EPISODE 1382

[INTRODUCTION]

[00:00:00] KP: With a few impressive exceptions, software is rarely written by one person. It takes a team and is that team outgrows a single shared office coordination and communication become emergent problems. There are lots of lessons to be learned from companies who have already found approaches that scale.

In this episode, I interviewed Tramale Turner, Head of Engineering, Traffic and Seattle Site Lead at Stripe. We discuss infrastructure, organizational structure, and some insights made at stripe. It's an in-depth conversation with useful advice for all stages of the journey of a modern software engineer.

[INTERVIEW]

[00:00:38] KP: Tramale, welcome to Software Engineering Daily.

[00:00:41] TT: Thanks for having me. Super excited to be here.

[00:00:44] KP: Well, to kick things off, can you tell me a little bit about where you got started in your journey as a software engineer?

[00:00:50] TT: Sure, I think, it's interesting story, or maybe not so interesting, depending on your perspective, but I had that sort of privileged background of getting a computer when I was 9 or 10, started off with a Commodore 64. And then my dad brought home an Apple IIe and then a PC and lo and behold, I had this plethora of computing devices around me and my dad also owned an arcade when I was growing up. And so, I had sort of the Ricky Schroder, silver spoons, arcade game thing going in my bedroom as well, even though we weren't necessarily wealthy, per se, just maybe solidly middle class.

In any case, I was surrounded by all of these devices and I would always look on where were these things made? Where were they manufacture? I was really, really curious to open them up and just learn more about whether these wonderful devices were sourced. I would invariably see Japan, made in Japan, made in Japan, every once in a while, maybe made in China, very rarely made in the USA. So, that sort of planted the seed, and as I graduated high school, I had this opportunity to select – I enrolled into the University of Pennsylvania and all of the computer science engineering students at Penn, I think different than the other engineering students, had to choose a foreign language to study.

So, I had again, the seed planted in my head that all of this cool tech that I was into, was sourced in Japan, and I was 16. I didn't know any better and I love Nintendo and all these other things. So, I decided to study Japanese, enrolled in Penn as a CSE student, and more or less, a Japanese minor even though I didn't officially declare a minor at that point. I studied the language, studied the culture, started even taking graduate classes in the language. And then I dropped out of Penn when I was 19. I moved to Tokyo to work as a software engineer.

Did that at a little startup for about a year, and then came back to the States lived in San Francisco for a bit, commuted between San Francisco and Tokyo, then moved back to my home state of Michigan and started a company, eventually sold that company and went to a couple of failed startups thereafter in the digital marketing space, what we now call the digital marketing space. Went from those to eventually landing in the Volkswagen Group of all places. Volkswagen as in the folks that make the cars, Audi, Bugatti, Lamborghini, Bentley, et cetera. And I joined as a marketing professional, believe it or not in the Volkswagen brand. I was identified as the sort of technical person within the brand to make sure that the brand was making digital marketing decisions that were founded in solid technical foundations.

So, the team built this thing called the New Beetle online buying program, which was the first time in the United States that automotive OEM actually sold cars online. And technically, that's not how it actually worked. There was this sort of marshalling of the dealer inventory onto vw.com. But as far as a customer knew, it was being able to buy directly from the OEM. So, that was a pretty big deal. And a lot of folks got a lot of good credit and some promotions, all kinds of great things. But I spent about 10 years at Volkswagen, eventually moving from the marketing department and the Volkswagen brand to core engineering and supporting all of the brands as a portfolio manager for the Americas traveling all around the United States, Canada, and eventually relocating my office to Puebla, Mexico.

I lived in Puebla for about a year, traveled again frequently to Germany. For some reason South Africa was also in my portfolio because the South African technical team didn't want to align itself with Germany for reasons we could probably talk about in a different podcast, and went from Volkswagen to Nissan, and again joined Nissan and marketing and a captive agency as an account director working on the Infinity brand. I spent a few months, actually only six months there, before I got a call from this tiny little video game company. Again, these references earlier in your life coming back later in your life to haunt you or maybe it's just providence, or maybe it's just coincidence, who knows? But the call came from this company called Nintendo, which I don't know, if a lot of your listeners have ever heard of.

