Life of An SRE with Sabrina Farmer

[THEME MUSIC]

MP: Hello, and welcome to the Google SRE Podcast, or as we affectionately refer to it, the "Prodcast." I'm your host, MP. And here with me as co-host is Rita.

RITA: Hello, I'm happy to be here again.

MP: And I am very excited for today's episode because for the first time in the history of the prodcast, we have an executive guest. Let's hear from her.

SABRINA: Hi, everybody. I'm Sabrina Farmer. And I am VP of Engineering here at Google for SRE.

MP: So I think we should just start off with the most obvious question while we have you here. What does a VP of SRE do?

SABRINA: [LAUGHS] Every day is a little bit different. I think when you think about a VP position at Google, really, all leaders actually need to make decisions based on Google, their product area, their team, and their people. And I think at the VP level, your starting point is always Google.

So my job is to make decisions on behalf of Google across my portfolio. So my portfolio here at Google covers most of Google's beloved products, everything from Search and Geo, but also our platforms like Chrome and the Play Store, and all the infrastructure for the products, so the authentication systems, the abuse systems, data analytics, and things like that. And so with that broad view, I'm looking at what problems are each of the product areas having, and then what does this tell me about the opportunity to solve those problems at a higher level. So I'm constantly looking for that.

But as any good SRE, we're the problem solvers. And so I would say, pretty much guaranteed every day, I'm hearing about all the problems that are happening across the product areas. And my job is to help resolve them as quickly as possible. And they could be technical problems, org problems, or conflicts between teams as well.

RITA: If you're the VP of Engineering, I'm wondering how much time you spend actually getting involved in engineering-type decisions versus maybe other business outcome decisions as you were mentioning, like you always keep Google in mind when you are going about in your day-to-day life. What's the balance between those two? Or does it change a lot depending on the time?

SABRINA: I am happiest when I'm reading design docs or getting updates on the different projects that are happening across my teams. I enjoy the technical problem solving much more so than all the other areas. But I would say it's pretty mixed. I'm not a tech lead. I don't need to be the tech leads at the various products. And I want them to feel empowered.

So while I'm reading all these things, a lot of times, I get design docs because people want me to sponsor their work. They want me to hear their idea and give them feedback about how to get it prioritized, how to get it staffed. And so that's often why they send me the design docs. But I enjoy reading the technical aspects of how people are approaching problems.

And it's good to read these and see how people are approaching problems because that's the opportunity for me. If I see people trying to solve the same problems, I can connect them. I can bring their ideas together because that will be a better outcome for Google. But I would say it's pretty mixed. And it changes week over week, honestly.

RITA: And so you're mentioning that you often see designs from tech leads within your team. I'm sure they're very smart people. And they all have very good opinions about what should be done. But sometimes their opinions would be different. How do you resolve conflict and issues when it comes up between engineering teams?

SABRINA: Yes, it's true. Especially at Google, people are very empowered. And you might see a problem. And a lot of people want to jump on the problem. And I will observe often that people are solving from within their world. And it can be very siloed approach. And they believe that they are solving for all of Google. But in reality, they're solving a problem very specific to their product area. And they're extrapolating that everyone has the same experience as their product.

And I think sometimes you need to make sure they understand that they're looking at it from one perspective and then challenge them to broaden their perspective. And that's really when, if people are solving the same problems, you want to bring them together. And I think sometimes, you have to help them understand that just talking to each other is not enough. You want to actually go in as a collaboration because people tend to want to carve it up and keep it within their domain because that is what they are in control for. That is where they feel empowered.

And as a leader, what you want to do is help them feel empowered more broadly even though it makes the problem a little bit more challenging. When you need to negotiate across multiple teams, it's much harder to find that common ground and agree on it. But if the groups are committed to doing that, that's how we get the better outcome for Google.

And sometimes, you facilitate that. And sometimes, your job is to realize that neither one will produce the generalized solution, the solution that will meet all. And then you help them scope it to stay within the domain that they are solving for. And that sometimes is the right answer. Not every solution should be generalized because especially at Google, our product areas have different pressures on them. They have different outcomes that they're shooting for. It's great when we find common ground. But that is not always the right answer for us.

