initially agree, but overtime, he convinced us, and we did it, and he was totally right. It is a much better experience.

Our users, they don't mind that there's extra functionality in GitLab, as long as if they want to use something else, like Jira, or Jenkins, or GitHub, they can do so, and that's possible. We have plug-ins. We have APIs. We have integrations. But the benefit of having more and more in a single application, a single user interface, a single concept of what a user can do, that was totally right. Our best strategic move ever wasn't because I said it. It wasn't because Dimitri said it. We were actually against it. It was because someone that we didn't knew a year before came with an idea pushed for it. To this day, one of the values at GitLab is disagree, commit, and disagree. Please feel free to disagree with your people. Do commit if a decision is being made, but feel free to keep pushing for what you know is right, because our initial response might not be the right one.

[00:16:51] JM: The competitive landscape of software today, you've got giant, horizontal cloud providers. You’ve got individual point solution products. You’ve got open source projects, and

then there's the enterprise buyer, and then there's obviously the startup buyer who eventually becomes the enterprise. Each of these different players, you have to decide how to form partnerships, how to create alliances. You have to acknowledge that some people are simply competing with you directly.

What is your process for categorizing other players in the software landscape in terms of their relative levels of competition to you?

[00:17:35] SS: Yeah. First of all, we do welcome competition. We welcome anybody to integrate with GitLab even if there's functionality already in GitLab that does the same thing. It's great if our customers, our users have options. If you want to integrate with GitLab, please do. We have open APIs. We welcome the integration. We'll promote you on our blog. Nobody wants to be locked in. It's very important that we keep signaling to the market. GitLab is an open platform and people can build on that. We're not going to favor what is in GitLab over anything else.

Our most important partners are the major cloud providers. Where at GitLab Commit, it's sponsored by AWS and Google, and I hope on future edition, we'll also be sponsored by Azure and by VMware, because that's where people are deploying workloads to and we want to make sure that GitLab is the best tool to do multicloud to use different cloud providers. It would have the same workflow, have the same way to measure engineering productivity. Has the same way to ensure compliance, to ensure security. Do that in GitLab and be able to use any cloud.

[00:18:46] JM: There's so much discussion about AWS deploying services that are copies of open source projects. Why hasn't AWS offered an Amazon GitLab service?

[00:19:01] SS: You would have to ask them.

[00:19:02] JM: Do you have any suspicions If you put yourself in their shoes?

[00:19:06] SS: I think what's important to AWS is that they sell compute in the end, and although GitLab does require compute, for example, running tests can be quite a workload. In practice, the most compute is generated by things that are in production. Those are the things

that GitLab deploys, but GitLab itself is not that. I think it's less attractive from that view, that perspective.

[00:19:35] JM: Your strategy involves breadth over depth, and we talked about this a lot in the previous episode we did. Breadth over depth is an uncommon strategy. How do you convey that strategy properly to your team as it has grown to more than 1,100 people?

[00:19:57] SS: Yeah, by putting it on the homepage. If you go to our homepage, you'll see what parts of GitLab we're still trying to ship, and that's not just to communicate it to our users, but also to communicate it internally. What is logical is like we love our customers, and our customers will never ask us to ship a big, new – Like add additional scope to GitLab, because they're already using all the parts of GitLab and you want to see that improved. That's very logical, but we got to make sure that if you want to get the maximum rewards, like the more parts of the DevOps process are in the single application, it does like exponential returns. There're two parts in GitLab that's nice, but if there's four parts or five parts, that's way, way better. That's why we had that strategy. We want to push ourselves to encompass the entire DevOps process.

[00:20:53] JM: You want to have lots of areas of the GitLab product even if some of them are not as complete as you would like. What are the downsides of the breadth over depth strategy?

[00:21:11] SS: I think there is a downside if it doesn't improve. One of our values is iteration. The first time we ship any new functionality, it will be a minimum viable product or a minimum viable change. But then it's important that it keeps improving. There're a couple of things that we know are essential. One of them is dog fooding. Say, if we have a minimal product, we got to make sure we use it ourselves, because that's a really quick way to improve it. The second thing is we got to get the contributions from customers. If we bite of something new, it's important to ship it as part of our core offering that everyone uses, because then we have many more potential people who can contribute.

