Google Prodcast Season Four Episode 6: The One with Startups and Adam Fletcher

[JAVI BELTRAN, "TELEBOT"]

STEVE MCGHEE: Hi, everyone. Welcome to season four of The Prodcast, Google's podcast about site reliability engineering and production software. I'm your host, Steve McGhee. This season, our theme is Friends and Trends. It's all about what's coming up in the SRE space, from new technology to modernizing processes. And of course, the most important part is the friends we made along the way. So happy listening, and remember, hope is not a strategy.

—

STEVE MCGHEE: Hey, welcome back, everyone, to the Prodcast. This is Google's podcast on SRE and production software. This is a fun one. We have a guest that actually was from way back. Adam, you helped start the Prodcast when it was strictly internal. I think it was you and John, right--

ADAM FLETCHER: That's right, that's right.

STEVE MCGHEE: --in the San Francisco office. And then we also have-- Matt is here with us. Hi, Matt. How's it going?

MATT: Hello.

STEVE MCGHEE: And I'm Steve, I should say that. I have a name also. Adam and I met in the San Francisco office at Google way back in the olden place. But I think you worked at other places before that and worked at different places since then. Is that true? That seems like a truth.

ADAM FLETCHER: That's the rumor and also the truth. Yeah, I was at Google for a long time. I came over via the ITA Software acquisition. I'm wearing my ITA Software shirt, which is confusing.

STEVE MCGHEE: That was the plane one, right?

ADAM: Yeah, and makes the airplanes, yeah.

[LAUGHTER]

Yeah, we made flight search and we made airline reservation systems. I was on the reservation system side. Google bought us largely for our flight search side but powers Google Flights to this day. I was out of the Cambridge office, moved out to San Francisco, where I did meet you, Steve. And this is 10 years, more than that? 12 years ago, 13 years ago?

STEVE MCGHEE: I try not to count years.

ADAM FLETCHER: Sure. It was-- yeah, yeah, back when we were both 21. And then we--

[LAUGHTER]

And then I was at Google for a long time, loved working at it, loved working with folks like Steve and Paul, who's-- you can't see him, but he's here, and all the folks over there, super smart group. Wanted to start my own company, so I left to start and build startups.

STEVE MCGHEE: That's awesome. Adam gave a very good talk that I think is on the internet somewhere, which is about how flight search is actually complicated even though it doesn't feel like it should be. So I highly recommend checking that out if you have the means. We'll put it in the show notes or something like that.

But basically, there used to be a lot of Lisp and curly braces and stuff back in the day. And mainframes are real still to this day, I believe.

ADAM FLETCHER: That's right.

STEVE MCGHEE: My favorite take of yours is that the airline industry doesn't count nines because they have a 100 instead.

ADAM FLETCHER: Yeah, that's right, that's right. They have 100% uptime, no nines.

[LAUGHTER]

STEVE MCGHEE: So it turns out it's possible, but only if you spend a lot, a lot, a lot, a lot of money and time and do a very limited set of things and have a huge amount of, I don't know, complications, redundancy, and plans and things like that. So it's a pretty cool background. But I'm guessing you don't do flight stuff anymore. Is that true, Adam?

ADAM FLETCHER: Yeah, that's right. I left Google to build companies. And the first company I built

was-- this was back before we had LLMs and we just talked about machine learning, which really was just fancy math.

STEVE MCGHEE: Decades ago, I'm sure.

ADAM FLETCHER: Decades, decades, and decades ago, hundreds of years ago. And did some stuff in the mobile space on that, built some real-time stream processing engines that were really neat and sold those to a security firm, did really cool threat intelligence security stuff for a while, which I loved.

Started another company that did called bit.io, which we did serverless Postgres. First to market serverless Postgres, everyone loves serverless things now and everyone loves Postgres. So that was super fun, did that for a bunch of years, bought by Databricks, worked at Databricks, left Databricks.

And now I'm running a company called MarketStreet, where we are helping small business owners have a platform to communicate and collaborate on. So it's really far away from everything I've ever done. It's not infrastructure related.

My joke when I raised money this time was like, look, I shouldn't be allowed to make websites. I should only be allowed to make command line tools. But here I am making websites. So I don't know what happened. The world is topsy turvy and upside down.

MATT: What's new making websites these days? It can't be any different.

STEVE MCGHEE: Yeah, it shouldn't--

ADAM FLETCHER: You just ask-- yeah, you ask an LLM to do it for you because you don't know how to do it. [LAUGHS] And you use-- there's, server-side TypeScript or something like that. There's Next.js and all those things. It's wild, the advances in the website technology.

STEVE MCGHEE: I actually just did this the other day, Adam. I vibe coded my own personal website that I've been meaning to do for a long time. And I just was like, yo, computer, just search the internet for my name and videos of me and make a website for it. And it just did it. And it's just the framework--

ADAM FLETCHER: It's amazing.

STEVE MCGHEE: And I was like, OK, you say so.

ADAM FLETCHER: You're like, I hope it works.

MATT: I did it, too, Steve. It was great.

ADAM FLETCHER: Really?

