EPISODE 952

[INTRODUCTION]

[00:00:00] JM: Crash monitoring emerged as a software category over the last decade. Crash monitoring software allows developers to understand when their applications are crashing on client devices. For example, we have an app for sSoftware Engineering Daily that people download on Android of iOS. Users download the app to their smartphone and when the user is playing an episode and the app crashes, the details of the crash are sent to a server that collects all of these crash reports.

Since our app is not perfect, it does crash, and these crash reports are extremely useful for helping us understand when our application is breaking on those client devices. This is really important, because there are so many client surfaces to test. You have iOS, you have a million different flavors of Android and all the different hardware configurations. You've got browsers. You've got the specific configurations of the browsers given those client operating systems. It's quite a detailed spectrum of different clients that you are serving with your application, and so you get weird crashes as a result of those different client devices sometimes.

As a business, crash monitoring is a category that has some similarities to log management, because there are lots of companies that offer crash monitoring, much like log management, and the problem is extremely heterogeneous.

At first glance, it seems like kind of a simple problem to solve. It seems like just a simple engineering problem. It's a market without winner take all or winner take most dynamics. But at scale, crash monitoring becomes a deeply complex engineering problem. From indexing, to database choices, to complex distributed systems tradeoffs. Crash monitoring is not a simple business and it promises to provide an extremely good business for the few companies who are able to out execute the crowded market of different crash monitoring providers.

James Smith is the CEO of Bugsnag, a company that makes crash monitoring and application stability tools. James returns to the show to discuss the growth stage engineering challenges of error monitoring and the business opportunities that come with them.

[SPONSOR MESSAGE]

[00:02:28] JM: This episode of Software Engineering Daily is brought to you by Datadog, a full stack monitoring platform that integrates with over 350 technologies like Gremlin, PagerDuty, AWS Lambda, Spinnaker and more. With rich visualizations and algorithmic alerts, Datadog can help you monitor the effects of chaos experiments. It can also identify weaknesses and improve the reliability of your systems. Visit softwareengineeringdaily.com/datadog to start a free 14-day trial and receive one of Datadog's famously cozy t-shirts. That's softwareengineeringdaily.com/datadog.

Thank you to Datadog for being a long-running sponsor of Software Engineering Daily.

[INTERVIEW]

[00:03:22] JM: James Smith, welcome back to Software Engineering Daily.

[00:03:24] JS: Thanks for having me again, Jeff.

[00:03:26] JM: You're 7 years into building Bugsnag. Let's start with a simple question. What's the biggest mistake you've made building this company so far?

[00:03:35] JS: The one biggest mistake? I think when you build a business as a first time founder, my cofounder Simon and I both – It's our first time being a founder in a business. I think that in hindsight, a lot of the things you do, you look back on and say, "I would definitely do that differently." There is a laundry list of things that if we were going to start again day one with the knowledge that we knew now, we would probably do differently.

I think with our business, some of the mistakes that we've made have been around – This is a bit of a weird one, but maybe being a big ambitious with product scope and new releases. I remember when we launched our – We called it internally dashboard two, which was a brand new dashboard that was powered by Elasticsearch and fully indexed under the hood so we could provide our search functionality.

We decided not only to change the database. We decided to also change the entire interface. We decided to move to React at the same time. We decided to throw in a couple of our first microservices at the same time. When we launched, it was exactly what we wanted it to be, but it was a big creaky for a while. We had some performance issues for a couple of months. We luckily were able to squash out the big ones first, but I think being ambitions on product scope is a lot easier when you are a company that doesn't have any customers. When you have customers, and especially in a business like ours, you're ingesting billions of crash reports per day. You have to make sure that when you cut over to that new system, it is smooth and it is reliable and it scales.

Yeah, I think we're less – I think when we're rolling out new technologies now, we try and do them one at a time or a couple at a time, and our ambitions get reeled in a little bit and product has to be more creative about how we break up and scope projects into chunks nowadays. Probably we've done that a couple of times now and we almost have a hard rule. It's not written down, which is like break things up into chunks. Don't rewrite the entire product.

[00:05:35] JM: Those different things, I mean, the Elasticsearch part of it sounds like you had to do that, but maybe not the microservices. Maybe not the migrating to React. Would you have done those things sequentially instead of all at the same time?

[00:05:49] JS: I think it would have been really difficult. The Elasticsearch part we had to do. The Elasticsearch part as well I think was only difficult because we ended up running a very large Elasticsearch cluster with very little Elasticsearch experience. You have to learn all the quirks of it. Throw more RAM at the JVM, all that kind of stuff. Once you know how to deal with the beast, it's fine.

I think what we tried to do is we tried to cut it over in chunks, and I think probably if we were going to do it again, we would design the system such we could cut over certain parts of it, test it, scale it and then cut over different parts in the future. For example, put a lot of data into Elasticsearch. Set up the indexes that match the previous Mongo indexes. Let that run and hum for a while while we're tuning it, and then maybe merge and roll out the React changes and the UI changes. That probably would have been the way to go. Yeah, all new technologies at once.

[00:06:42] JM: Tell me about building a developer tool in 2019. How has the environment changed and how has the company changed as you've matured?

[00:06:52] JS: Wow! It's night and day when we started the company. Simon and I quit our jobs at our previous company to start working on Bugsnag at the end of 2012. We incorporated the business in February 2013. In February 2013, in order to – It wasn't the recession, but in order to raise money for a developer platform, you needed investors that really understood what a developer platform was and why they matter.

Back in 2013, that wasn't the norm. It's taken a lot of, I think, wins over the years with Twilio's IPO and GitHub getting acquired by Microsoft for all VCs to realize there is a scalable business model behind these things. I think that, yeah, people – I mean, you even use the word developer tool as well. For investor perspective, the word tool to them implies this is a one-off thing. Something you get in, get out. This is not a part of your day-to-day. This is not a crucial part of your process.

Yeah, I think attitudes towards developer tools as a real productivity increase, and therefore something that is worth paying for has changed massively. I remember – I won't name names, but I remember the week after the Microsoft acquired GitHub. I got numerous VC emails coming in saying, "Hey! We have a new thesis around investing in developer tools," and a couple of these investors were investors that may have had feedback during our series A that didn't like the developer tool space.