With that, we got to show improvement. We got to show that it improves every month. You can read our process. Many, many parts of GitLab are improving every month. We got to be very clear about what parts of GitLab are mature and not. On our homepage you'll find that table with

what we still plan to make, but below there is a button, and if you click that button, it shows you the maturity of any part of GitLab we already shipped. Even though we already shipped it, it can be minimal, it can be viable, it can be complete or it can be lovable. We are very honest about which parts are very mature and which parts still needs a lot of work. I think being honest about that helps our customers evaluate which parts they're going to adapt or not, because they have different risk profiles. You're going to act differently if someone, something, you have a great solution already and it's super risky and there's a lot of attention on it, versus you don't have an existing solution and its relatively low stakes.

[00:22:50] JM: Can you envision a world where GitLab becomes a cloud provider?

[00:22:54] SS: We have no intention on becoming a cloud provider. It's a very different business model with a lot of capital expenditures, and there is a great economy of scale. The world is not going to have 10 big cloud providers. I don't know whether it will be three, of four, or five, but it's going to be a limited set and we're not going to be amongst them. We're going to help people get the most out of their cloud providers.

[00:23:18] JM: What about being a layer 2 cloud provider where somebody wants to deploy something and you take care of the selection of the underlying deployment medium?

[00:23:31] SS: I think we do want to enable to people to deploy two different clouds. Today if you use GitLab, you can easily add Kubernetes clusters. You can do so manually, but for AWS and Google, we have integrations where we set the cluster up for you. We make it easier to consume that. Then you can assign those clusters to different workloads. You can have your development workload on a different cluster than the production workload, and that's what we want to enable people to do. Make it easier to shift between them, so give people more choice. Then the choice they make, I think people – That's not the problem. The problem is not for people to pick. The problem is to make sure your application deploys as well to one cloud as another cloud and you have the same workflow, the same productivity, the same compliance in every cloud, and that's what we're setting out to do.

[SPONSOR MESSAGE]

[00:24:31] 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 CONTINUED]

[00:26:02] JM: What do you think the long-term competitive breakdown between AWS, Google and Azure looks like? What are their differentiating factors?

[00:26:15] SS: AWS has the biggest breadth. They have the most services. Azure is doing an amazing job of speaking to the .NET ecosystem, and Google is doing a great job of embracing different open source projects and making sure there's a business model for them as well. Very different strategies, all of them great ones, and what we're going to see out in the market how that plays out, but it's certainly very exciting to watch from the sidelines.

[00:26:46] JM: What's been the hardest part of scaling up the sales process at GitLab?

[00:26:52] SS: I think one of the hard things is that we have a hybrid sales process. For smaller companies, we have a self-serve model. For medium size deals and companies, we have inside sales, or the salespeople, like video call with the customers. For the top of the market, we have enterprise sales, where we go visit the customer with solution architects and everything else. Many companies specialize in one or the other. We have to offer all three. That makes it harder.

Another hard thing is if you scale really rapidly, you have to hire way ahead. Our average deal cycle is, let's say for a big deal, it's six months. Then we got to make sure that we already start recruiting and then have that person in place six months before, because it will take them a while to ramp-up. Every time you're late in hiring someone, that's going to affect your revenue six months downstream.

[00:27:48] JM: We've done a fair bit of coverage of scaling a remote workforce in terms of engineering. Is there anything that has surprised you about scaling up the sales process and sales function of remote sales teams?

[00:28:07] SS: I think that a misunderstood thing is that sales teams have a bigger office culture. Because salespeople have to be at the customer location, they're very used to kind of being effective while on the road. They frequently live close to their customers. Sales organizations are stewards split up by geographies. There's not a big need for the salespeople to be at the office every day. What they do like is to come together. We have a sales kickoff. We're introducing a presidents club for the people that achieved amazing goals. Go on a trip together.

With sales, they're used to being remote, but events at like in-person is still important, and you can do that with events but also, for example, our trainings are more in-person for the salespeople than they are for all the departments.

[00:29:02] JM: You've declared that the GitLab company strategy involves going public sometime in 2020. What you need to accomplish in order to be in a position to go public later this year?

[00:29:13] SS: Most important thing is growth. We're a growth company. So we got to show revenue growth. The other thing we have to show is consistency. We have to be able to predict upfront how we're going to do, and we have to show consistent growth. Predictable growth is the most important thing we need to show to the public.