[INTERPOSING VOICES]

MATT: --website also, yeah.

ADAM FLETCHER: The vibe coding thing is wild. I had to learn to think bigger. I used to give it-- I used to decompose in my head the problem and make an off page. And then it's like-- I'm like, maybe I should just ask it to make the whole thing. And then it just made the whole thing.

And then I had to use every aspect of my years of experience to debug the problems that the LLM made and how to deploy it and how to launch it. It turns out SRE matters at that point because you're just left with a bunch of TypeScript that does-- then people start saying stuff like Docker or containers.

MATT: I think it just hit on the first-- not in the wood here, which is cool. So a lot of great automation gets you something up from nothing. And then you take your wisdom to it. And you look a little closer and you're like, oh, wait, there's a lot-- something wrong here. There's a lot, right here, this is great, but-- and then a whole lot of buts.

So tell us how that's going. Where we are compared to where we were, say, five years ago is miraculous. And yet there's a whole lot of caveats. But you know some stuff, so tell us about this stuff.

ADAM FLETCHER: Yeah, it's funny, I can give a real practical example. So we're building-- the stuff I'm building now involves a lot of data management to make sure our LLMs can say smart intelligent things. And the deployment of that-- what we would've call ETL directed acyclic graph execution stuff. We deployed it on Vercel because we used v0 to help build it. We use Cursor. We used Vercel.

And then, I can't have long running background functions in Vercel. And this is something that I think if you don't know what all those words I just said mean and you don't know what to do, yeah, so I'm like, OK, well I'll switch over-- first, I thought it was my back end database provider.

So I switched those to a different one. And then the different one is like, you can't use async I/O against us. We blow up when you do that because we have 95 proxies in the way, and they are not transactional aware.

So then we moved again and then had to deploy on fly.io instead of Vercel for-- it's like a nightmare. And all those technologies are great. They significantly enhance you.

If you don't know what all that stuff is and have spent a long time and a career building that out, how do you interpret that? Even asking the LLMs, like pasting the async I/O, Postgres error into-- they're like, sorry, I don't know what to do. You get to a point where it just doesn't know. And there is a little specific--

STEVE MCGHEE: Yeah, a lot of these platforms have these constraints in them, which is great. It helps them execute well. But if you don't understand what the constraint is, you don't really know what to ask for or how to-- more importantly, I guess, is how to interpret the “when it doesn't work” situation.

ADAM FLETCHER: Yeah, I think that's a critical skill. And it's this interesting gap of like-- I don't know how people are going to get to that point. The actual infrastructure side is very challenging still unless you have these narrow guardrails that don't scale necessarily or they don't have all the things that you expect and want.

I don't-- but on the flip side of that, I think from a startup perspective, in many ways, you don't care. If it works well enough and you get your minimum viable product out, you worry about those other problems later. And you can hire someone to solve them.

So I have a bit of a curse where I'm like, oh, I know how to solve that problem. I'll just dive in. And then two days go by and I'm like, why did I do that? But you have to, it has to work. And if you're building anything of significance, I think it matters.

STEVE MCGHEE: Well, I know that sometimes SREs overfocus, I guess, on eliminating technical debt. But I believe a lot of-- tell me if I'm wrong, but a lot of startup work is ignoring technical debt. It's just powering through. And is this a problem in your SRE heart of hearts?

ADAM FLETCHER: It was, initially. I'm much better about it now. But it did take a long time to learn. And now I call it swiping the credit card of technical debt. So you know that you're gaining-- you're accruing debt, and it has interest. You want that metaphor real strong in your head to know.

But there's nothing wrong with it. In a sense and in a startup, you're fighting for survival. So in a way, if it doesn't all work, you just declare bankruptcy and move on and all that technical debt goes away. Who cares?

Or another way to say it is users matter more than perfect code and perfect-- in a way, you care more about product market fit than anything else and growth more than anything else. And so technical debt doesn't matter almost to the point where some aspects of security are like, yeah, I'm not doing this perfect. But I'm going to change that.

But you are swiping a credit card of technical debt because, if you become successful, if you find product market fit, you owe on that. And I think that's true in many cases. There are examples-- counter examples of that would be if you're building security software, if you're building or cryptography. You're doing stuff like that. Or bit.io, where we did databases, we have to have asset semantics. If you don't have asset semantics, people notice and they get mad. It is a tremendous thing or tremendously important thing.

So there are special cases where you're effectively selling reliability or security as part of your product. And that, of course, then you have to do it right. But when you're not, yeah, you have to learn to ignore all that stuff in some ways.

STEVE MCGHEE: Yeah, I think just, to belabor the metaphor a little bit, if you're swiping the credit card of technical debt, it helps to understand that a thousand dollars of debt versus a million dollars of debt, what currency are we using? Is a million a lot? I'm not sure.

ADAM FLETCHER: Yeah, no, that's a great point.

STEVE MCGHEE: Is this debt that we're accruing-- is this something that we will be able to-- assuming that we do succeed, is this a thing that's just going to ruin us in post-acquisition or just before acquisition? It might be a little bit worse-- and things like that.