Whether that is just unrelated feedback and they're just trying to be nice to us or whatever, but in reality, everyone now is onboard with the fact that this audience is really important and the software is eating the world and developers are in charge of that. Give them the best tools. The pickaxe analogy always comes to mind in the gold rush that people who really made the money with the pickaxe sellers. It makes sense in our space too.

[00:08:44] JM: Your investors are very strong. You have Matrix, Benchmark and Google Ventures for your – That's seed A and B, right?

[00:08:54] JS: That's right. Yeah.

[00:08:55] JM: How did you convince those top investors? Because I remember earlier on, I've been following Bugsnag since I started the podcast four years ago, and earlier on, I couldn't tell Bugsnag from the other 5 to 10 crash monitoring –

[00:09:13] JS: That's right. There were a lot of players in the space.

[00:09:16] JM: There are a lot of players. At this point you seem like you've pulled away from the pack. But earlier on, you did manage to differentiate yourself in how you presented to investors. How did you differentiate yourself? Because I'm sure you got those questions from those top investors. How are you different from X, Y and Z?

[00:09:34] JS: That's right. Yeah. I think there are two parts to it. I think there're a boring answer and a fine answer. The boring answer is that Simon and I cared about revenue from day one in the company. When we quit our previous job, we spent the first – I think it was 8 or 9 months bootstrapping the business. We bootstrap the business with an eye on revenue because we wanted to see if this was a thing that could make money.

Also, more realistically, I think I talked about this before, we were living in an apartment in the Sunset, me, my wife, Simon, his then fiancé, both of our cats, and we had to make rent. I think even Paul Graham talks about ramen profitability. For us, I think it was rent and sandwiches pretty much. We had a laser-focus on building revenue. When you go out to raise a seed round on an A round, you've got to prove that this is a business. Yes, there's a huge vision part on how you're going to make this a billion dollar business, but you have to prove that you can make money now.

We went into these meetings with Charles that look pretty nice. We were like, "This is how many developers are using us, and this is how much revenue we're making, and here are some cool logos that you may have heard of." Even I remember one of our really, really customers was WWE, the wrestling company. They weren't spending that much money with us, but it was really cool to have that on a deck and show that to investors, and I thought it was cool as well. That's the boring answer. The boring answer is every time we've gone out to raise, every time we've

gone out to pitch, we've had financials that looked appealing to investors. Fundamentally, that's what investors exists for. They exist a return for their LPs. That's the boring the answer.

I think the fun answer, which is the second layer on these decks and these conversations I had with these investors is – First off, there were investors, all three of the firms, Matrix, Benchmark and GV all understand this space. They're not people who just – They're not the people who emailed me later and said, "We have a thesis on this now." They've got it from day one, which is fantastic.

But then I was kind of explaining to them how as a software engineer, I don't write much code anymore, but as a software engineer, I would spent something like 40%, 50% of my time finding and fixing bugs. There are some software engineers out there that are very motivated to do that and spend their time doing that, and I have met them, they do exist. But most people, especially product-centric software engineers want to be building features, want to be building products.

Pitching the argument around if technology is key to your business and every company is adapting technology to differentiate, then you need to have the best tools and the best software available and your customers will notice if it's broken or is it bad. It's kind of obvious now that we've got to this stage and size that we're at, but at the time, I think you have to paint that picture for the investors. This is a very costly thing in terms of revenue if you're causing customers to churn, for example, and also in terms of like churn of software engineers who don't want to work on fixing bugs. Yeah, a bit of we're making money and it's up into the right and a bit of this product vision matters in the new world of software.

[00:12:35] JM: But I'm sure they looked at the other providers. I mean, it's always interesting, because – I mean, you might be the Datadog of this space. Datadog, I'm still trying to unpack why Datadog was the company, has become the company to beat at logging. There's more a logging provider. There's probably 10 times the number of logging providers that there are for crash reporting. I don't know why Datadog pulled away.

[00:13:06] JS: I love Datadog. We spend a large amount of money on Datadog per month in our infrastructure budget, but it's not one of those products where you spend that money and you go, "Ugh!" There's some log management providers, which I won't mention the name of. We

hear time and time again that people are just spending – I don't know. I guess. It's the Splunks and the Sumo Logics of the world, where there's terabytes of data going into these things and there's no insights coming out.

None of the software team, none of the developers or product teams have logins or credentials to the system and you need to have a degree in quantum Splunk-puting in order to understand how to extract that data. Genuinely, I forgot what the name of it is, but there's a qualification you can take, like Splunk certified systems engineer or something like that.

It's not accessible, and so there's this – I think you regret. Not regret it. You don't want to spend all that money for things that aren't accessible and people don't want to go into that system. I think if you look at a Datadog, especially because the thing that I think attracted a lot of people to them was the StatsD metrics side of things in their product.

Innately, out of the box, you got charts that show you the things going up and down, the alert side of the box. There wasn't any – You didn't have to have a degree to configure the settings of these things. You got a lot of wild factor out of the box and it's a product philosophy that we agree with as well. Specializing on something, doing something really well and having an opinion, but making everything extensible. I think with the audience that we share, the software engineering and product audience, you have to do that. You want to be able to configure things, but you don't want to spend your life configuring things. At least at that level we share a philosophy.

One of the things that we talk about a lot as a company is the concept of software stability, and I think that, as you said, 4 or 5 years ago, there were a lot of different what used to be called error management, error monitoring solutions in the space. We realized we were kind of at a fork in the road. There were companies that were going to go towards being more like APM companies, and APM companies have historically been bought in by infrastructure, or SRE, or DevOps teams. We decided to go at a different direction.

We decided to say, "Look. Actually, what we're measuring is stability. Stability can be measured with a single number that you can share between product and engineering and it should be at all for the product and engineering team, not for the infra ops and SRE teams." We designed our

entire product around getting information and answers to that different audience. I think that's maybe why we've board away a little bit.