I couldn't resist at least coming out and having deeper conversations, had never been to the Pacific Northwest, never been to Portland, or Seattle or Vancouver, anywhere, in this general area. I came and loved the place. Fell in love with like the scenery and just sort of everything about it. And of course, it was Nintendo. So, I was like, “Sure, I'm going to do this.” Joined Nintendo launched the 3Ds, the Wii U, and this thing called the Switch, which again, I'm not sure if folks have heard of, worked on Account Services, the E shop and developer productivity. If you use a Switch, and you have a Nintendo Network account ID or NNID, using the Nintendo Network account services, my team was responsible for building that so you can blame me. Probably a lot of folks are happy that they don't exclusively have to use their friend codes in order to make online connections.

In any case, spent eight years about at Nintendo, then went to our F5 Networks for about 13 months, before someone pinged me on LinkedIn and said, “Hey, you should come have lunch.” And it was from this. I think, at that time, still in within tech, knowing really well, but not known very well outside of tech, this little company called Stripe. And I went to lunch on a taco Tuesday, and I never left and I've been at Stripe for the last three years doing a plethora of things. But we can probably talk about that a little later.

[00:06:44] KP: For sure. What was the initial grab that made Stripe interesting at that time?

[00:06:50] TT: So, I love telling the story, because I think there's no such thing as like a bad reason for joining a company. It's the reason that makes most sense to you. But I had a fairly decent career, and I started university when I was 16. So, I'm still relatively young, at least, I like to pretend that. I looked at this company and I'm like, “It's a payment services provider. What's so exciting about Stripe?” And so, I sent a mail out to a friend of mine that had joined a few months prior, this gentleman named Niels Provos. And Niels had spent a number of years at Google and was responsible for a number of core security primitives that your listeners are likely aware of. But Niels and I have been friends for like 25 years.

So, I sent Niels a note, and I was like, “Why did you join Stripe? You left Google, you had this really great position, really great surface area, billions and billions of people touched by the things that you do.” And Niels gave me three reasons. He said, one, he wanted to be at a company where he believed in the mission. And it like made sense to him what the company was doing and why they were doing it. Two, he wanted to be somewhere where he felt like he could have the agency to do the right thing, which was an interesting point, I thought, because I felt like at least externally observing what his time was at Google that he had, but it turned out in subsequent conversations that that wasn't a certainly case. And then third, he wanted to be around people that he just really enjoyed working with.

So, when I went in and had my conversations about what was possible, at Stripe, I started to sort of index on those principles and index on those opportunities to, be mission focused, and to work with really incredible people, and to do like really interesting work, like that mission and purpose thing that we all crave. And it turned out that even from the very first conversations, I could just see that emanating from everyone that I spoke to at Stripe. Everyone was extremely happy, extremely positive, super kind, but still doing quite rigorous work and really, like adamant about the importance of the things that they're working on. And then I found that it wasn't quite just a payment services provider, that there are a lot of things that the industry by and large, I think has learned more about what Stripe stands, what the so-called global payments and Treasury network or GPTN. And there's so much more coming that I unfortunately can't talk about in this conversation, but I can't wait for the world to see.

[00:09:25] KP: Very cool. I know when some people start a new position, sometimes it's a company that's large and successful like Stripe that just says, “Hire smart people and we'll figure it out.” Other times you start and you need to start it yesterday and there's a mandate for you. What was your onboarding experience?

[00:09:44] TT: My experience was interesting in that I joined an infrastructure team that had already had sort of a – well, at least it started to build a great deal of trust and credibility within the organization for executing with excellence and having sort of a generally solid perspective on operational integrity. But the team was still somewhat new, nascent in its structure and its identity. So, a lot of the time when we're talking about what a team's identity is, in these Gale technical organizations, we talk about what the charter is like. What is the purpose of your team? How does it impact the business? What is it going to do to continue moving the business forward, whatever needles, or whatever metrics it might assign itself to?

As I found out more about this team that I was joining, the metrics were fine. But were they things that were super motivating? Were they things that really, if you were not part of the team, and you were objectively on the outside looking at those metrics, would it be easy to reason like what the team's purpose was and why it existed? And arguably, as an engineer, you probably won't be able to, but if you're a marketing person, or a salesperson or some other function that's trying to understand, like, what does this team do for me? I didn't know that the articulation was as crisp as I would have liked it.