[00:29:34] JM: In terms of developing the predictable growth metrics that you need, do you feel like it is more of an iterative expansion of the current strategy or do you feel like you have to figure out some one weird trick of expanding into some new product like a data engineering product or something? Do feel like your core product in its current state and with the current market susceptibility is going to get you where you need to go?

[00:30:06] SS: Yeah. We think that our current product can do that. To give you an idea about the market opportunity, there're 20 million developers. There're about 20 million operations and security people. For our top-tier gold, the price of that is over $1,000 per user per year. You got 40 million people at a maximum price point at the – Currently, $1,000. That's a $40 billion market. We're going to cross 100 million ARs. There is ample room for expansion. Even if we get to half of the market, that is, we can be 100 times bigger than we are today.

[00:30:46] JM: Have you considered the direct listing model?

[00:30:48] SS: For sure. We're considering both for direct listing in an IPO. It's easier to do an IPO. So we're focusing on the direct listing and we can always make it easier on ourselves if we decide otherwise.

[00:31:03] JM: What do you see is the main advantages to the direct listing model?

[00:31:06] SS: You don't this kind of IPO pop that is common. So you leave less money on the table. You get to talk more frank with investors. You can share more information about your financial model. There's no lockup period. So you don't have this period of half a year where there's trading, but not everyone can trade. It's kind of everyone is still very unsure about what the real share price is. Those are some of the advantages.

[00:31:33] JM: How do you feel employee incentives compare between the IPO model and the direct listing model?

[00:31:42] SS: I don't think it's a major difference, although it's nice to not have a lockup. The lockup [inaudible 00:31:47] as well.

[00:31:49] JM: You've got a solid product as it is and you have plenty of room for this iterative expansion, but you are also still experimenting with things like Meltano, and I'm sure you've got some other experiments in the lab. How do you allocate company resources between core product work and experimentation?

[00:32:13] SS: Yeah. Actually we don't have any experience beyond Meltano. There's no secret list of experience. No.

[00:32:19] JM: I don't believe you.

[00:32:20] SS: Okay. Well, you don't have to believe me. Meltano, we think it's a great idea to make the data lifecycle easier and to make it easier to go from extracting data to getting the graphs you want. The size of the investment is [inaudible 00:32:34]. There's something to having a team of people, like between 5 and 10 people working on a problem. It's a great size. That's the size of that team. I will keep sustaining that size until we have a clear path to kind of – We're starting to generate revenue and it's time to expand. Until that, we'll keep it at that size, which is I think less than 2% of our total costs.

[00:33:02] JM: The days of Google being able to – Say, everybody's got 20% time and we hope you work on something cool. Then Google occasionally getting these outsize returns from the 20% time, like occasionally somebody builds Gmail. Do you think those days are gone in terms of software innovation from employees just randomly producing cool projects? Because we're kind of at a point where if you're an entrepreneurial employee within an organization, you probably just leave that organization and start your own thing, right? Can you still get entrepreneurial startup level efforts out of internal employees?

[00:33:46] SS: I think you can, but I think the 20% time is a myth. Gmail was not created in 20% time. It was created in 100% time. It was the person's full-time focus. They got clearance to work on that the entire time. Also, the 20% time was more like you can work on your own stuff on Sunday. It feels more like 120% time. I think what's important is that you can use company resources to do something new. You don't need permission to work on something new as long as it doesn't affect your day job. That's exactly the mission we have. Everyone can contribute. If you want to contribute something else, great. Feel free to write that code.

We also exemplify by the value we have, and that is short toes, because people who join the company are like, “I'm not sure I can do that, because it’s someone else's job.” We’re like, “We have short toes here. You can contribute something. If you want to contribute something to something that someone else's responsibility, you should be glad that you are helping them. If it's not what they think the right direction is, they can always say no to it. I think that's really important. If that's exemplified by a 20% kind of myth that, that's great. We support that. But I do think it's important that people have a balance and that they're not – That the 20% time is not something which we glorify working too many hours.

[00:35:22] JM: Could you say more about that? I would just love to know an expansion on that perspective of how many hours we should be working these days and what are “work-life balance” should look like, especially once we have commute factored out. For example, we have this question, if you're an employee and you do have access to the remote work-life style, you can take a break during the day and go to the gym or go shopping or go have coffee with friends, but then you might have this pang of responsibility in yourself that says, “Maybe I should go check my email at 7:30 PM to make up for lost time. Maybe I should be working all the time that I'm not doing the things that relax me.”