And my job is to be the judgment of that over time. Sometimes, what I see is I will say, hey, your solution is not ready to be generalized. Solve it for your team first. And then when we get to that point that it's either mature or everyone else has finally come to where they need to to make that solution more accessible to them, I'm looking out for that timing. Because sometimes, you can have a really good idea and the timing is all wrong.

And a year later, everyone is now ready for that solution. And I think what you want to do is you want to be able to pause things successfully because fighting for a year is not a better outcome than just pausing something for a year and then bringing it back when everyone is ready for that idea.

RITA: That's really insightful. And it sounds like you have to have still such a breadth of technical knowledge to be able to tell when it is the right time to be making these decisions and promoting these solutions more broadly. Even on the smaller scope that I'm working with, I have seen teams propose solutions to the same problem that persists year over year. And it just doesn't work until finally you have a missing piece that is ready for it to be working in production for it to be generalized. And the problem didn't change much. But the solution is just more ready.

And it is about the timing. Do you have any concrete examples of resolving these kinds of issues?

SABRINA: If we were to talk about Google, one of the things that when we're building currently in SRE, we're driving an effort of convergence, which is how everyone manages their systems in production. My team started out-- when I was just running what is now known as Workspace, each of those products was actually much more like a silo than a suite.

So we started, because everyone had its own release system, and everyone had its own way of managing capacity, and everyone had their own performance metrics and things like that, and as the lead, I was expected to staff each of those different efforts within each of the silos. So for me, that was like, wow, that's really inefficient. So I brought everyone together. And I said, look, we really have to think about sharing our solutions.

That was actually really, really hard for people in SRE at the time to think about giving up the solution that they had built. So we started very slowly. And we created a configuration management system. And we tested it in one part of the Workspace world. And immediately, everyone was like, we should use this for everyone. And I was like, no.

One, not everyone was ready. Two, we hadn't proved that it would deliver on all of the value propositions that people thought were on the table. And I think because we kept it within a single problem space, we worked through a lot of those problems that had we gone more broadly, we would just have 10 times, 100 times more different scenarios to solve for. And so you really want to make sure you have a true proof of concept that if we had tried to push it on day one, we would have never gotten through all the problems.

I would say that for the configuration framework that I'm talking about, we had learned that lesson the hard way before we had done that. We had created a tool that would help people push things very intentionally into production. So it was like, what do you want production to look like? The system would, when a machine came up, it would put it into the state you wanted it to be in.

But we went broad with that product right away. And in retrospect, we realized like, oh, we should have narrowed the focus, worked out how the whole thing would work for a single large system before we tried to do multiple large systems at the same time. And that project took a lot longer to take hold across the fleet to then our configuration system, where we had worked out all the bugs within a single domain. And then we just had to focus on the generalization problem after the fact, instead of at the same time you're trying to figure out what's all the functionality and how do you generalize it at the same time, the complexity and coordination is very, very difficult.

Whereas doing a full end-to-end proof of concept and then generalizing that, you actually realize your vision much quicker, even though it feels counterintuitive to the people who are very

ambitious about their idea. And I think, as a leader, one of the things you have to do is help people realize this is a really good idea. We are driving towards your vision. It's going to be slower in the beginning and then faster at the end. But that's sometimes hard for people to believe in and trust. And so that's why my job is hard. [LAUGHS]

RITA: At Google, we're always encouraged to think about things at scale. I mean, it's nice that you have a solution for this one particular case. But what about all the generalized cases? And sometimes, I can see very enthusiastic and very smart engineers getting ahead of them and thinking about the future when they have not even solved the problem fully for the present.

SABRINA: Yeah.

MP: I'm curious about the opposite. We've heard a lot about overscoping, and trying to make sure you show the right amount of restraint, and proving the idea in a small enough domain to be manageable. Have you encountered cases where you have to really push to get the scope of something grown? That it was kept too narrowly focused and you really had to drive for it to be a larger project?

SABRINA: I've also seen that. Less common at Google because people want to solve the big problems, typically. But I have seen where a tech lead will have an idea, they're driving that idea, and they don't want the distractions from external parties. In that case, they have a really good idea. It actually can be bigger.

So you get them to work out what they want. But then you need to give them a lot of support so that they're comfortable with what will be criticism that comes from external parties. That it's often the individuals uncomfortable with that moment, where you have to work out everything everyone else wants. And it's going to change what they have in their head.