So, my onboarding was really learning about that, and Stripe gives you the space. In fact, you're actively encouraged to listen first and not be so eager, especially if you're joining us, as a leader to make any changes or try and drive any specific impact, but just learn and absorb and try to reason about what the opportunities are for your teams. So, I was super appreciative of the fact that I was given a space and it's kind of an industry meme, that Stripe likes to write a lot, which means there are a ton of things to read. It's really easy to get sort of a longitudinal history of what decisions were made, why they were made.

I think, when I joined Stripe internally was probably a bit more transparent than it is now. Because at the scale that it's operating at there, there are a few more controls that it has to pay attention to. But there was a time where internally, you could read almost anything, like any source code, any documentation, any sort of – we call it falafelling internally, but it basically just means having a conversation about a topic. Any other notes from any of those conversations were available to you. And so, if you're a person that likes to nerd out and just read interesting things and learn from the perspectives of others, it's a playground. I found myself just being overly thrilled at the opportunity to learn about the – at that point, about eight-year or nine-year history of the organization, and for me, at least. getting up to speed quickly was enabled by that transparency. So, I was quite thrilled to and – sort of the summary is that it was a relatively empowering startup for me.

[00:13:07] KP: Well, I definitely see a lot of advantages in having that transparency in history to go back and learn things, maybe even from employees that aren't there anymore, or to learn about initiatives with institutional knowledge and that sort of thing. I know, you've probably been in roles where that wasn't around. Can you compare and contrast what the experience was like having that sort of access?

[00:13:29] TT: Yeah. So, I won't name any companies. Because one thing, this industry is super small. And so, a little bit of advice to anyone listening, don't burn any bridges. You never know when you might be working with somebody or for somebody, or they may be working for you. But I will say the contrast with an organization that already has its controls in place, and probably has a bit of these so-called barriers, or what I like to refer to as high friction environments, not even intentionally, it's just the nature of an organization sort of growing either deliberately or organically over time. And there are many functions that might creep up in many pillars that might creep up and there are dividing lines. And some organizations even codify those dividing lines and titles or organizational structure, and every software engineer who's been doing it for a little while, and even folks who are maybe just early in their careers coming into the industry, have likely heard of Conway's Law, where you tend to ship your organizational structure.

I think in those more rigid organizations, it's sometimes an unintentional consequence that the structures and the barriers that are in place, make it very difficult to reason about how you have sort of that horizontal ability to drive impact within an organization. How do I talk to a as an infrastructure engineer, platform engineer, and a product engineer, and maybe some other supporting functions and maybe business functions and bring them all together in order to take a really good idea and drive it through the execution saliently, without having to go through a bunch of different meetings and a bunch of different escalations? I think in those really tight, rigid organizations where people live and die by their titles, you see them moving a bit more slowly. Arguably, much more slowly.

Now, that's not to say that Stripe doesn't have rigidity, and doesn't have rigor, it has all of those things. But for example, we don't have titles. So, another thing that I love talking about is David Singleton, who's our CTO. David's work day, job, family, and title is the exact same as mine, software engineer. That's what we are. We're software engineers. Now, we have internal levels. My level is in the sort of manager leveling, and David's is in the exec leveling. But that doesn't sort of amplify or project itself out into our conversations. Now, of course, everyone knows that David's our CTO, but the idea is that you can walk up to me, you can walk up to David, you can walk up to anyone, regardless of your tenure, regardless of your function, and if you can proffer a good idea, or if you have a challenge, or you have an escalation, the sort of implied rigidity of a barrier is not there. I find that that helps us move a lot more quickly.

[00:16:35] KP: You mentioned joining an operations team that was already assembled. Can you talk about what it's like to join something that's has that momentum as opposed to starting a greenfield project?

[00:16:46] TT: Yeah, I've done both and I've since done both at Stripe. And I think the thing that I learned as a lesson, and one that I have to keep relearning, to be honest, is that every change is an inflection that you have to sort of take the time to consider and understand how does that inflection implicate you, and how does it impact others? I think that also segues into this notion that intent and impact are always different.

So, what I intend to do, you and I are having a conversation right now. And if I say, “Kyle, fill in the blank of the thing that I need you to do”, what I intend to do is perhaps to enable you and to give you support, and to tell you, “This is an opportunity for something that would be great. And if you invest your time in it, it's going to be wonderful.” What you may hear and interpret is, “Tramale just told me to go do something. I better go get it done.” And I think a lot of leaders, actually, it doesn't even have to be that sort of the “leadership conversation”, just a lot of folks who might implicate others and work that needs to be done, need to be clear and understand that what your intent and what your impact might be, are often not congruent, unless you are deliberately taking the time to make sure that there is mutual understanding on both sides.