Give me your perspective, the holistic perspective on modern “work-life balance”.

[00:36:22] SS: Yeah. Thanks for asking. I think that the things you mentioned are very important. Give people back their commute time and let it be accredited to the team member, not to the employer. Give people the flexibility to intersperse work and private things, like go to the gym when it's quiet. Also, flexibility, like your kids are, you’re already at home. You're able to

quickly take care of them. If there's something coming up with family or someone who you need to let into the house, you're able to do that.

We want everyone to be able to – We have a 40-hour workweek in most countries, kind of depending on the country people are in, and we want people to be able to feel empowered to kind of work within theat. I think a clear indicator of what our people feel empowered to setting their own time is whether they take vacations and relief to see my chief of staff, Emily [inaudible 00:37:24], tweeted recently, "Hey, I'm a millennial and I care about my job, but I still took six weeks off this year." I responded in the treat stream, "Well, I got you beat. But after all, I'm the CEO, I should lead by example." I took I think almost 8 week off this year.

[00:37:42] JM: Really?

[00:37:44] SS: Really. It's much more about making the hours count. We don't waste time at GitLab. Meetings start on time. They have an agenda. We come to a conclusion. If a meeting is done, it's done. Our ET meeting this week, 50-minute schedule. We took a 20-minute meeting and we were out of agenda items and we stopped the meeting. Everyone's busy enough already.

A part of a meeting isn't relevant to you, you can totally do other things. Leave your web cam on, but just – It's fine to check emails, fine to check Facebook. We don't mind. We're not the boss of your attention. I think one of the most important things is don't reward hours. Reward results. Managers are not allowed to talk with you about how long you work unless they're suspect you're working too long. We don't celebrate people working outside of office hours. It's forbidden to thank people in a public setting for doing that. Those are the things that I think we can steer and, yeah, we really try to get people to have a sustainable balance because we want them to stay with the company for a long time. Our retention is 85% year-over-year. Just twice as good as the rest of the industry, and hopefully that's also because it's something they can sustain over a long period.

[00:39:08] JM: I like what you just said there, because one of the things that actually caused me to basically leave my career as a software engineer and become a podcaster was I did not like the fact that your workplace doubles as a disciplinarian, and it seems like you have somehow

arrived at that perspective that it doesn't make sense for your workplace to be your disciplinarian. Your workplace should get value out of you as an employee. We should not be a disciplinarian.

[00:39:41] SS: Yeah. It's the result to produce. At GitLab, we don't do cross-functional teams. Everyone gets a manager who understands their job, because otherwise they cannot assess your results. When they cannot assess your results, they're going to assess your hours, and that's what we don't want to happen.

[00:40:00] JM: To make this a timely podcast, there is a subtle debate going on Twitter right now about performance reviews. Give a perspective on what component of performance review should play within a company's management structure.

[00:40:15] SS: We think it's really important to do kind of these formal reviews as well. Of course, feedback should flow freely. Every time you exchange a message, whether that's Slack, whether that's on an issue tracker, whether that's in a video call, you can give people feedback about their work. But it is important to on a regular cadence, say, "Hey, how am I doing?" and check-in not just with your manager, but with everyone. We do a 360 review two times a year.

It's also important to not have to be the compensation review, because if you combine the two, the compensation is so important to people's lives that it sometimes overshadows the performance feedback they're getting. We have to do 360 reviews and they serve as a feedback to the yearly compensation adjustment we do, but the timing – We're not timing them in the same week or month.

[00:41:17] JM: There's a phrase that I've heard about companies wanting to facilitate an employee's best work, the best work of their career. I have mixed feelings about this phrase, because to some extent if you as a knowledge worker are doing your "best work" through the vessel of an employer, that means that your "best work" is not contributed to something that is your own personal artifact, your own vessel of creativity. My guess is that much of your work on GitLab is driven by a creative instinct. You want to bring your own personal vision into the world.

Assuming that's true, can you help me reconcile those two ideas? The idea that you personally have worked so hard and have found so much validation in the idea that you are building this company that is your vision. Contrast it with the idea that some people structure their company with the belief that they are going to get the best creative work of an employee's life out of that employee in the 10-year of the company. Should it be perhaps the role of an entrepreneur, a leader, to inspire their employees to pursue something that is their own creative vision?