[00:15:38] JM: The distinction between APM and stability reporting. I think of an APM tool as like I'm looking at dashboard. I'm like seeing line charts go up and down.

[00:15:51] JS: Aggregates and averages across the board. Obviously, the APM, the big players in the APM space, the New Relics and the App Dynamics of the world, they have large product suites now. They do a lot of things, but their core products and their roots I think come from the DevOps scene, and in that space, you're effectively saying, "Detecting when something is broken and escalating it to someone. We don't believe in the philosophy.

We think the we should help you understand why something is broken and who the most valuable customers are that are affected, or exactly what the steps are to reproduce and therefore fix the bug. That's not a philosophy that's shared by the APM players.

I mean, this is, again, historical thing. But a lot of APM tools, like a New Relic for example, are adapted by the CIO and, sure, there's a component of understanding customer impact and things like that. But quite, they're adapted to optimize and maintain cloud spend. So by spending however many thousands a month on a New Relic, you can see which functions are slow and therefore which functions to optimize first to reduce your number of AWS instances that you have spun up.

There is a ROI component that's kind of purely mathematical, but none of that points to this customer is seeing a problem in here's the exact steps to reproduce, or tying software issues really tightly to your project management tool, like a JIRA, for example, where you're planning and running you your projects. Yeah, fundamentally, I think APM is for people who are running applications and stability management is is for people who are building applications.

[SPONSOR MESSAGE]

[00:17:41] 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]

[00:19:31] JM: Integrations, that's another thing like when I try to solve the Datadog mystery, why Datadog. Integration seems like one component of this sort of attention detail kind of thing. They just build all the integrations and made them work well and then they built some kind of scalable system for maintaining those integrations, which to me seems hard. I think in an ideal world, all the providers that you're integrating with would never break compatibility with API changes, but I imagine that that does happen.

[00:20:06] JS: Yup.

[00:20:07] JM: How do you manage the swath of integrations? You got to integrate with Slack, and Asana, and this, and that, and everything.

[00:20:13] JS: We have about 40 different integrations right now from over 40 integrations, and we don't officially tie them out like this, but I think the way that we do this is we have integrations that really, really matter to the business, and there's may be five. There's a common suite of things that pretty much all of our customers use. They all use Slack. They all use JIRA. They all use PagerDuty. They all use GitHub. They all use an SSO provider and there're basically two that matter.

There're maybe 5 or 6 integrations that are completely essential to our business. For those, my engineering team as based out of Bath in the UK and they build up integration services. Java microservices and things, a lot of Go being written right now in the suite.

The metrics and the detail that goes into when something goes wrong on those integrations is insanely good, and a lot of it goes in the Datadog actually. So we know, we'll get an alert from Datadog if suddenly we get an increased 500 rate form JIRA, for example. And a lot of the time it might just be because our third-party services API is down, but sometimes it will be like – For example, recently, Atlassian deprecated some of that APIs and authentication techniques. They sent out a notice, but it didn't come to the right email address. So we caught it and patched it really quickly based on the metrics that we have.

I think that integrations providers are pretty good at not hard to deprecating old APIs, which means that for the remaining 32 to 35 integrations that we have, we don't need to – We can leave them kind of as is. We can keep them running as they're going right now, and a lot of time they disappear. We had a HipChat integration for a long time, and now HipChat is not a thing. It kind of self-deprecates itself. It disappears.

Yeah, I think that –

[00:21:58] JM: I wouldn't count that happening very often.

[00:22:00] JS: You'd be surprised. When get to the long tail, when you do have 40+ integration, there are companies that come and go, especially – We were talking about how crowded certain spaces are right now, a lot of log management. Project management and issue tracking is probably one of the most crowded spaces out there.

[00:22:15] JM: People actually churn out of that one.

[00:22:17] JS: Big time. Big time. Yeah, there are times where we are like, "Okay. They've said they're shutting the service down. Great. We don't have to maintain the integration anymore."

[00:22:24] JM: Do people churn out of old error monitoring tools?

[00:22:28] JS: Absolutely. We scoop off a lot of customers that way.

[00:22:30] JM: Really?

[00:22:30] JS: Yes. You know this space well. You can probably imagine which players were big in the past don't' really show up much anymore. Yeah.

[00:22:43] JM: It's interesting, just having this conversation, I'm observing – I feel like seven years ago, probably the time when you started this business. If we were having this conversation about all the tools that you use and all the things that you have to integrate with, the conversation will be much more about open source tools and libraries, programming languages.

Now we're talking about SaaS tools. It's like your business – Businesses have become, instead of a composition of open source tools, it's a composition of SaaS tools. Do you have any just observations on how software development has changed with just the fact that we're now software development is the process of SaaS composition, rather than open source composition.

[00:23:24] JS: I think that the open source composition layer is never going to go away, and I think that especially when it comes to the infrastructure that you're building your business on, a lot of businesses I really, really respect are built on this open source mentality, like Docker and HashiCorp, for example, are both building the infrastructure layer in an open source manner and then monetizing via services and then premium features on top of that. That makes a lot of sense to me as a former CTO in terms of the ability to switch out.

We've done at least once in the past – In my previous company, we moved from dedicated service to AWS. Bugsnag, we moved from AWS to GCP, to Google cloud's offering. By running on open source while supported technologies, you can make those – Previously, these used to take months and month and months to make these kind of moves, and you can make those moves in a relatively quick and easy way. It's still painful a little, but you get the benefits of switching provider without all of the downsides.

We tend to adapt open source infrastructure, and I think that's why it's so popular across the board. when it comes to things like – I do not think a core to the business, not core IP. Even 5, 10 years ago, I think, in this space, error monitoring, people would roll their own. There are customers that we have and still today there are customers that we talk to have their own version of Bugsnag. Always, it's horrendous, obviously because it is built by a developer on their side time on a weekend project and they bring it in to solve a problem that existed maybe before Bugsnag existed or maybe because they hadn't discovered Bugsnag yet.

We come in, and more often than not, the developer who built that, there's no politics there. They're like, "Thanks goodness that someone else has come in and built this with a team of fortysomething people to build and maintain this rather than me having to do it on the weekend."