So, to your question, joining a team that's already functioning and high functioning, I think you really had to take care to come in to honor the work that's been done and respect the work. I actually had a conversation with a peer of mine, who was vice president of Microsoft, who had recently joined Stripe, and I mentioned to them like, “Hey, in some of your language, the way that you're coming across, it sounds like you are – I totally understand you're trying to support the team. But it sounds like you're being like very directive. And it almost sounds like you don't understand the work that has come prior, like 10, 11 years of solid development by really smart people that have blood, sweat, and tears into a lot of the scaffolding.”

Sure, there are still gaps. And sure, there are still opportunities for growth, and maybe even the need to have some dramatic change and some of the ways and behaviors that we have implicated and the design of these systems and services. But let's make sure that we're using the type of language that, again, is inclusive, is supportive, it's giving people sort of recognition for the good, solid work that they've done. And again, like I said, at the beginning of this, I have to remind myself of that constantly, because I think it's a challenge. We are hired in these roles, engineer, leader, and everything that intersects and is in between in order to make a difference. And many of us have a bias for engagement and for action. If you're moving too fast, you want to make sure that you're not sort of rolling over people and leaving a trail of dead bodies behind you, which is a horrible, horrible, anachronistic metaphor. But still, I think, fairly impression that you want to make sure that you're giving people the space to teach you what they've done for you to integrate into an environment well, and then contribute and earn trust. We could talk a lot about trust, if you'd like. But that's one of the key factors for me joining a team that's already operating well.

[00:20:16] KP: Yeah, let's get into trust, actually. I mean, there's trust at many levels. There's trust between manager and mentee, and between departments. Where are the areas you focus?

[00:20:16] TT: So, I always use the phrase earned trust, and I know that sometimes rubs people the wrong way, because many people believe that trust is implicit. I don't. I think that there's a real good reason why. If I just joined your team, even if we've been working in the same organization for tens of years, you don't know me. You don't know the way that I'm intending to show up as this new function be it, leadership, again, I'm subordinate to you, doesn't matter.

The dynamics of our relationship have just changed. Because there again, it's one of those inflections. And so, the environment has changed, the expectations have changed.

We have to be super crisp and clear about our expectations on both sides. We have to be super crisp and clear on why we're doing what we're doing. Who's our user? Why are we doing this? What metrics drive the intent and purposeful work that we're doing? And we have to be really clear on what brings each of us joy? Why are we doing this individually? And how are we intending to grow individually and as a collective as a team? And all of those conversations can be had, if there's a relationship between the people who have the relationship, or who have this new connection, or going through this new inflection, maybe it's not that difficult.

In many cases, I think there is probably the need for at least one crucial conversation within sort of the litany of things that I just described. And when you recognize that those opportunities exist for having the conversation, you sort of recognize and see that, "Ah, there is an opportunity to either reinforce, reengage or strengthen the trust narrative that we have between the two of us, or between myself and the team, or even between myself and an organization." And trust is, as I sometimes like to say, a bit of a leaky abstraction, because people all have sort of a definition. And of course, there's a Webster's canonical according to Hoyle's definition of trust that you can rely upon. But I think if you were to sit down with five people and pull them, what does trust mean to you? You would likely get five nuanced answers. And so, it's really important to have those crisp, transparent, and sometimes vulnerable conversations. So that you can, again, either strengthen the trust that you have, or build that trust between you and whatever object that you're intending to have trust in an organization, a team or another individual.

[00:23:19] KP: Has that process changed a lot in light of the pandemic, and remote work and things along these lines?

[00:23:25] TT: I do think that the pandemic has definitely added nuance. And it really presents a number of challenges for everyone, not just leaders, not just people who are trying to be led or supported, sponsored, championed, what have you. There is something about the sort of two-dimensional projection of a person, even though you can see motion and you can hear audio and you you sort of know that it's a very real person that you're talking to. But there is this uncanny valley of the distance and separation that, some Zoom or Teams or Meet or what have you. The thing that's marshaling the conversation between individuals or individuals and teams or teams in teams, it allows for – you can lose just a lot of context and a lot of again, nuance, if you allow it to happen that way.