[00:43:02] SS: I certainly hope that a lot of people at GitLab of the 1,100 team members we have are doing the best work that they'll ever do. That's not the criteria we have. We want them to do great work. We want them to achieve the results, achieve the goals that we set as a company. I don't think that because your work is part of something bigger or part of the vessel of a company that it's anything less. It's still your work. It still can be attributed to you. These days with version control and contribution to code, even to our handbook, like until the end of time we'll be able to trace back who invented coffee chats, and his name is [inaudible 00:43:50]. I do think you can attribute it to you personally.

Now, I'm an entrepreneur. I love entrepreneurship. Not everyone can be starting a company. Some people will be leading the company. Some people will work within the company. A lot of that people are starting companies and very proud that, for example, our old VP of product, [inaudible 00:44:14], started remote.com. I hope that there will be lots of people who their next thing after GitLab will be starting their own company, and I hope that they copy parts of our handbook when they do so. But I do want to be a place where people can do great work. For some people, that's going to be the greatest work they'll ever do, and I think that's something to be – To celebrate and be really proud of.

[00:44:41] JM: Coming back to GitLab itself. GitLab builds a unified experience, because the goal is to save engineers some of the work of integrating different products. Can you give an example of the types of integration issues that a typical enterprise is burning their resources on?

[00:45:03] SS: Yeah, for sure. Thanks to that. I want to start a new project. I make the code. I commit a diversion control. Now I want to deploy it. With GitLab, I can just push it on the DevOps [inaudible 00:45:15]. If not, first I got to set up CI. It's Jenkins. I have to contact the build engineer to help me get set up. Great. That's done. Now I got to go into production. GitLab

maybe at a Kubernetes cluster takes care of it. Otherwise you got to go to the ops team, ask them for a couple of scripts.

Now there's a problem. I need to look at the log files. Now at GitLab, I need to set up logging, have look at it. Now I suspect there's a problem but I'm not sure. I need metrics. [inaudible 00:45:49] GitLab. I don't know. Add data dock to it. That goes on and on. It goes on for security where there are different vendor selling dependency and container scanning, static and dynamic code analysis. Overtime, if you look at the big project in a company, overtime, all these tools got integrated. They just have the 15 or 20 tools, but the majority of projects don't. Every time they kind of bump their head, they have to add another tool. It's not clear where things are. Sometimes the ops people are looking at the metrics. Those metrics are not accessible to the developers that are actually working on the code.

Sometimes developers introduce a performance regression without knowing it. For example, GitLab has a function called web performance where you can automatically see that. I can't discuss all the things, but it's like all these tools are something you have to consciously add if you don't have GitLab, and people end up using different tools and adding them in different ways which makes collaboration way harder.

[SPONSOR MESSAGE]

[00:47:12] JM: When I'm building a new product, G2i is the company that I call on to help me find a developer who can build the first version of my product. G2i is a hiring platform run by engineers that matches you with React, React Native, GraphQL and mobile engineers who you can trust. Whether you are a new company building your first product, like me, or an established company that wants additional engineering help, G2i has the talent that you need to accomplish your goals.

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

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

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

[INTERVIEW CONTINUED]

[00:49:01] JM: GitLab runs contrary to some of the conventions of software. Most of the cloud tools we buy, it does one thing particularly well. You want SMS, you get Twilio. You want servers, you buy EC2. You want payments, you use Stripe. GitLab on the other hand does a lot of things, and I know as you alluded to earlier, GitLabCI was kind of the formative piece of infrastructure of that strategy. Is there some particular reason why GitLab was able to succeed in this streamlining, in this consolidation of functionality? Something regarding timing or, I mean, why was it that that strategy was able to work?

[00:49:52] SS: I think a couple of things. I think DevOps is getting more mature. You're seeing that different parts of the DevOps tool chain are getting better defined. The other thing is that open source is now such a powerful force. I don't think it's possible for any single company to create this complete DevOps platform.

Because developers have great taste and if there's parts of the products they don't like, they're not going to adapt it completely just because this one part they don't like. I think with open source, people being able to say, “Okay. Well, I like everything at GitLab, except for this. So I’m going to fix it.” That contributes so much.

It's not just in lines of code, but it's also in like discovering these itches is super, super hard. Having people contributed just short-circuits of that cycle where you otherwise have to do a ton of product discovery to even find them. I think by making a complete DevOps product is so

much work. That's only possible if a ton of companies collaborate together, and that's what's happening with GitLab.