And what I found is you just give them much more support. You help coordinate the communication across different teams because it's often they're not comfortable with that. But if you give them a program manager, for example, who can coordinate these things, collect requirements, that can help them realize that they can actually scale this without the burden totally falling on them to work out what all these additional requirements will look like.

Help them understand they won't have to do this on their own and that there's a bigger picture here. So it depends on the individual and what's making them uncomfortable about expanding the scope, and then supporting them to make sure that their worst case scenario is protected and mitigated.

MP: Yeah, I definitely know from my own experiences that increasing of scope, you have to be able to handle a lot of feedback. And that's not always the most pleasant thing if you're very proud of an idea and now you have to hear everyone's, well, this could be different, this could be different.

SABRINA: But sometimes, it's that disagreement where the real breakthroughs come out. Creating a very safe environment for that to happen is critically important for any leader at any level. How do you bring disagreements into your team but have it result in a positive outcome and a better outcome? Because I do think these different perspectives make better products when you can factor them in ahead of time. And it's a lot harder to retrofit anything you build later on. So just being open early on to that kind of feedback can really help you make a better design and a more sustainable design in the future.

MP: How do you think this decision-making, and decision-evolving, design-evolving process, has that always been how has functioned? Has that always been how Google development more broadly has functioned? Have you seen cultural changes within SRE over the years on how we deal with these large scope decisions?

SABRINA: I mean, I think it's definitely evolved. I don't think it's evolved in one direction, really. I don't think it's gotten necessarily always better over time. I think there's been different points in behavior of how this is done.

So I joined Google in 2005. I joined Google as an IC. And one of the first things that I was introduced to-- because we had all these design doc templates. But before you ever wrote your design doc, when I started, you had a premeeting. And this was where you had an idea and you brought together the stakeholders. And you briefly described your idea. And people gave you feedback. And then you could work through some of the details. People gave you feedback.

And you had that meeting prior to the design doc. So then you'd write the design doc knowing what people were looking for. So you could factor that into your design and into your ideas. And your design doc, at that point, would be fully vetted. And then the review you got after was really like, OK now that I see it end to end, what is my feedback? That was what it was like when I joined.

I found over time, what happened is when people had an idea, they did a full design doc in isolation. And then they would ask for feedback. And we started to call it the monkey knife fight because what would happen was everybody was then on that design doc with all the different problems. And it was just such a uncomfortable experience because everyone was coming in cold. And it all sounded really harsh.

So, I've seen it change. In order for the feedback to not come across so harsh, then what you did had multiple copies of your design doc. And you would send it out to different audiences. And then you'd have to merge the feedback after. And so there's a lot of ways in which it's changed over time.

I still like when people get together and brainstorm as a group because that is more supportive of the person who's writing the design, even though it requires a little bit more time. It feels like,

oh, I can just write my design doc. And that'll be quicker. But again, if you get the early feedback on your idea, sometimes, you might decide, hey, this isn't going to work at all.

So you don't write the design. You don't write it down. Or that feedback takes you in another direction. And that can save time, even though it feels like it's going to take more time to do that, those premeetings, that prefeedback. I still think what we see is people are doing that whole design doc. And they're sharing their idea as a fully vetted thought. And so we have a lot more designs than we have solutions because of that. But maybe it's all good.

RITA: One pattern that I'm seeing now that mentioned this evolution of how we make decisions and how we design solutions is some hybrid of both approaches because I guess with the distributed nature of SRE teams, it can be quite difficult to get everybody, all the stakeholders from multiple teams, into the same room. So I've seen the pattern of you start within your own team locally or maybe across one site, you pitch the idea to them. And you come up with something that your team, at least, has alignment on.

And then you start with the external partner teams and send it to them for offline review, give them some more time to think about it to come up with feedback and suggestions. And I think that kind of fulfills the first round of bouncing ideas off of other people. Just so you can vet the initial solutions and cut the ones that are pretty obviously not feasible in this context. And that also gives you the benefit of not having to just work around the real practicalities of scheduling a meeting with many, many people, especially if you have senior stakeholders in multiple time zones, and giving them the time to give feedback on their own time.

SABRINA: Yeah, it was definitely a lot easier when I started. In 2005, we were pretty small back then.