I think when it comes to things like monitoring, and payments, and authentication or all these kind of pluggable things, I think that you're less bound to the infrastructure. You can, in theory, swap them in and swap them out, and then it's on ours as a product company to make sure that our product is good enough, there's a sticky solution that people don't churn off. It's less sticky because of the – You've built on a proprietary technology, but is sticky because the customers like the product. I think that's the thing. It's not a clean line, but I think there is a distinction between infrastructure and supporting services.

[00:25:49] JM: Why did you move to GCP?

[00:25:51] JS: That was a fun one. I think most of the reasons that we moved have actually been addressed by AWS now, which is slightly annoying. But I'll get this wrong, because my cofounder and team did the move. But there were few things like AWS used to shutdown instances when there was a problem with an instance or a security patch had to be applied at the kernel or whatever. They would send you an email saying, “We’re going to shut down this instance. Make sure you move it across from A to B or whatever.”

For most of our services these days, they’re in Kubernetes clusters. It doesn't matter where they’re running. But for our databases, we are not putting or currently putting our databases in a Kubernetes cluster. We’re manually managing the primaries and secondaries and failover and all that kind of stuff. If AWS happened to want to shut down your database primary, then you had to stop on what your infrastructure team is doing and work on flipping it over.

GCP have magic migration under the hood. They have a layer of abstraction that means that you never know when an instance goes away. That was a huge win. The other thing I think that GCP has had going for, I’m not sure if AWS has improved this yet, was the way that you could reserve instances. With AWS, this is with marketplace concept, where you had to reserve, and then if you change the size of instances you are using, you would then have to sell your instances that you bought previously, reserved previously on a marketplace. It’s like we didn't get into building a company to be eBay sellers. It doesn't make any sense.

GCP basically just, again, abstracts away and you just get volume discounts for using more services. They were the two major things, I think, that made us make the leap. Also Google was really early on on Kubernetes as a service as well. We’re heavy into Kubernetes. That was kind of the three reasons.

[00:27:33] JM: You manage your own databases?

[00:27:35] JS: We do. We kind of have to. I kind of mention this earlier. We process something like over a billion crash reports per day. Crash reports are not small. It’s not just integer that

we're commencing. They are between 10 and 500 kB. I think averaging something like 50 kB these days with a lot of mobile payloads. When we get a crash report, it has a Stack trace. It has a full thread dump. It has diagnostic information. It has what we call breadcrumbs, which is a list of user actions that led up to a crash. It's quite a lot of text and data that we have to process and store.

I think this is true. We're still the only provider that stores and indexes all of these crash reports. I think a lot of the other players in the space take the stance of, "Hey, we can just toss some of these away and do some sampling and hide that for the user." We store and save everything.

We're also processing something like over 7 billion sessions per day. On a client-side application or, for example, every time you open the app, we want to detect that a session has happened so that we can give you a baselined stability score. Rather than just saying you had 20 crashes today. We can say 99.9% of your sessions were crash free, which is they had a baseline numbers. It's an apples to apples comparison even across multiple applications.

We're processing billions and billions and billions of data points per day, and the databases as a service don't really work well at that scale. We have a very strong gross margin business and we're able to maintain those strong gross margins by understanding databases and how to scale them. I think maybe – We're always looking at this. Obviously, we don't want to run databases. It's a skill that we kind of built via necessity. Again, coming back to that bootstrap mentality and that revenue-centric mentality I said earlier, we're always going to keep an eye on the cost dynamics of these kinds of things. But if you think about it, these businesses that run databases as services need to make a profit. They need to make a premium over the baseline.

For us, we have a relatively small infrastructure team. We have – I think it's three now. Actually, someone literally just started today, the third person on the infrastructure team. So we have three people on our infrastructure operations team managing this huge Kubernetes cluster and this huge Elasticsearch cluster and this huge MongoDB cluster and they're doing a great job of it. We don't need to move yet. If the cost dynamics come down, then we'll probably think about it.

[00:29:59] JM: Where did the managed databases fall over?

[00:30:02] JS: You get excellent economies of scale. But in cloud computing land, there is a limit. It's not like if you're pricing something on a per seat basis. It's zero marginal cost to discount on additional seats. But in cloud computing, it always comes down to CPU and memory usage under the hood or actual service with server time that you're leasing out to customers via instances or whatever. The economies of scale aren't there.

[00:30:30] JM: Wow!

[00:30:30] JS: It's a significant difference as well. It's not just like a 10%.

[00:30:33] JM: Okay. It's not that they don't perform well. It's literally the economics.

[00:30:38] JS: Yeah. I remember, this is really early on at Bugsnag. At one point, we still run on MongoDB for our canonical document store. Every crash report that comes in is stored in MongoDB. We do that because it's fairly unstructured data. It's structured but the Android crash report differs wildly from a Ruby on Rails crash report, for example.

That made sense at the time and it still makes sense, but I remember, we went to a Mongo conference and we found out at one point that we were running a Mongo cluster incomparable size to Facebook. Yeah, that's mainly because Facebook weren't using Mongo as their main database, but they were using for pause and other things under the hood. Yeah, we were like, "Oh wow! Okay."

If you look at a lot of these databases as a service providers that really, really optimize for people like I'm doing a CRUD application and their whole business model on pricing model is optimized around that. When you get into big volumes and big scale, I think almost everyone runs their own database instances at that scale. Yeah. It seems to be working out so far. I'll report back next time.

[00:31:47] JM: Have you looked at anything else like Cassandra or looked at relational databases?

[00:31:53] JS: We did. At some point, I imagine. We might do. Now we have Elasticsearch in place as our search indexing store. I think we may split out our Mongo database into a relational database and then maybe a flat file store. Mongo, effectively, there's a few indices in there, but the main index is lookup crash report by ID.

Do you know what's really good at doing that? File stores. S3. That's exactly what S3 does. That way, we would not have to run a huge cluster with MongoDB. We could just put it in S3 or Google's equivalent of S3. But our Mongo store right now is crash report data plus the relational layer of users and accounts and projects and all that kind of stuff.