[00:51:11] JM: GitLab has grown popular in the context of Kubernetes. Is there any coincidence there? Is there any way in which you've taken advantage of the rise of Kubernetes?

[00:51:21] SS: Kubernetes came just at the right time for us, because we were moving from CI to CD and more to production roles, and we were considering, "Okay. This means we'll have to integrate with every single cloud." Then out of the blue, Kubernetes rose up. I'm like, "Wow! There's going to be a common interface for every cloud. So we can just program to Kubernetes. We don't have to make it for AWS and then make it for Azure and then make it for TCP and then make it for DigitalOcean. That was such a blessing. We got really lucky with Kubernetes emerging at just the right time for us to make it very easy to deploy your software to Kubernetes, and with Kubernetes, you can deploy to any cloud.

[00:52:06] JM: Tell me something unconventional about how you manage GitLab.

[00:52:11] SS: We have the shadows here. That's something unconventional to start with.

[00:52:15] JM: Who are these people?

[00:52:16] SS: Yeah. [inaudible 00:52:18]. The shadows follow me in my meeting. They're invited to about 90% of my meeting. Whatever I do, whether it's something like this, a podcast or an investor coming by or having lunch, wherever people feel comfortable with it, they are invited. It allows them to see the entire company. We're a functionally organized companies. Normally, you're always within your department in engineering, in people, in finance. All those departments end up reporting to me. So I'm the only one that has that broader view, and we have a lot of things that are transparent, but there's nothing the granularity of detail that you get by being in all these meetings. We invite people to spend two weeks and to come to every meeting and get a better perspective of how the company operates and how we take decisions.

[00:53:17] JM: Andy Jassy was a shadow of Bezos before he was part of AWS.

[00:53:21] SS: He was a bit more. He was a chief of staff. But, yes, that was certainly an inspiration. Also hiring for a chief of staff. So that's a separate role and that's a role where even more contribution is requested and it's going to be much longer-term. But when I was looking at chief of staffs, I saw like listed as the advantage that every 1-1/2 years with a chief of staff, you'll train one person in all aspects of the company. I felt, "Well, that's marvelous, but we've grown from 400 to 1,100 people last year. We need a bit more than one every 1-1/2 years." With this program we're able to graduate 4 people every year. Give a lot more people in the company that broader perspective.

[00:54:03] JM: You have like just groups of people that follow you around to each of your meetings on a rotational basis and these people become core components of institutional knowledge to spread throughout the company.

[00:54:17] SS: Exactly, and it's two people. One person in their first week, one person in their second week. They also help. They help too, for example, with note taking.

[00:54:28] JM: That is unconventional.

[00:54:31] SS: Have any of the processes that you defined earlier in the company, maybe at 400 people, had they started to break as you've gotten larger?

[00:54:43] SS: I think the one thing that is breaking is our company call and our breakout call. Every four days a week, we have a company call with announcements. You can also view the announcements into Google doc, and then we have are breakout call where you get a group with a set of people that you can talk about, like how your week was or even give you a discussion question.

The attendance at those calls state the same in absolute numbers. That means it's rapidly dropping is like a relative number, like a smaller percentage of the company is attending them. That feels like it's not scaling, and I think what we need to do is bring people together based on interests. S some people might be interested in photography, others in fitness. I think that's a better way to connect people than the randomness that we're currently practicing.

[00:55:41] JM: When you're running a remote company, does that make it harder or easier for people to express non-consensus views?

[00:55:49] SS: I think easier. I think if you're in a meeting, it sometimes even hard to get the work. People talk over each other. With GitLab, in a meeting, the word is handed to people based on when they put it in the agenda. Even that is fair. But if it's in an issue or something like that, it's even easier to like take a step back, think it through and then put your view in there. I think you see a lot more diversity of thought when we take a decision than I've seen at other companies.

[00:56:25] JM: Your handbook has a list of the 24 biggest risks to GitLab. Which one of those risks is the biggest one today?

[00:56:33] SS: Lowering our hiring part. We've more than doubling the amount of people in the company and we'll hire over a thousand people this year. If we place feeling a role above finding the right person, that will very quickly destroy the company.

[00:56:52] JM: Are there any hiring practices you've put in place to alleviate that, like a bar raiser?

[00:56:58] SS: We've talked about it. There's some experiments going on, but we don't have a bar raiser. That's part of all the interview processes today.