It'd be super simple to just say, “Hey, let's get on a Zoom. Let's get on to Meet or whatever, and we'll have a chat and any issue that we need to talk through, it'll be fine. We'll talk through it.” But you can't see body language as effectively as you could in person, or you may not have the opportunity as you might in person to have those hallway conversations sort of serendipitous chats, where at Stripe we like to call them collisions, where you just sort of walk into someone and you're like, “Oh, yeah, well, you know what we should have had a bit more of a deeper conversation about this one topic, maybe you want to go grab lunch together and talk it through?”

Now, that all being said, there are plenty of teams that are highly effective, and doing a tremendous amount of good work who have been fully remote far before the pandemic. And the intrepid leader who needs to enable his or her team will be one that will look to those already successful organizations and try to learn as much as they can, and try to take those learnings to test them against their own organization, and then see where their own organization can iterate, and innovate, to become much better at having those conversations that might have been in person persistent in the past, and now might be hybrid, or might be remote forward, or might be slowly returning to office, but not quite fully back to the office because, “Hey, we're still in a pandemic.”

So, yes, the pandemic has influenced how we build trust. But I think it's just like any other inflection, you have to take the time to listen, to learn, and to build off of those learning so that you can move forward faster.

[00:26:17] KP: Well, I find that most professional, software engineers especially or people adjacent to that in technology, they're not just people who get a job and they think this is where I'll be for the next 40 years. Everyone's on a career journey, and a lot of ways, especially around learning and being challenged. Do you have any thoughts on how to create the right paths for people?

[00:26:36] TT: Pathing is an individualized thing. It's something where, you know, you have to have support. I think you are building a structure and a path for yourself. But there are many who had to do that exact same thing, who've gone through that exact same exercise. And though their path may not be exactly as yours will be, there are probably many lessons that you can learn. You can sort of think of it, as I'm a distributed systems person. So, I think a lot about fault domains and failure modes. I think about learning objectively how to avoid those, but also realizing that they're going to happen. Any system of sufficient complexity is going to break. We all know this. It's just part of the universal law, scalability.

I think human systems are the most complicated systems that we'll ever be exposed to. So, regardless of any path that you follow, in fact, famously, there is the manager's path. **[inaudible 00:27:48]** has written this tome that many leaders in the engineering space have read, and have learned quite a bit from. I am hopeful for, it's a great book, everyone should read it. But remember that it's just guidance and you are likely going to in your career, and your journey as a leader, and your ability to or your journey to sort of build a reasonably great and comfortable situation for yourself, you're going to hit those failure modes, and you're going to hit those fault domains. One of the more interesting conversations, I think, to have is, have you built your resiliency muscle? Are you able to be adaptive and aware and introspective so that when you hit those fault domains and those failure modes, you learn about what just happened? And how to avoid it for yourself or how to navigate it more effectively, the next time something of that shape occurs?

[00:28:51] KP: Do you guys do blameless post mortems, or any process like that, that you find effective?

[00:28:56] TT: Absolutely. And I think that one thing I'll say about blameless post mortems that we don't talk about a lot in the industry is attribution. I think, obviously, you don't want to walk into a situation where people are blame throwing because of an event. There are so many contributing factors. In fact, the even the language that we use, has to comport to this notion of the the system is complex, the service substrate is complicated, and it's likely that it was no individual deliberate action that caused this fault to happen. It was a number of contributing factors and understanding those factors is important.

But, whilst you're understanding those factors, attribution of what occurred, again the compute control plane or the whatever software defined network topology was, for some reason, the assumption that we had about that were incorrect. And understanding where those misplace assumptions are, is really important and really critical. But that's not blame. That's just attribution. So yes, indeed. We do have blameless post mortems, but with high attribution so that we are clear on where we specifically need to build improvements. And again, hopefully avoid similar shaped incidents in the future.

[00:30:28] KP: Well, I realize a lot of the infrastructure might be, private NDA type conversation. But I'm curious if there's any aspects of it that you would stress that help make you guys be a more resilient organization.

[00:30:41] TT: I think one of the things that helps us with resiliency is really being rigorous about how we test and how – and when I say test, I don't mean strictly performance while I am testing or having some QA team. At Stripe engineers and service providers and product organizations are chiefly responsible for testing their own services, their own code, their own products. Tests, both written into their software, and tests, from the perspective of volume testing, or heuristic testing, or fuzzing and what have you.