[00:32:38] JM: What's your aching layer?

[00:32:39] JS: We have Redis for all of our caching, but we have a lot of queues these days. I'm going to get this wrong. We have RabMQ, but we also have – Oh! Kafka. That's right. We have RabbitMQ and Kafka. We use Redis for queuing at one point, right at the beginning of the company. Then we got to the point where we needed to guarantee ingestion, or no back pressure on ingestion.

We needed to make sure that if one of our services fell over, that we could still back process the events that came in. Again, we're processing a billion crash reports per day. That back pressure fills up pretty quickly. Imagine getting a clog in your pipe, in your toilet. It's going to come back to the client.

[00:33:18] JM: It's a very busy toilet.

[00:33:20] JS: It's a very busy toilet and a lot of our customers. Yeah, a lot of our customers would be really mad if they lost some crash reports. So we can't do that. Everyone jokes with me all the time about, "Hey! Who monitors the monitoring solution?" We effectively have to –

[00:33:36] JM: Datadog.

[00:33:36] JS: Yeah, exactly, or Datadog for a lot of these stuff.

[00:33:43] JM: This show is brought to you by Datadog.

[00:33:45] JS: That's right. We're going to get commission I think on this. We have obviously a SaaS offering of Bugsnag, which is where everyone signs up for trials in kicks the tires and most of our revenue comes from. But we also have an on-premise version of Bugsnag. It's the same product, but you can install it on your own servers. That's more for compliance reasons and all that kind of stuff.

We actually run our own on-premise version of Bugsnag as well as our cloud instance, because if there's a bug in our cloud instance, we actually report the errors to a separate instance of Bugsnag, which is our internal on-premise instance of Bugsnag.

[00:34:17] JM: That's a nice dog fooding.

[00:34:18] JS: It's dog fooding and it's also a decoupling on those also. Yeah.

[00:34:22] JM: The on-prem solution – It's my cat. Cat doesn't like the on-prem solution. What was hard about building that? I mean, you lose operational control.

[00:34:34] JS: Yeah. On-prem, you do not get the visibility of what's happening. That's the main frustration. With our SaaS service, we're able to see – Measure the success of our customers a lot easier. So we can see if someone is adapting, adding more users to their account, or we can see if someone's set up a Rails app and then maybe they look to our mobile offerings and add an Android and iOS app at the same time. We can even, with the permission of the customer, obviously, we can go and have a look at their account and recommend things that they can improve to improve the quality of their experience inside of Bugsnag.

With on-prem, you can't do any of that, and it is by design. It's a black box. We allow people to run Bugsnag behind their firewall. No phone home. So the hardest thing I think is the customer success aspect. We have to be really intentional. We have to almost force people to have an on-site QBR quarterly business review where we come on-site and talk about how – Look at the dashboard on their premises and see how it's looking and recommend improvements.

In terms of the tech side of building it, we cobbled together our own kind of on-prem, I guess you'd cooler cluster management thing when we launch Bugsnag on-prem first, and it was great. A lot of our early on-premise customers loved using it. We moved about a year and a half, two years ago, to actually a service code Replicated. Replicated is a service that helps you do container orchestration in on-prem environments effectively.

[00:35:57] JM: They just became a sponsor of the show.

[00:35:58] JS: Did they? Really.

[00:36:00] JM: Yeah.

[00:36:00] JS: There you go. Thumbs up to Replicated.

[00:36:01] JM: You are doing a great – Datadog has been a long-time sponsor.

[00:36:04] JS: I'm accidentally giving props to all these sponsors.

[00:36:06] JM: This is serious sponsor content.

[00:36:08] JS: Yeah. Hopefully. When you have Replicated on me now, they can give us props the other way hopefully. Yeah, it takes all of the pain of orchestrating containers on an on-prem environment away, things like license management. Yeah, it's not a huge part of our business in terms of revenue, but the customers that are using it are – I can't talk about them. I wish I could. You know their names. They are very, very big companies with very specific compliance requirements. So it's a very strategically important part of our business.

[00:36:40] JM: It has been an interesting downstream effect of Kubernetes is the fact that there are these companies now that do the distributed systems delivery process. It is basically the distributed systems equivalent of installing a CD on your computer. That's what Kubernetes allows, packaging.

[00:36:59] JS: When you think about Kubernetes and all the great stuff it brings. But we actually didn't adapt Kubernetes for a while. We're all in now, and it's great, but you really need to know what you're doing running a Kubernetes cluster. There is a lot of stuff going on. If you're trying to take all the benefits of a Kubernetes cluster and deliver them on to you customer's infrastructure, yeah, you need to abstract that away. We do not want our customers calling us up and saying, "Hey! How do I run this Kub control function, whatever." They don't care about. They're using Bugsnag because they want us to solve their problems, not create problems.

Yeah, container orchestration and Kubernetes management is – I mean, for us, on SaaS, we use Google's Kubernetes environment. It is pretty nice. We get a lot of out of the box and we don't feel like we're tied into some proprietary thing like Amazon have that container service that's not Kubernetes.

[00:37:55] JM: ECS.

[00:37:55] JS: Yeah, that's right.

[00:37:55] JM: But they also have EKS now.

[00:37:57] JS: They have EKS as well, but I'm not sure if they are all in on that. They're running both still. Google basically said, "It's going to be Kubernetes. This is how we're going to do it.

[00:38:11] JM: Right. I think that's more of a legacy aspect.

[00:38:14] JS: Yeah. It might be just because they have so many customers on ECS that they have to support that environment. You're probably right.

[00:38:19] JM: Well, because they – ECS was an interesting bet, because they made that during the contain orchestration wars when we were unsure of Mesos was going win or Kubernetes was going to win.

[00:38:28] JS: What is it? Docker Swarm had a go as well.

[00:38:29] JM: Docker Swarm. What are the dos and don'ts of marketing a developer product? I mean, there are so many ways you can spend money. You can go to AWS Reinvent and spend your entire marketing budget, or you can spend zero money like Stripe.