So maybe you have to have your eyes open about this debt in terms of being able to understand it, not just blindly swiping. But you still accept it. You still mark it on the table and move on, I guess.

ADAM FLETCHER: 100%. I think part of that is that risk management and how it tactically applies. The flip side of that same metaphor is that you only have so many innovation tokens to spend. And you really want to be spending them on the things that get you product market fit and serve your users or serve your growth or whatever you're trying to do.

And you don't want to spend it on moving your database and cloud provider 17 times over a weekend. And yet, that's what I did this weekend. So this is like a-- what is the expression of cobbler's? Their kids

don't have shoes or whatever. I can't let go sometimes. That's the problem.

MATT: So a second ago, you said this now market fit.

ADAM FLETCHER: Mm-hmm.

MATT: I don't know, I read an article about what people are using AI for in product ideation, the brainstorming process very early on in the infancy of a startup or even, before you get in that point, so just what do we want to build? Or what is this thing you have a kernel of?

And before I even get a whole bunch of engineers in a room to make slides or even slap a bunch of code together, just start-- just have the AI concept a bunch of websites, make some ideas. And then you have the people with deep pockets look at those things and go, nah, I just-- I can't understand why you want to build this.

Are you seeing now much faster iteration on discarding bad ideas before they get to engineers spent time on? Or are we now seeing things built a little more thoroughly?

STEVE MCGHEE: Is it getting better or worse?

MATT: Yeah, are we failing faster at bad ideas or are we building them out more thoroughly before we--

STEVE MCGHEE: Are we DDoSing the VCs?

MATT: Yeah, yeah, that's a good--

ADAM FLETCHER: That's a great-- yeah. So I think my journey on this is a little different than some folks. There are founders and CEOs and entrepreneurs and stuff that are incredibly passionate about a specific product or problem, I should say. And those people are like-- they're like, my life is designed to fix health care or whatever.

And I love those people. They're going to do their thing no matter what. I'm much more of a analyze the market, come up with a bunch of ideas during the brainstorming phase. Me and the person I built the first two companies with, we have a spreadsheet of 150 ideas.

And we've been just building it for 10 years. And everything is on there. We have stuff-- great ideas, what I call great ideas like what if we had house blimps? What a great idea, right? And then obviously

not-- yeah, ship it, ship it. Or practical ideas, like literally serverless Postgres was on there. And that's what we built.

And then we just keep adding columns to these rows in the sheet that are all the exclusion or inclusion criteria. And why I bring this up now is that, what AI has really changed for us in the ideation phase-- or that idea maze is the cool startup kid way to say it these days-- is like, what is venture fundable?

So for us with AI, you can build so much faster with so many fewer engineers. And you can get to prototype phase and you can test if it works so much faster. That is great because you get to find out if you have product market fit, you get to do all these things in much cheaper and spend less money. But it also means that everyone else in the world has that ability also.

So your moat, the thing that protects your company or your idea, you need to find new moats than you previously had, was like we were just a bunch of smart technologists. And we built hard technology and wrote a lot of code. And that's our moat. It's our intellectual property, et cetera.

But now someone else can be like, yo, vibe code me Instacart. And they do. And the moat is a lot, lot, lot different these days. And so you have to find new aspects. With what I'm building now, the aspect is community and networking between people. I think those connections are persistent.

I think GitHub has that. I think you look at long-term valuable-- I think long-term valuable things have that side of community or they're like 10x technologically better. And the LLM is not going to build something 10x technologically better. And it can't do the community side of it.

But it can do so many other consumer apps or SaaS. Infrastructure SaaS is a great thing that-- the LLMs can do that. And they will build that. And it'll be fast. And so I think it changes the dynamic.

I think you do get a lot of people spamming. I added AI to X to the VCs. But I think the smarter VCs are saying, how is this going to possibly 100X my investment when anyone else in the world can just ask the LLM to build the same thing?

And so you need to have a different lens I think when you're choosing your ideas. And it has to be that lens of-- well, one, it has to have a lens of, is this just a Jira ticket at OpenAI or in Gemini? Is this just going to happen in two years?

Is this some product manager just going to do it? And then also, what makes this AI-proof or what builds a moat that defends it against other people vibe coding it as well?

MATT: So taking us to the general theme of our podcast here, how does it mature? How does it-- you unlock the achievements of requires reliability engineering now? We go from thing with slide deck to thing with random website with some-- we bought some time on the cloud too. OK, we have some employment.

We have people building stuff. And now there's stuff that people use and they expect them to stay running. When do you decide it's time that we don't just have stuff that's running, we have actually people accountable for running, and we have expectations?

STEVE MCGHEE: We have the growth now.

ADAM FLETCHER: Yeah.

MATT: Tell us about your long arc of experiencing this here and in your own industries.

ADAM FLETCHER: Yeah, and bit.io is a good example of this, where bit.io, serverless Postgres, has to work, but it also has this unique thing, where with serverless Postgres, you-- the goal is, in a way, is to shut the machine down. Shut the compute down when people aren't using it.