RITA: You were talking about this process of decision making and coming to the right solutions for the problems. I was wondering how your experience as an SRE starting out as an Individual Contributor and rising up through the ranks through upper management informed your decision making. Because I imagine you have to make many more decisions now that are more impactful but you have less and less time to spend on those decisions. How do you make sure you're getting the right information to make sensible decisions in this case?

SABRINA: When I first started, we were very, very empowered. A lot of people talk about Google's bottoms-up approach to planning and ideas and things like that. And I think that that's true. But don't think it was ever an absolute. Work always came from both directions. And I think that sometimes gets lost in the lore of Google's culture.

I feel very, very strongly that engineers should be empowered. They have good ideas. That does not mean everyone is empowered to pursue all their ideas. But what we want is for everyone to feel comfortable bringing their ideas to the forefront, having discussions about them, talking about when might the right time be.

And so I spend a lot of time trying to get them to answer these questions. And I often will say, how are you thinking about this, this, and this, much more so how would I think about it. What are the things that I would want to see considered? And I try to get them to answer those questions along the way. Primarily so that they know what principles will I follow when I have to make a prioritization.

Because you cannot do everything, even though we are now a pretty large organization, not every idea can be pursued at the same time. And so we have to pace these ideas. For me as a leader, the most important thing to me, what's really important to me to do is to let people know how I would make decisions. So that they can understand when I make a decision that they don't like, what went into it ahead of time.

Because sometimes people are after the fact, wanting me to explain it. And that's fine. I can do that. But it is so much more valuable for them to understand how I make decisions, why I make them the way that I do so that they factor that in. And then that makes my job easier. They'll be happier with the outcome.

But I think sometimes if you just pursue something in isolation and you miss that what are the decisions or how are things going to be prioritized, you can very easily be disappointed in the outcome and even jaded, jaded in the fact that you don't feel empowered to work on the things that you think are the most important. So I do think that is a huge responsibility for leaders to set that foundation for how things will get done within the organization.

And I think as an IC, you want to understand what's going on around you. You want to be paying attention to that so that you can factor that into your consideration for what to work on next, et cetera. I think if you're only looking at your view and not looking at what are the external factors, then I think you're not really setting yourself up for success. So even if you are the most junior SRE, you don't just want to do the task. You want to understand why we're doing this.

You can have a mandate from on high. And you can execute against that mandate. But if you don't understand the why, why do we have this mandate? What problem is it solving? You're not actually going to have the context for when to use your judgment about sometimes the mandates are wrong and it needs to go in another way. But if you don't know what the intended outcome is, you won't see that for yourself.

RITA: I've definitely personally seen the difference between success and failure of a project is whether the context was understood and whether the intention behind the project was understood so that you can self steer. And at critical decision points, you go in the direction that matches the spirit of why this whole thing was started, rather than to the word of what needed to be done. So as an Individual Contributor, how can I know how my VP is making decisions? Is it through formal channels such as overall OKRs for the entire organization? Or are there also other channels that you communicate this?

SABRINA: I personally do a couple of things. I'll publish a strategy for my team. I'm very clear on what the guardrails are with my leaders. Because I want my leaders to be empowered. I want their leaders to be empowered. And so for me I don't say, hey, we're doing A, B, and C. I'll be like, hey, we're working between A and D. So that they have room to make their own decisions along the way.

But I try really, really hard to give what are the business context closest to your work. What am I hearing on the ground? What's probably some of the existential threats that might interrupt a business or things like that? And that can be stuff like, especially right now, I think most SREs understand that there's a lot of regulations coming within the different countries that you work that you need to really understand and spend time with.

And so when there's something new, we try to describe what's going on that these different boards or governments are trying to put regulation on, especially as we work across different countries. There's a lot of norms that differ between countries. And what's critically important, although we're a US company, is that we want to do business globally.

And so we need everyone to understand what are the different pressures on our business. How do we make it more accessible so you understand it? And then how do we help you understand when it's applied and when it's not applied? And so that's a big part of what the leadership team has to give to all of the ICs within their organization.