[00:38:44] JS: Yeah. You've been to Reinvent, I'm sure.

[00:38:47] JM: Actually, I haven't.

[00:38:48] JS: You haven't.

[00:38:49] JM: I'm thinking of going this year. I know it's like Mecca. It's like the worst and the best of Mecca. I mean, it is a –

[00:38:57] JS: It's everything everyone's told you it is. You go there and there are bajillions of software engineers and product people all at one place, and it's this heaving with people and there is – It's in –What's it? In the Venetian, the conference floor. There are people who have got a startup and they're like, "Right. Marketing says we need to make a splash. So we need to spend $20,000, $50,000 on a booth at Reinvent."

[00:39:22] JM: Dude! That's not going to make a splash at Reinvent.

[00:39:24] JS: So you end up at Reinvent and you've got a little corner 6x6 and you're towered over by someone else. I mean, I joke about this. We've certainly done it and we've done Reinvent before. Maybe we'll do Reinvent next year again. But I think people talk a lot – I mean, brand is very important, obviously. But you go to be really strategic about how you deploy that. If you go there at Reinvent and you spend – $50,000 for a startup is a lot of money. But at Reinvent, it's irrelevant. So you end up in a corner and you've got these tens of thousands of software developers walking around. Maybe 3% of them will pass your booth.

I think a lot of people make that mistake. They're like, "Let's just get a booth at a big conference." For us, events are a really key part of our strategy. We do 10 to 15 events per year. We just hired a new event specialist who is going to be ramping up our events for next year as well. We've historically done a lot of a grassroots platform driven events. For example, this year,

I think it's in November. We'll be at Droidcon SF. Droidcon SF is it's not a huge conference, but if you look at the quality of the speakers that goes to that conference, it's just absolutely excellent.

[00:40:28] JM: You said Android Conference?

[00:40:29] JS: Android Conference. It's the non-Google Android Conference. It's not Google IO and it's accessible. They do three joint confs. I think there're New York, Boston and San Francisco in America, North America. We go to all three of them and we get a booth every time. I thing with pretty much the de facto commercial stability management tool in the Android space and in the iOS space. Part of it is meeting customers.

I remember, actually, last year's Droidcon SF, we had a booth. I went down because it's just down the street. So I went and said hi to everyone. Airbnb is a customer and Slack is a customer and they both had speakers. They've stopped by our booth to say hi. We found out after the Airbnb talk that the speakers from Airbnb had namedropped Bugsnag in the talk, because suddenly all of these people who are in the total were like, "Hey, we're figuring out how to manage stability in our application as well," and we were just swarmed at the booth.

We like conferences where there's may be 5 to 10 exhibitors there and there's a really good audience fair and we can actually genuinely spend time talking to people about what their problems are and how we might be able to help.

Having said that, now we have an event specialist. We're getting a little bit more ambitious. We're doing GitHub Universe this year, and I think – I'm not sure if I can say what we're sponsoring yet, but we have a cool – If you're at GitHub Universe, swing by. We have a really cool different sponsorship opportunity.

[00:41:54] JM: I wouldn't be surprised if GitHub Universe was still underpriced, because that's kind of like – To me, I haven't been there. But from what I've seen and heard, it's an nascent reinvent.

[00:42:03] JS: It's getting bigger now. They always pick an amazing venue. I'd give GitHub props, and that their events team really care about the experience. I think that maybe AWS Reinvent did at the beginning, but they can't now.

[00:42:17] JM: No, they can't.

[00:42:18] JS: It's just a car park full of booths at one point. Whereas the GitHub Universe – Again, I'm not sure if we're able to talk about this yet. But it's not just the table. We have a really cool thing that we're doing at GitHub Universe.

[00:42:29] JM: Is it bouncy castle?

[00:42:35] JS: You asked about dos or don'ts for marketing for developers. I'll tell you what. Being condescending is a hard no for us. There are a lot of people who don't – Simon and I are both math and computer science majors. We're both product engineering people. We've been building software our whole lives. Every time we think about how do we want to talk to people about our product, we think, "What would make us happy? What would make us excited?" My marketing team is super tuned into this as well.

You see people given out – I sort of tweet about this the other the day. Someone at conference was giving out Nerf guns in a conference. I'm like, "Why are you treating software developers like teenagers or kids? It's so condescending." It's like, "You know what? Just go there. Be classy. Have great conversations and be genuine."

I think if you do that and you understand how to be genuine with the audience, which I think comes naturally to us. I think you get results and you get success. I think that applies to events. It applies to advertising. It applies to podcast sponsorships. It's not like we do a hundred of the podcast sponsorships. We picked the ones where we like the content and we think the content is genuine, because we think it aligns with our brand. That's being genuine and not being condescending are probably two major rules there.

[SPONSOR MESSAGE]

[00:43:55] JM: As a programmer, you think an object. With MongoDB, so does your database. MongoDB is the most popular document-based database built for modern application developers and the cloud area. Millions of developers use MongoDB to power the world's most innovative products and services, from crypto currency, to online gaming, IoT and more. Try Mongo DB today with Atlas, the global cloud database service that runs on AWS, Azure and Google Cloud. Configure, deploy and connect to your database in just a few minutes. Check it out at mongodb.com/atlas. That's mongodb.com/atlas.

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

[INTERVIEW CONTINUED]

[00:44:52] JM: We were talking about before the show that you used to work at Heyzap, which I didn't really realize until I interviewed Jude, who runs Golden now, which is a really ambitious and interesting company. Heyzap was an ad tech company. Do you actually learn anything about advertising at an ad tech company or is it more like you just learn about auctions?

[00:45:16] JS: There are a lot of things that you learn about. You don't necessarily learn about the whys behind why people are buying ads or why they're doing the particular behaviors that they're doing. Also, Heyzap was interesting, and that when I joined in 2009, it wasn't an ad tech company. It started off as, "Oh! This is going to really age the company now."

It started off as an embeddable Flash arcade. So you could take your Flash games – Yeah. Flash don't even really exist anymore, but you could take your Flash games that you were playing on Miniclip or Kongregate or whatever and embed them into an arcade and put them on your website. Then it evolved into a mobile game recommendation app. So the core throughout the entire life of the company was recommending other games that you could play.