But it's a database. So how do you do that? How do you do that safely? How do you do that without corrupting data? How do you do that with transactions in flight? What do you do, time them out? You have to start thinking about those things.

But when do you start thinking about them? You start thinking about them when you have what I consider as a successful failure, if you will. You have some user using you and they're like, yo, this thing broke. And they're mad about it because it means something to them.

But hopefully it's not-- it's not something you can necessarily anticipate or see because you should deal with those if you can. But you want to have enough load-- you effectively want the Twitter Fail Whale. Twitter did an amazing thing of turning the Fail Whale into marketing-- genius. I don't know how they got that done, but genius.

And then you say, we have to spend time on this because it matters or we're losing customers or people are noticing or whatever. And I think that-- that isn't to say-- I think if you can, you should plan it well. You should take care of building your software to plan it well. You should know where your problems are going to be, like the back to the credit card thing of technical debt.

But I always think about it as, when my users notice or complain or if it's some fundamental problem in the entire product I'm offering, is reliability-- like we said earlier, is reliability and security part of my product? If it is, then, yeah, we have to be-- it's got to be built in.

But man, you got to wait till the user tells you something's wrong because if you don't, you'll build the wrong thing. You'll invest all this time in fixing a problem that doesn't exist.

STEVE MCGHEE: So if we can follow the model that you're describing here, do-- let's say we have a couple startup founders who don't have reliability background, don't have security background. And they're not building a reliability or security product. They're building a consumer product or whatever. They're building Instagram 2 or whatever.

At what point would-- what's the switch where you think a founder has to be like, uh, we need an infrastructure person-- or maybe not infrastructure, but a reliability person? At what point--

ADAM FLETCHER: Reliability or security person-- yeah. And I think that when you start seeing the growth--

STEVE MCGHEE: A specialist, right?

ADAM FLETCHER: Yeah, I think when you start seeing the growth, when you say-- when you can project your growth curve out of users or whatever, the consumer app space, where you're saying, if this keeps going this way, we are going to run into these problems. You should make those projections and say-- and it can be hard to know.

Like you said in this case, if you have a founder who's not a SRE or a security person or just doesn't have exposure to, what can fail and when it goes wrong? I think that they still need to-- you should still say to yourself, we're growing so fast. How does this impact what I've built? Can the choices I made-- which in a modern startup, you're composing it of SaaS services.

A modern startup is, like I said earlier, you use Vercel and Fly and Supabase and Neon, whatever. You compose all these other SaaS, which is not traditionally how you do it. Usually, it'd all be in one cloud provider. And you may need to go in that direction.

You may need to go-- I think eventually, one of the reasons the major cloud providers are so successful is that they offer such a coherent and built-in policy around-- they offer you this path to scale and this path to security with the downside of the complexity that comes.

But I think it is-- I think you just always have to be looking at your growth and say-- question your underlying building blocks, do they work? Are they big enough? Are they strong enough? It's hard.

STEVE MCGHEE: Another thing that you mentioned was-- and maybe I'm projecting here a little bit. But it sounds like the story you're getting at is also-- the failures that you're going to encounter when it comes to scaling up, you can't actually see them coming.

Some of them, maybe you're like, well, last time I did this, it was when we got to a million concurrent sessions per whatever. But now you're on Fly or whatever or you're on service.foo that has never existed before, and so maybe their ceiling is much higher or much lower.

And so there is this community out there called Learning from Incidents. I don't know if you've heard about this one. But it's basically-- when I first heard about it, it totally made sense to me, even though I hadn't heard their actual background hero story or anything.

But it was like, the way that you scale things over time is that you fall occasionally. And you learn when you fall. And you're like, aha, that's where the line-- that's where the bump is. We found the bump.

And the trick is to be able to fall gracefully, to extend the metaphor a little too much, to only fall in parts of the globe at once potentially, if that's possible, like in some of your user base. So we get into the continuous deployments but across failure domains in a gradual rollout kind of way, blah, blah, blah.

So I guess what I'm getting at is, in that modern infrastructure-y world that we're talking about, how many of these principles transfer effectively? You and I worked on onboarding together with GSLB and all the things inside of Google. Do these words-- not those words exactly. But do these concepts translate into this today's operational world? And do people get it? Is there a Rosetta Stone that needs to be built for this world?

ADAM FLETCHER: Yeah, I think there is. There's definitely a Rosetta Stone that needs to be built from Google out, yeah. If you're a Googler and you want-- I'm not encouraging any Googlers to quit. If you want to build a startup, I think you will need a Rosetta Stone. The way we build software at Google is very different.

However, I think you have things like the SRE book. You have just the simple thing of ask the LLM, ask Gemini like, hey, I have this. I am deploying on Vercel with Next.js. My user rate is growing. Where am I going to run into issues?

You can just literally ask that question and it will give you good answers because a lot of these best practices are well codified that the LLMs know. And they'll be good enough to get you to the next step. And then you can literally be like-- it'll start saying things like, often, scaling comes, well, check your database. Are you on some free tier of some sketchy database? Maybe don't do that. Do you have indexes?