We also have horizontal organizations. For instance, we have a very rigorous, and I think, high competency reliability organization that spends a lot of its time helping teams understand attribution, and helping teams understand where there are opportunities to dive deeper into areas of potential failure mode, and fault domain. I think the more that you have open conversations, one of my favorite conversations that we have weekly at Stripe is operations review, ops review. It happens every week on a Thursday. We shift the time just because we have different geos and we want to make sure that we give as many folks within the tech org, as we call it, which is just the amalgam of engineering and product, the opportunity to participate.

It's a conversation that I just really am thrilled to be a part of every week because you learn so much, things that I wouldn't maybe not within my pillar, which we didn't talk about specifically what I do at Stripe, but I am responsible for networking, compliance and a new infrastructure product that we are bootstrapping. And being able to hear my partners on billing or on issuing, or any of the other products or any of the platform enablement for those products, talk about the types of failure modes that they're seeing, and as I am responsible for the service mesh. If I hear something about, “Oh, you know, leader election for these services is taking an inordinately high amount of time. And my metrics for whatever reason, we're not surfacing that type of issue. I can then go deeper.” Like, “Oh, how did you notice that?” “Oh, if you look at this in our end”, I think it's safe to say that we have a ton of different telemetry panes of glass where we can see what's happening within the system.

But Signal FS is one of those things that we use this and someone might point me to a Signal FS graphic. Yeah, if you look here, over time, you can see that the sort of service level commitment that we have in the SLI is that we have driving towards that service commitment were violated on these days. And this is what we saw, and this is how we resolved it. And then I can take that back to my team and have a deeper conversation about oh, “How do we make it so that that team doesn't even have to care about looking at leader election and the amount of latency that there might be built within the system as services are trying to communicate with each other?” Because they probably shouldn't need to. And and then we can have deeper conversations about building more reliability and committing to higher service levels and the areas that we enable for those teams.

So, resiliency for me is, and I'll probably say this quite a bit in our conversation, it's about the human process of opening communications, and having conversations about what the systems are telling you and how different orgs have built the structures of their organizations and telemetry controls in order to get information about the state of how their services are performing, and how you, especially, an infrastructure engineer can learn from what they've done, so that you can make their lives better, easier, and so that they can move with higher velocity.

[00:34:37] KP: How much of that, I guess forecasting or maybe prognosticating, would be a better word about resources and infrastructure? Do you have to do versus assume that product and development teams will be ahead of the game on as they build something? Are you coming in and just supporting them or do you try and anticipate their needs?

[00:34:56] TT: This is a great question. This is something we talk about all the time and here's why it's important. If you are not able to anticipate the needs of your users, and I do call them my users, because having that sort of user centricity and customer focus, bless the hearts of all of the AWS/Amazon folks that really introduce this mode of thinking into our vernacular, is something I think it's critically important. And I do give them credit, because I think though, they might not have been the first to talk so heavily about user centricity, they've definitely been the loudest.

Having that sense of where your customer is going, knowing what their needs are, and then getting so comfortable with where they are and where they're going, that you can almost anticipate what they're going to need, helps greatly within those conversations. This is the intersection of art and science, because we can look at scale metrics, and the factors that are growing like we're getting more RPS request per second. We are seeing that services that we assume to be highly durable, may not be durable as we reach, again, certain tiers of engagement and inflection. We are seeing that, "Wow, for whatever reason, we're running console on this fleet of servers and console", is not to throw any stones at my friends at HashiCorp, I'm just using this as an example.

Console is not keeping up with the number of service discovery responsibilities, the the number of services that are starting to proliferate within the environment or something like that, of that shape. Those are core infrastructure metrics that obviously I would, maybe not obviously, but definitely I would be probably dereliction of duty, if I'm not looking after those. But at the same time, "Oh, wow, that part of the business over there is doing really well, like extraordinarily well." And I could just look at business metrics, like the daily average users, the amount of revenue that they're generating, the amount of service requests and the types of service requests that are coming in. I think the important thing for an engineering org leader, especially an engineering org leader in a scale – it doesn't even really have to be a scale organization, it could be a startup, it could be a three-person garage shop, if you will. Knowing what the business implications of what your products and services are doing, and how that might influence and impact the infrastructure that enables those products and services, is critically important. And that's everybody's responsibility, right?

If I were just a network person, I only care about networks, and I only care about the Vnlbs, the load balancers, and I only care about the the connection and durability of connections and the amount of packets that I'm sending back and forth, ingressing into our service substrate. I'm probably again, engaged in a dereliction of duty.