Naturally, it evolved into an ad tech company, because if you know a customer likes playing game A and B and you can then predict they like game C, it's a massive benefit over. Just random ad presentation. We learned about ad conversion. We learned about how we could use recommendation to improve the quality of ads. I don't think I'd work in the ad tech space again, but I think it's nice to have that recommendation stuff, because not only do you get high-quality

conversions on your ads so it's more money for everyone, but also you're giving ads that customers like to see. It's not an industry that I think is exciting to me anymore.

Yeah, I didn't really learn a lot about ads. I'll tell you what we did learn about though, scale. Some of the things – Simon, my cofounder, I brought him out to San Francisco. Convinced him to join Heyzap. He worked at Heyzap for a year, year and a half to run my mobile team at Heyzap, and convinced him to come out. He would also work on some infrastructure projects and architecture projects. Holy molly! The amount of traffic that comes in from an ad company, we joke about how Bugsnag is effectively we're being DDoS'd all the time, because our client is inside of all of these major brand name applications like Airbnb, and Lyft, and Pandora and all these kind of apps.

If any of these apps crash, we have to take the brunt of all of that. So it's not like we're being attacked, but it's the same traffic pattern as at DDoS. Ad tech is exactly the same. If you think about it, we learned a lot about scale. We learned a lot about how to make SDKs. We built SDKs for these ad folks, game developers and publishers that would present the ads. But also, you had to report back every single ad impression. That's a lot of data points coming back. I learned a little bit about ads, but I learned a lot about scale and I learned a lot about mobile SDKs big time.

[00:47:45] JM: Bringing this back to Bugsnag, the scale that you deal with, does that require you to make some kind of negotiations with the cloud provider, like – I don't know. Can we get bulk pricing or burst pricing or something?

[00:48:01] JS: It's funny because – Again, I'm not trying to advertise GCP. GCP has a really nice discount curve. It seems to work pretty well and they have nice reserved instances or whatever they call them over there. The funny thing is even though we have all these scale, we're pretty efficient about how we run our infrastructure.

When I hear rumors of how much some other Bay Area unicorn companies are spending on their cloud instances, it's whispered and nothing compared to them. I wonder if we haven't really worked on trying to get the discounts directly, special discounts. But I imagine we're a small fry

compared to an Airbnb, for example. I imagine Airbnb has that cloud and the amount of spend where they could come in and say, “Let’s get a discount here. Otherwise, it’s going to move off.”

I think the good thing about having these three major cloud players now is that there’s a lot of competition. I love the fact – AWS does this all the time, but GCP, I get the emails where it’s like, “Hey! We just made everything cheaper.” I’m like, “Huh! Wow! We get that without opting in. It just happens.” Everyone’s trying to catch up with AWS. So there’s a lot of competition in the space. I think they’re well-incentivized to make sure the pricing curves are really, really well-tuned. We don’t want to have to be negotiating with these guys either. They save sales budget and we don’t have to waste time talking to them as well.

[00:49:21] JM: How do you see Google Cloud and AWS diverging in the future? What are their core competencies?

[00:49:27] JS: That’s interesting. I like the way that Google is focusing on developers, and you think everyone is focusing on developers? Actually, Microsoft is starting to take a stab at this as well. But I think that AWS started off being very obviously focused at infrastructure teams, whereas Google have built developer platform layers on top of that. So they have a mobile developer platform with Firebase. They’ve been acquiring in actually a lot of companies that have helped them build this portfolio of strong developer projects.

But they’re also obviously developing Kubernetes, which is a big fan favorite in the developer space. They’re also developing Spinnaker, just an up and coming technology that I think is going to be really big. Everything gets put back to the community. You don’t really see that with AWS. It’s not really – Sure, they open source stuff, but it’s not really part of their DNA.

Yeah, I’m a bit biased, because we use their service and Google Ventures is an investor in us as well. But as a developer, I think that one of the big three that’s putting the most effort into winning the developers I think is Google. Yeah, they’ve got to find an angle, because AWS is such a behemoth. They’ve been just a huge company that got in early and they’ve got to find angles to win. I think that’s one they’re going for.

[00:50:40] JM: Did you ever take any acquisition offer seriously?

[00:50:44] JS: We get people knocking on the door. Simon and I talk about the motivations for building our business. When we've got a new hire that's joining or we're interviewing someone, these questions come up all the time. We're productized. We're motivated by building something. So we're builders.

So right now, what we're really focused on is just building the absolute best product that we can. One of the things that I think is frustrating about building a business is you always want to do more. We're 6, 7 years into building this company and the roadmap just keeps on getting longer. There's so much stuff that we have to build. I think right now we're just focused on building out that roadmap.

24 months ago, we just changed our product and our positioning around the concept of stability instead of error monitoring and we're like, "Great! Let's double down on that and let's move on to the next thing." But then as soon as we did that, all of our customer are like, "This is great. We love this." Can you do this? Can we do this? Can we do this?" It's exciting as a product person. It's obviously exciting to have that roadmap ahead of you, but it's almost like that's the only thing we can focus on right now. The business is growing well. We're getting excellent logos and excellent customers, revenues growing. Just continuing to build the best product, because we know there's a long way to go still, is kind of taking up the most time. But we've had – There're people who knock on the door. We've had conversations, but it's never been exciting for us. Building the business out, especially even the growth we've got in the moment is the focus.

[00:52:20] JM: I wonder how many years until you've got a Datadog-sized opportunity. I mean, to me, it seems inevitable. I think today, the amount of data that's going into logging is obviously some multiple of the amount of data that's going into a crash monitoring tool. But given the increased number of companies that are going to need crash monitoring, eventually the volume of data is going to be the same that Datadog is ingesting today.

I imagine your per bit price of crash monitoring data, you probably have better margins than the per bit price of a logging provider. You think about the unit of economics of your business compared to Datadog.