And the big gap to me would be we still-- I talk to a lot of founders, technical founders even. That idea, create indexes in your database, that idea has been around for 50 years, literally forever. And that is still mind-blowing science to a lot of people. They don't think about those problems. They just assume that it does it all right for you.

And so I think the luxury we had at Google or you guys have at Google is that a lot of those things-- one, there's very smart people like you guys to answer those questions when they come up. But also, there's great frameworks to handle it. And we've thought a lot about it.

But we thought about it in the context of the infrastructure Google has internally built, which is very different from what the rest of the world is like. You don't get those things. You don't have GSLB. Even down to the disk infrastructure of D and all the way up, the big table, you don't have those things. Colossus, you don't have any of that. And you have different assumptions on how things are built.

MATT: This really brings to mind something. Two things come into my head. One is, first of all, you've done this sort of meta hero's journey a few times, which is kind of fascinating to me, which is going around the clock with multiple times of, OK, first, you have nothing.

Now you have toy website. Now you have toy website with superpower X. And then let's put thing on cloud. Oh, never mind, scrap that. Let's do it again. OK, now, you have a magic flute and now you're going to go into Castle X.

So my question is, the actual question is-- so you take some off-the-shelf, perhaps open source tools, then migrate to possibly paid sources made by third party. Then you go, mm, actually, that thing can't do a thing I need because it won't scale. What is that like to migrate from paid tool sort of does what you want, now you're going to do it yourself?

ADAM FLETCHER: Oh, your own-- oh, yeah.

MATT: What's that you're doing it yourself part like? It's big. Because you're going to do it, it's got to

scale up for the thing you're doing. And now you're writing yourself and you have to support it. And my guess that's where the, quote, "fun" happens. And scaling and the reliability part becomes scary and opportunity. That, I find really fascinating.

ADAM FLETCHER: Yeah, and I think you're right. I think there's these inflection points. Now, for me, personally, and for the last co-founder and my current co-founder, we come from this background. So I'm excited. I'm like, great, I get to do cool shit. And then everyone who's ever worked for me, reminds me, is like, no, you don't. You have to go talk to an investor. You're not allowed to build any of those things.

The migrations are scary first, especially things like database migrations. Or fundamentally, the stateless web migrations can be fairly straightforward. But migrations between database providers, authentication providers, things like that, migrations from alternative cloud to an honest cloud platform like AWS or GCP or Azure or whatever, those are hard.

And they're hard for a lot of interesting and not fun reasons. If I never have to deal with IAM and policies again in my life, I would be-- it would be happy. I'd be in my happy place. But you have to deal with them. And they're there for a reason.

Try explaining, oh, well, you need a private link between these two VCPs. By the way, you want to be multicloud, so those terms don't apply in other clouds. How do you set up a redundant VPN link between two clouds? Yeah, you hire a specialist. This is an area where you-- that's very, very hard.

The reason these alternative clouds exist for people like me or people who are building startups is because they want it to be easy. Bit.io, the whole premise of bit,io was it sucks to run databases, don't ever think about it again. And we will tune it. We will do the right thing. We will scale it up and scale it down for you. We'll do all of that for you.

And that was successful. People loved it. They still loved it. Neon and Supabase exist because of that. But yet eventually, yeah, you got to move sometimes. Sometimes you have to move because someone buys your company. People of bit.io, I apologize when we got bought and we had to move databases. That sucks. Sorry, I hope we helped you. [LAUGHS]

Yeah, those transition points are hard. They're very, very hard. And they're still hard. And no one has made that easy.

STEVE MCGHEE: Doing the classic podcast thing, we now require you to predict into the future. So projecting from what we've been talking about for the next one to two years, what else do you see that will become weirder or less of a pain in the butt or something like-- how does this arrow point forward?

ADAM FLETCHER: It's going to be significantly easier to create software. And that, actually, I love. I've spent a career trying to build tools that make it easy for people to build software, developer tools and things like that.

One of my big things is software is the most amazing thing that we've come up with. I think of a problem and I solve the problem. Amazing.

And I think it's just going to be easier and easier to do that in two years. You're going to have hyperspecialized, hyper-quick builds. You're going to have a lot more internal software built to solve internal problems that will be very customized.

I think you're going to have a lot less need for software engineering in the traditional sense. You'll have more product people build software using the vibe coding stuff and all that. And that'll all be net good initially.

MATT: What about some sort of mutual, some happy coexistence between us and the robots?

ADAM FLETCHER: Yeah, I definitely think that's possible. The whole premise of the company I'm currently building, MarketStreet, is that the human connection is the important part. Small business owner to small business owner, those people will still exist.

They will have-- they're doing real problems, talking to real people. They're working in those spaces. And I think that AI can complement a lot of what they do and help them grow their businesses without being extractive against them and can help bring to them-- help build human connection and help build human outreach. That's my goal.

And what I'm doing now is, and the driver of that is focus on the humans, not the computers, and focus on the human connection and focus on the interplay in neighborhoods and between people. And I think that's actually where we need to be spending. Obviously, I'm betting on it. But I think that's actually where we need to be spending time.