[00:57:07] JM: Do you have any tips for maintaining psychological composure when your work is getting overwhelming?

[00:57:13] SS: I think it's important to take a step back. Because in a meeting, if you're very disappointed or you're angry or something else, say, "Hey, I'm not feeling well." Step out of the meeting. If you're having a super bad day, step out of work. I think the way to deal with being overwhelmed is to reduce inputs and stepping away. I'm very proud that at GitLab we view mental health the same way we view physical health, and it's a perfectly valid reason to have to cancel a meeting or something like that.

[00:57:49] JM: How does the maintenance of the employee handbook at GitLab compare to the maintenance of the codebase?

[00:57:57] SS: On one hand it's similar. It's under version control. Everyone can contribute and you have kind of maintainers for certain sections. It's organized by functions. Most of the time, it's clear who kind of like – Who's in charge of a certain section. On the other hand, there is fewer automated tests, like correctness is harder. It's English. The highest level of programming language that is in use in companies. That part makes it harder. It's also multi-format. It's not just text. It's also video and pictures.

[00:58:32] JM: Late stage financing. How does the decision process of raising late stage financing compare to the early stage financing?

[00:58:39] SS: Yeah. I think what changes overtime is that your seed round will be based on the people in the company and you're A round is going to be based on product market fit. The B round is going to be based on growth, and the C round is going to be based on kind of unit economics. From the D round on, it's all about like the financial , kind of the net present value of your future cash stream. As you get more data, the focus is going to shift to the finances.

[00:59:11] JM: And the kind of auctioneering process of that, how does that compare?

[00:59:16] SS: With auctioneering, you mean selecting the investors?

[00:59:19] JM: Yeah. Maybe that's the wrong word.

[00:59:22] SS: I think no one likes to be auctioned off and investors are no different. In the beginning of the company, you choose very much by board member. Who's the person that's going to join my board? Because that's a person you'll be working with for many, many years and you'll take many important decisions together.

I think, for example, we just raised a series E, and then it's very important to us to have long-term holders. We talked, we selected companies based on how long are they known to hold your stock. You have companies that hold stock on average for more than a year, they're long.

But you also have like funds that are under that they trade a lot more and you want – it's helpful to get those long-term holders interested in your company, and I follow it actively, and the best way to get that is to have them invest.

[01:00:18] JM: Last question. From 2008 to 2012, you were working on a company called Comcoaster, and as far as I can tell, this startup was not as successful as GitLab. Were there any lessons you took away from that company that you apply to GitLab?

[01:00:36] SS: Yeah, for sure. For sure it wasn't as successful. We ended up selling it for a very small amount. Our main product was a site called AppAppeal, and it was an app store for web applications, like the iOS Store or the Android Play Store [inaudible 01:00:54] web apps able to compare web applications, and it didn't take off. People don't go to a central place for their web apps. It's something that kind of the Play Store and the iOS store kind of for some people and then it works and then advertising that works. But if there is no central place, you're going to use Google as the app store.

What we learned is that Google is the app store and you're seeing that now with people talking about the Google Tax. If you Google a certain company's names, that you get the competitive advertising. You got to bid on your own trademarks. That was the state.

I think there are lots of lessons we learned. I don't know any – There are lessons all throughout my career. I worked him many organizations. I also worked in a lot of really big organizations and it's kind of fun to see what I learned. From that, I saw a lot of things that go wrong at scale and that we're trying to prevent at GitLab.

[01:01:52] JM: Actually, so just one last questions so that we're right up against time. But we're at GitLab Commit, which is the conference that you're putting on. I go to a fair number of software engineering conferences. What are you trying to do differently? What are you trying to pioneer at GitLab Commit?

[01:02:06] SS: I think – Well, first of all, I hope we're doing a lot of the things that other conferences are doing right. I think hopefully you'll see that the venue is as interesting, as well

put on. We're trying to provide value for people that are practicing DevOps. We're trying to teach them how they can get better at that. We're trying to show what GitLab can mean for them.

We're also trying to connect people, connect people to each other, sharing best practices. But also we see this as a big collaboration. It's not us from the company broadcasting a message. It's also us receiving and getting better together and trying to live our mission of everyone can contribute.

[01:02:50] JM: Sid, thanks for coming back on the show.

[01:02:52] SS: Thanks for your questions. They were awesome.

[END OF INTERVIEW]

[01:03:03] 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.

[END]