[00:53:14] JS: Well, it's interesting as well. The way I think about it and the way a lot of our customers talk about this is the price per data that they actually look at. Like I said earlier with these logging solutions, I think a lot of the time there's a black hole of data. You're just dumping data, terabytes and terabytes of data into a giant file effectively.

With Bugsnag, yeah, we're collecting a lot of data, but we're applying a lot of meaning layers to it off to be ingested, which is – To be honest, that's a bulk of our – I think of our cloud spend is that processing and storage of these crash reports in a meaningful format. Yeah, I think that people tell us all the time. I said this about Datadog, StatsD engine. I forgot what the product is called now. If you get the data into a meaningful, useful format, then you're going to feel better about paying for this kind of stuff. Yeah, we do price based on the volume of crash reports that you send in.

If we are pricing based on the volume of data that you're sending in and our customers know they need all that data and happy about that data, that's a much better position to be in than these kind of legacy log management solutions that you're annoyed about how much data is being pumped into these systems, because it costs you a lot of money.

Yeah, for us, I think it's about awareness and brand building. I think a lot of people have legacy systems in place that they've cobbled together, like I said. Once people get to the position where they're like, "We absolutely need this. This is crucial to our business." Over the past 6 or 7 years, the tide is turning. So I'm hoping that continues, in which case, yeah. I think we could be. I think we could be a Datadog-sized business.

[00:54:44] JM: Do you think you'll have to raise money again or is it straight path to IPO at this point?

[00:54:50] JS: I think on the path to IPO, people raise money. The right motivation to raise money is when you have something that is working that you want to put in on a flywheel. I think it's probably likely that we'll raise money again. I can't really talk about when I'm thinking about doing that, but there are things in our business that are working really well and we obviously want to double down and invest on it.

We've been able to increase our spend pretty regularly, things like marketing spend. We've increased our marketing spend. We're very ROI-centric on marketing. We've been able to increase it from the original budget for 2019 I think three times already this year. I think we're able to be lean and reinvest the money wisely. Yeah, when you find engines, you want to put fuel into those engines.

[00:55:34] JM: Is there any conventional wisdom about building a software company that you have avoided?

[00:55:41] JS: I'm not sure about avoided. I think there are things that we've considered, especially things like pricing models. There're a lot of people who might expect our pricing to be like a pure per seat pricing, like a Slack. You pay per developer using their product. We definitely are avoiding doing that. We're closely to an AWS utility-based ingestion, ingestion-based model, than we are a Slack.

I think that a lot of companies spend a ton of money on marketing and will raise – There are a lot of companies who raise a ton of money and invest it without thinking about ROI. You've seen this. The classic Bay Area venture-backed company where they are, "Oh! Let's go and raise this huge, massive round and then let's just throw it to AdWords to juice on numbers." It's just not really in our DNA to do that. We invest heavily where we see ROI, and it's working, and it's paying off. We're going to continue to do that, but we're not going to take the spray and pray approach to growth, to hope something sticks. I think maybe that's the way we've avoided doing things.

[00:56:43] JM: Last question. Building a business in San Francisco, overated or under?

[00:56:48] JS: That's a really good question. I think we're well-placed to discuss that, because we have a 40somethign person company and half of our company is here in San Francisco and half of our company is in the UK and in Bath, which is where our engineering team is based. There is a reason two guys with a British accent are starting a company in one of the most expensive cities in the world.

Some of the best product and engineering teams, the most forward-thinking product and engineering teams are here in San Francisco or down the peninsula. Our product has evolved rapidly based on our conversations with these customers. We're just down the street. We're right next to Twitter right here. We're just down the street from Uber. We're able to go on site with these prospects or we're able to go to Airbnb and we tell them, "Hey! Use us as a therapy session. Moan us. Tell us what sucks about software development. Tell us what sucks about Bugsnag if you think anything sucks about Bugsnag."

Rather than being a company that's halfway across the world that's just going back and forth of the support tickets, we then come back. We discuss it as a product team. We productize our roadmap based on the things that we've heard, and I think that's what's driven our direction of our product.

Our engineering team however is in Bath. The engineering team is absolutely fantastic there. I was just in the UK two weeks ago spending time at the office. We have access to world-class talent. We've got people who've got 10something years of software engineering experience. They're building scalable services in Java and Go. Maybe Java is the not the sexiest language these days, but it's really appropriate for what we do. So they're building the services.

[00:58:20] JM: Sure is.

[00:58:21] JS: I think if you're working in the developer space or if your customers are going to start off as technology companies, which our original customer base was, you probably need to spend time with those technology companies, and a lot of them are here. That's why as a founding team and a sales and marketing team, we're based here in San Francisco. Engineering, I think you'll see this on most companies these days. It can be anywhere in the world as long as you work great management and great communication and great collaboration. That's the way I think about it at least. I'm a bit biased there though.

[00:58:49] JM: James, great conversation. I really enjoyed this.

[00:58:51] JS: Yes. Fun as always. Thanks for having me, Jeff.

[00:58:53] JM: Okay. Thank you.

[END OF INTERVIEW]

[00:59:03] JM: Monday.com is a team management platform that brings all of your work, external tools and communications into one place making cross-team collaboration easy. You can try Monday.com and get a 14-day trial by going to Monday.com/sedaily. If you decide to become a customer, you will get 10% off by coupon code SEDAILY.

What I love most about Monday.com is how fast it is. Many project management tools are hard to use because they take so long to respond, and when you're engaging with project management and communication software, you need it to be fast. You need it to be responsive and you need the UI to be intuitive.

Monday.com has a modern interface that's beautiful to look at. There are lots of ways to use Monday, but it doesn't feel overly opinionated. It's flexible. It can adapt to whatever application you need, dashboards, communication, Kanban boards, issue tracking.

If you're ready to change the way that you work online, give Monday.com a try by going to Monday.com/sedaily and get a free 14-day trial, and you will also get 10% off if you use the discount code SEDAILY.

Monday.com received a Webby award for productivity app of the year, and that's because many teams have used Monday.com to become productive. Companies like WeWork, and Philips and Wix.com. Try out Monday.com today by going to Monday.com/sedaily.

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

[END]