There, of course, needs to be building the technology that gets us there. But I think that people don't

go away. We have to work with each other. And I think the human connection is the non-replaceable thing for the AI. And we need to focus on that.

STEVE MCGHEE: Yeah, on an optimistic take, one of the lines that we use in the resilience community is like, it's a sociotechnical machine. So it's not just the machine driving, but there's people in the loop. We can always-- I don't want to say hope on, count on--

ADAM FLETCHER: Hope is not a strategy.

STEVE MCGHEE: It's not a thing that we're supposed to--

ADAM FLETCHER: It's not a strategy.

STEVE MCGHEE: But there are people to at least assist in making decisions and steer things and observe problems and move in the right direction.

ADAM FLETCHER: I want us to go there. I want us to be more creative via artificial intelligence. I really want this to be the time. Like we always say, when technology comes and we bring automation in, that frees people up to have their time and their needs taken care of. And I really want that to be true.

STEVE MCGHEE: So far, sometimes it works. But a little more tangible, I want to bring us back to another point. So specifically, if we skip the big philosophical angle and strictly focus on the SaaSs of the world and the PaaSs and the whatevers of the world, do you have-- the reason why I'm picking on you is because you're an SRE who is further, further towards the edge of this technology than most people that we talk to.

So I'm curious what you think are the problems with the current offerings that you think might be solutionified and turned into platformy things next. Or what's your wish list really, in terms of how to progress this stuff so that the next round, Adam 3.0 will come around behind you and have a better time solving problems like this?

ADAM FLETCHER: What I really want-- and people have heard this rant before. But what I really want is the software as a service, the platform as a service, that is the SRE model from Google to exist in the world, where-- at Google--

STEVE MCGHEE: Aw, shucks.

ADAM FLETCHER: Yeah, I know. Well, we have ownership. I think that every time-- we had a product that we onboarded into SRE, it was a paired partnership, ownership with product and with everybody involved, where SRE had the full and reason and expectation of modifying the code, building the code, making it-- solving the aspect of reliability often to security and things like that would come in and we would just, hey, we changed your code.

Literally, I think one of the first projects I worked on at Google when I stopped doing airline stuff was Google News. We moved, we ported Google News from the old framework. It was on to a framework, an internal framework Google has, called Apps Framework, which maybe still exists, I don't know.

But it offered underlying it-- it was like an SRE-led initiative that underlying it with all of these things like load shedding and all these different reliability things that made it just much, much easier to operate at scale and much more reliable. And we did the port of from the bespoke--

STEVE MCGHEE: We, the SRE team.

ADAM FLETCHER: We, the SRE team, did the port over to Apps Framework. And it made Google News a much more stable and reliable platform and helped to grow-- or helped stay alive and grow from then on. And I wish we would see that in the world today.

We get all these people building SRE, quote, "SRE" SaaS products that are not that. They're very reactive. They're very about monitoring and they're very about all these other things. And they're all reactive.

They're all like, oh, we reduced your MTTR. I'm like, well, why don't you just change the code? I want to check in my code and I want it to be like, yo, I added all this stuff. And by the way, all of it's awesome now. All the reliability stuff is awesome now.

You weren't doing multiregion, I got to configure it if you wanted. Here's your code changed. You weren't doing load shedding, I got it, I turned it on. That's what I want from my next SRE SaaS. I don't need more monitoring, and I don't need alerting. And I don't need incident management or SLOs or all that. I have all that.

STEVE MCGHEE: Isn't that requisite at the code layer? It's not at the platform infrastructure layer.

ADAM FLETCHER: 100% in the code. It's in the code.

STEVE MCGHEE: This isn't a platform hot swap, Indiana Jones situation. This is a you now need to build the thing this way. You're going to use this framework. Yeah, there's a migration event. But that has-- there's a lot of-- I work with customers a lot. Migrations don't happen honestly very often.

ADAM FLETCHER: You don't want them to. They're costly.

STEVE MCGHEE: They're costly. And they feel to be zero value often.

ADAM FLETCHER: Or negative value, potentially.

MATT: Yeah, you're now introducing a maintenance burden to your development team, that's going to be, well, why am I doing this? You're like, well, your uptime is going to go up. And you're like, and I now don't understand this new platform I'm on--

STEVE MCGHEE: Right.

MATT: --yet. I look forward to understanding it in the future. But right now, it's just, the cost of understanding this is very high. And your reliability team is like, look, it's going to be great. It's like, but I haven't been convinced and I have a bunch of new features I want out next quarter.

And that sounds to me to be a discussion that we keep having. And if you are a big, big company, you can afford that cost. And your developer team will give you a lot of rope to work with. But if you're really small, that is a really difficult decision to make for your team, especially if you're working with limited budget.

And I agree with you. If these things were in place all around you, you don't need to figure this out yourself. We've already built you a resilient framework. Just use this one. It has load shedding in it and monitoring tools in it and alerting in it-- just why would you do the other thing yourself? That would be madness.