So, to answer your question quite directly, it's important to get out and get into your customer's business, to understand them, to a point where you can absolutely avoid sort of the reactionary motion where, "Oh, you guys need to go do this thing, because the business has grown so dramatically, and you're not keeping up, that's like a horrible place to be." So, keeping parody and keeping pace is great. But if you can get ahead of them, and say, "We've already built something that will enable you and help you to move faster and will scale with your business." That's sort of the Holy Grail. That's the perfect place to be. That also helps with, to your question of capacity, and planning for resourcing, so that you can, hopefully, again, build out sustainable and durable systems to support platform and product teams.

[00:38:54] KP: Well, they're scalable and durable for infrastructure, also for team size, which I think can be more challenging to scale at times. Can you talk a little bit about, I don't know if you're right sized currently, or if you're growing, but what you envision for adding new professionals to the team?

[00:39:10] TT: Yeah, this one's a difficult one, too, because what is the right size? I think, Will Larson, who used to work at Stripe, and maybe within our little industry, famously wrote this book, recently called *An Elegant Puzzle*. I was actually there as well was writing it and I would travel down to San Francisco and sit next to him while I was visiting. Literally, while he was writing this out, and I had been a longtime reader of his blog. He has a perspective on what is sort of a decent way in his experience of building teams and sizing teams and then deciding when you need to split teams. I both agree and disagree with everything Will writes. I think it's all really, depends upon context.

So, Will talk talks about this rule of eight. Once the team gets to about eight or nine folks, that's probably a good colossal size. That's a good size for an engineering manager, and a team of seven or eight folks, can durably sustain itself, can do an on call rotation that doesn't cause too much over indexing on a single individual or single few individuals carrying the pager, is resilient in the face of a lot of keep the lights on work, KTLO, and can also innovate and deliver features and services with some reasonably good last today.

I think that's great. But I also think that there are examples, for instance, famously at Google, of teams that might have 20 people on them reporting to a single leader. I don't recommend that for everyone. In fact, there are also famously many folks who absolutely detest that model. But you cannot argue with the results of Google and what it's been able to provide both the industry, its users, and just as a structure. There are a number of people who see it as the epitome of technical and engineering advancement within our industry.

So, you could then have a really healthy conversation about is the Google way, the right way? Is what Will proffers the right way? Is the so called two pizza team, that we hear about from Amazon the right way? I think the right way is the way that works for you and your organization. You have to figure out how to – to your question, which I think has a point embedded in it, how to be sustainable for those teams, how to make sure that people have the ability to find their purpose, to build mastery, and to have some autonomy so that they can continue to do really good work. And that you remove toil from the environment so that you're not burning people out, and that you were giving folks to the earlier point in our conversation, the space in agency enablement to build a career path and a trajectory for themselves, so that you're not seeing high attrition. I think attrition is one of those sort of proxy metrics that you can use to see whether or not you're doing the right thing. If people are leaving your team, or no one wants to join your team, you're probably doing something bad.

[00:42:29] KP: Are there any particular technical skills, I guess placed in context of your work at Stripe that you would be looking for if you were hiring for a new candidate?

[00:42:39] TT: For my organization, technical skills are, I looked at aptitude. Sure, if you are going developer, and you are fairly competent in AWS network primitives, if you have done at least a little bit, if not a lot of work within distributed systems, and scaled infrastructure, with many, many different types of hosts, and many different traffic patterns that you might see throughout the globe. So, people connecting mobile devices, and high frequency trading devices, or, like just any types of patterns, where there are different gradations of traffic that you're ingressing. And the need to make sure that you are highly reliable, that you've got item potency built into API requests and things of that shape, then you're probably going to have a good time at Stripe.

But if you are not familiar with all of those things that I just said, but maybe one of them, let's just say you're really solid go developer. I could probably teach you the rest. If you come in and you're able to learn and have what I like to say, you have high aptitude and learning is just part of your DNA. Or again, in our industry, I think many people say you have that so-called growth mindset, you will also likely have a good time at Stripe.

I was looking at a video earlier in the week by Simons and that was one of those thought leaders in leadership. And it's one that I've seen reference quite frequently, but it talks about Navy SEALs and how Navy SEALs determine who the best of the best of the best are. And super quick, the way, the punchline of the presentation is that Navy SEALs don't look for people with high competency in a functional area but low trust. They define trust as, “Do I trust you with my life? Do I trust you with my wife?” Like, is the thing that they sort of slightly remark on that.