ADAM FLETCHER: And I want the tools, the platform, to effectively do the migration with zero downtime for me. I want it to be like, here's the Git commit, by the way. We can run it and we'll split. You'll have an A/B test. We'll do it. And

It just does-- I don't have to think about it. A very practical one, like Django, it's an ORM for Python and a website building tool for Python. It's very, very reliable. It's been around forever. There's migrations in it.

But there's also you have to run it inside another framework from Gunicorn-- or Uvicorn is the new one, right? And that's the async version. What are the benefits of that? How do we explain that? How do we automatically port you over to it if you're in a legacy Django code base?

It should just do that. It should know. It should be like, and I've installed these extensions and they're configured and they're running, and we tested it here. And that's what I think. If someone came to Google SRE, they would own and do that. And of course, it's a little easier because the SRE org is convinced of the value of that.

And I think performance-- for what it's worth, I think performance is the magic answer to a lot of these questions. And it was at Google, too. When we would pitch things, people understood the value of a millisecond. I used to go visit the GFE team, the Google front end team, in Cambridge when I was there. And I'd go downstairs and they're writing code.

And they had this big diagram on the whiteboard that was how many nanoseconds each branch of the GFE took because they cared. That mattered a significant amount. You're this piece of infrastructure that can't take-- your budget is not milliseconds. It's nanoseconds. But performance results in delight. It results in the user.

And it results in-- I think you can actually track the value of performance. And so one thing you can say is, oh, we're doing this migration not just for reliability, but also, look at this cool thing, this performance thing, which actually increases your bottom line. It's a revenue-generator. It's not a cost center. And we give you clarity as to why it matters. I think that's one of the hacks you have as--

STEVE MCGHEE: There's a similar story from years before, around News where-- but this was in the-- if you remember, there was a downloader service for apps that Google provided for Windows and Mac and stuff like that?

ADAM FLETCHER: Oh yeah, yeah, yeah.

STEVE MCGHEE: It was like the blob serving. But it was one specifically for applications, so they had versions and package manifests and blah, blah, blah, blah. It was homegrown for years and years and years. And one very, very cool software engineer, who doesn't describe himself as an SRE, went in and was like, I have replaced you with a small Go script because I like Go a lot.

The reason why it worked was performance went through the roof. Simplification of the code, it

dropped dramatically. And they were able to shed all of the features that they didn't really use anymore. And no one would have done that on their own. But he was just a senior engineer, who was looking for something cool to do.

I feel like that value, if we could package that concept and bring it out to the world and be like, this is actually good, a win executed well-- you don't want someone spending 16 years trying to do that with a team of 55 people who constantly cycle and waste a bunch of money.

If you could atomically do that thing, you can get a huge amount of value out of it and not just in terms of money and performance, but reliability and probably security and all sorts of other things like maintainability and all these things. So that's really hard to get across to everybody, I think.

ADAM FLETCHER: I think you're right. I think it is very, very hard to get across. I harp on performance as this-- one of the reasons is because you can always show that cool graph, man, where it's like, this took 200 milliseconds, now it takes 10. Those graphs are very convincing.

And this is true for beyond the startup world. This is true in the small business, medium-sized business, big business world. Man, those graphs sell. You've really changed a lot. And you can point to the direct monetary impact of this-- less compute, less memory-- or maybe not less memory, maybe you're doing the trade off.

But still, you get the point, too. And I think that that's a big, big thing. And if you include in that reliability, which often you do-- and I think you made a good point about code simplification and architectural simplification. You got rid of all the things. You got rid of all the assumptions. You understood the abstractions better. A good refactor does that.

And I think that's one of the things that I'd love to see the SaaS tools start doing that. I think we're close. I think we're a year out from, really, the LLMs being able to do quite a bit of that with some, at first, human hand-holding.

STEVE MCGHEE: The ability for an LLM to go in and actually do open heart surgery on your code is new. It didn't exist. This didn't exist five years ago. No way.

ADAM FLETCHER: It wasn't here two years ago.

STEVE MCGHEE: Good point.

ADAM FLETCHER: The first time I tried doing something two years ago with Copilot, I was like, wow, you can't even do math. And now I'm like, build my whole site. And it's like, I got you.

STEVE MCGHEE: Start over. Do it again.

ADAM FLETCHER: No, exactly. It's always like, oh, you're right.

STEVE MCGHEE: The other day, I was vibe coding and it did something weird. I literally wrote into the chat window, not like that. And it did the right thing.

ADAM FLETCHER: It did the right thing.

STEVE MCGHEE: I was like, wow.

ADAM FLETCHER: Yeah, I always feel like it's reading the first or second design doc a junior engineer has built, has written. And you come in and you're like, I see where you're coming from, great assumption.

STEVE MCGHEE: Cool plan, bro.

ADAM FLETCHER: Yeah, trust me on this one. You really want to do it this way, which is the actual best practice. And so then they're like, oh, cool. And they go do some research. And they find out--

MATT: Yeah, yeah, you could do it like an expert. And they're like, oh, OK.

ADAM FLETCHER: Yeah, OK. Oh, yeah, you're right, you're right.

MATT: No, no. Do it like a world-class expert.

ADAM FLETCHER: And he's like, oh, good idea. I should do it-- it's like, so sorry that it didn't before. I'm like, just do it before.

MATT: Well, yeah, I was trained on the internet, not the expert in it, sorry.

ADAM FLETCHER: Yeah, exactly.

MATT: All right. Well, we're getting close to the end of our time here, so I'd like to ask you a question, for a number-- a number in your travels with a unit, preferably a surprising number, like a really big number or really tiny number or a fast number or a long number, something that really is exceptional--

STEVE MCGHEE: A number with letters in it, anything.

MATT: Yeah, yeah, something-- yeah, something that's like, wow, that was impressive to you, that surprised you in your studies in the world. And you're like, wow, I found out this thing when I was looking at a thing. I was like, I had no idea this is a really big thing or this is what people need or want or whatever it is that you would like our listeners to hear about, that's something that really impressed you.

ADAM FLETCHER: Oh, boy. OK, I have a sort of funny anecdotal story about numbers and relates to my work at ITA, is that if-- you know the confirmation code you get on your ticket?

MATT: They had letters and numbers--

ADAM FLETCHER: They're letters and numbers, right? The reason those are letters and numbers is because, yeah-- [CHUCKLES] is that-- this is not a confirmation code. This is the flight-- our first flight we took on our reservation system.

STEVE MCGHEE: Close to it, yeah.

ADAM FLETCHER: Close to it. Anyway, confirmation code, those are six digits, six characters. The reason those are that is because when they first started storing your tickets and record locators on disk, that was the block address on disk of your--

STEVE MCGHEE: Physically speaking.

ADAM FLETCHER: It's physical pointer.

MATT: In cylinder somewhere.

ADAM FLETCHER: So it's literally-- yeah, literally a physical pointer. Imagine you printf the pointer and you got the address. And you're just like, here, this is yours. This is how you find it.

STEVE MCGHEE: Are those locator--

ADAM FLETCHER: They're not in hex. They're--

STEVE MCGHEE: I've seen Zs and stuff in there.

ADAM FLETCHER: They're not in hex. They're literally IBM 360 disk addresses.

STEVE MCGHEE: They're some sort of 36-character--

ADAM FLETCHER: Yeah, some sort of 36 character by 6. Yeah, factorial 6 or whatever it is, I am bad at math. And that number has-- that's always really impressed me, that the leakage-- just talking about leaking your abstractions. But it worked great. You can find it fast. Hash tables are the most profitable data structure.

MATT: Railroad-grade standard of flight locators, this is fantastic.

STEVE MCGHEE: Of one across a very large key space, let's do it.

ADAM FLETCHER: Yeah, let's do it. But now you have to support it. So all the new stuff that doesn't do that, it's like, what the [MUTED]? How are we supporting this? [LAUGHS]

STEVE MCGHEE: That's awesome.

ADAM FLETCHER: Yeah, yeah, it's pretty funny. The other fact that I find interesting about small business, related to small business and things like that, one, is every time I've talked to a small business owner and asked, do you use AI, they have laughed at me. Two, the median age--

STEVE MCGHEE: Which way do they laugh at you, the “of course I do” or “of course I don't”?

ADAM FLETCHER: “Of course I do not.” So the people that you-- when you walk into a retail store, think about the technological differences, the age gap, the exposure, the cultural context of all those people is so different than what we are--

STEVE MCGHEE: Let's remember what medians mean, above and below that number. So yeah, that's pretty awesome.

ADAM FLETCHER: Exactly. Yeah, a lot of it are-- obviously, 50% above, but yeah.

STEVE MCGHEE: Cool. Well, thank you very much, Adam. This has been a great talk, unsurprisingly. Is there anywhere on the internet that you would like people to hear your words in other ways or otherwise hear you make jokes or talk about runs you go on or whatever sort of socials--

ADAM FLETCHER: Yeah, I'm on Twitter, @WriteAheadLog, which, if you're a database person, you will find that somewhat funny or perhaps not, if you've ever used or had to repair one, then [LAUGHS] you might have PTSD. My company is MarketStreet, MarketST.com right now. There's a funny story about

the actual domain. I'll tell you later. And then, yeah, check us out.

And look, email me. And if you're some sort of estuary that's starting a small business, I want to hear from you. It's like a strange-- [CHUCKLES] but yeah. And computers are the best.

STEVE MCGHEE: Cool. Thanks, Adam.

MATT: Thank you.

STEVE MCGHEE: And don't forget, everyone, that hope is not a strategy. And we'll see you next time.

ADAM FLETCHER: Hope is not a strategy. All right. See you. Bye.

—

[JAVI BELTRAN, "TELEBOT"]

JORDAN GREENBERG: You've been listening to Prodcast, Google's podcast on site reliability engineering. Visit us on the web at SRE dot Google, where you can find papers, workshops, videos, and more about SRE.

This season's host is Steve McGhee, with contributions from Jordan Greenberg and Florian Rathgeber. The podcast is produced by Paul Guglielmino, Sunny Hsiao, and Salim Virji. The Prodcast theme is Telebot, by Javi Beltran. Special thanks to MP English and Jenn Petoff.