But the point being is if you look at this sort of x, y axis and you have a that's high competency, but low trust, which is what you see, arguably in a lot of situations where it's just like the drive is to ship to ship to ship to get things done, and you'll excuse the expletive, but sort of the asshole rich environment, you don't want that, right? No one wants that. No one wants to be in that environment where it's sort of rich with toxicity. What they say is, even if a person is lower in competency, but still high trust, I want that person on my team, right? Because competencies can be taught, right?

Things change all the time. In a few months, we're going to be hit with Go 1.1.8, right? And now everybody's going to have to learn how to use Generics and Go. I'm not going to expect that I'm going to interview unless they're on the Golang team at Google, or a contributor to Golang, I'm not going to expect that most of the folks that I'm interviewing are going to have deep, deep, deep expertise in Generics and Go. So, and so I probably won't test them for that. But I will test their aptitude and I will test their ability to maybe read a code base that has Generics and reason and ask me questions and say, “Well, why would you make that choice?” Really good question. Because not just Go C++, whatever, any language that supports Generics, they're not always the right answer. So, ask me about why I decided to do that this time, as opposed to using interfaces or reflection.

[00:46:23] KP: Makes sense. Well, maybe to wind down, we could talk a little bit about corporate culture. I'm curious if there's anything about Stripe's culture that – I mean, I guess every company is a little different. But is there anything about its payment space and the type of work you do that shapes it, or any ways that you see it evolve or feel you can be a fulcrum for that?

[00:46:42] TT: So, this, again, goes back to the beginning of our conversation, why am I at Stripe? Like many of us, guys in the industry and who have been operating, sort of understanding how to operate, I think is what I want to say, within the industry, a lot of opportunities. I could be in at a lot of different places. But I choose to be at this one, because I truly believe in the people and the mission as articulated by our founders and reflected by literally everyone that I get to engage with every day, all the way through to our users, to people who are using our products and services. And here's the thing about it.

Stripe, yes, it started as a payment services provider, providing these payment gateway services to make it easier to buy things on the internet. Super simple concept that just had, again, arguably horrible execution with the products and services that were available at the time when the coalescence decided to build out their solution framework. What it has evolved into is this notion of the Global Payments and Treasury Network. What does that mean?

Essentially, what it means is, it's the democratization of access to the financial system globally. If through software, we can make it easy for a person in Kenya or in Argentina, or in Uzbekistan, or in Detroit, Michigan, to take an idea and turn that into a business and to scale that business into an enterprise, that's empowerment, and that's global empowerment. Being able to be a part of that requires a lot of rigor, a lot of technical expertise, a lot of functional and high competency expertise. But it also requires an environment of high trust, because you're dealing with people's money. And people's money is their livelihood, and their ability to sort of define those paths, those objectives for themselves. And what I find about Stripe's culture, its operating and leadership principles is that they are framed in a way that amplifies the attitudes, the norms and the behaviors that are expected in order to deliver on that promise.

I love being there. I just love being surrounded by people who are mission focused, who understand what and why they are doing, who they are doing those things for. I don't begrudge my friends who work in ad tech, which has delivered a ton of remunerative advantage for a number of companies, and those companies have subsequently produced a lot of technical artifacts that I quite frankly use in the business today to help further my teams. Again, objectively the types of things that we're trying to deliver on behalf of our users internally, are a result of millions and billions of dollars that the industry has reaped from ad tech.

But this notion that I am able to help a business grow so that maybe a person who's got a great idea, and up to this point, didn't see a path, we're getting that idea executed online or in any sort of channel, brick and mortar or what have you, now can do that because of the software that we provide and they can put food on their table every night. That was the reason I joined Stripe and that's the reason I stay at Stripe, and that's the reason I want to see Stripe continue to build the types of system structures and variants and products that will further enable folks like the people that I grew up with back in Detroit, to capture some of the value and to be a part of the future, the inevitable future that we're all marching rapidly towards.

[00:50:39] KP: Well, it's a great vision. I definitely want to keep my eye on Stripe and learn more about this somewhat secretive project you alluded to in the future. Well, Tramale, thank you so much for taking the time to coming to Software Engineering Daily.

[00:50:43] TT: Thank you so much, it’s been a pleasure.

[END]