What are some of the things we need you to know about privacy so you make good decisions in your design reviews and you think about what systems are you splitting the data across? And how did they consider privacy? Because Google has a lot of data systems that have historically been data agnostic, being like we could index anything you sent us. But nowadays, we want you to treat the data differently. Because if that data belongs to a user, we don't want you to treat it as if it's something you just crawled on the web. We want you to treat it differently.

And we want our team members to understand the why of that, the value of us thinking about it proactively for our users, and for building trust with our users. If every mistake we make affects people's trust of us, and we have a huge responsibility to all of our users and globally that we understand the responsibility that we have and we act in accordance with that.

And so you should understand what's happening for your business. You should understand what pressures it's under. And our leadership team has to make sure that you are given that. And so sometimes, we do it with tech talks. Sometimes we do it in team meetings. What is coming up? What are they talking about at the executive level? And try and provide that context along the way. I don't know that we do it perfectly. But I do know that we try really hard to communicate to this and make it accessible to anyone open to hear it. But hopefully, that's everyone.

RITA: So looking back on your career, from your early days as an Individual Contributor that's

probably more focused on the direct execution of project, to now as an executive, where you're thinking a lot about how do I direct the people in my organization so that they can make good decisions so that they can understand the decisions I'm making, I'm wondering which skills that you've developed during your career are you the most appreciative of today and why.

SABRINA: Something that I worked really hard on and also something I'm constantly working on is communication. Being able to communicate your intent is so, so important. What I've also learned is it's really easy to miscommunicate, especially where I am now.

So when you're an IC and you're in your team, you're communicating to people who know you really well. And so you get a lot more forgiveness for misspeaking. And I would say, as a VP, you get very little room. It's super easy to misspeak. And then it's harder to correct because not everyone is going to know who you are as a person, even though I really try.

While I send email to every single person who joins my organization, that doesn't mean they know me. And it doesn't mean that they're going to, by default, trust me. That's something that I have to work on. So I think communication is such a critical skill.

Being able to know where your audience is, being connected to people is also very important. I think my technical writing course in college, I value that so much because I learned how to do feasibility studies in that class, which I think is so critical. I learned good technical writing, which, I think, really important. I love to document my ideas and get that feedback on paper and take people to the journey.

Before I write a strategy, a whole bunch of people have read it before. They read it along the way. They've given me feedback. And so I actually think that is probably one of the most valuable classes I ever took in college because I've used it my entire career, both from an IC to executive.

MP: I think feasibility study is a brand new phrase for me if you want to explain that a little bit more.

SABRINA: Oh yeah, oh my gosh, I tell everyone-- they have a really good idea. And I'll be like, OK, how can we discover whether or not that's going to be feasible? And you can do a small prototype. How would this work in different contexts? Because you want to do that research before you get something really far along.

And so I regularly, someone will have an idea, and then we'll map out what would be a good feasibility study for that. And we'll do that first before we build out a big plan to do something bigger. People love to jump to the bigger, though, even-- I'm working on this project. And everyone's like, oh, that's going to be so big. And I'm like, yeah, but I'm doing a feasibility study.

I'm going to make sure everybody can use this taxonomy or use this framework because before

I change the way they work, I actually want to know along the way if I'm on the right track. Because we can all think of ideas that we were like, oh, this is going to solve all the problems. And then you try to push out this new process to thousands of people, and then you realize, hey, that's not actually how people work.

So you really want to know how people would use it before you try to push something out to a large audience. And that's where feasibility study can be so, so valuable to you. Sometimes, I'll get a new process. And I'll wish they had done a feasibility study so that they could have seen that wow, that's not my workflow at all.

MP: I'm hearing some similarities to the idea of an MVP, a Minimum Viable Product. But I feel like there's a little bit of a nuance there where an MVP is kind of saying, can we do this thing. And the feasibility study is getting more at can we do this thing at scale. Or is there a bit more nuance there?

SABRINA: I think can we do it at scale, can we do it now? Sometimes it's a timing problem. It's not that it's a bad idea. It's just it's an idea before its time. And I think sometimes in a feasibility study is really where you uncover that. It's not just do people like the idea, but are people ready for the idea. Do we have the infrastructure to deliver on this idea? Which also actually is important for you to know now.

Because if you find out you don't have the infrastructure to pursue your idea, then maybe you pause your idea and spend time on how long will it take the infrastructure to be where you are? Sometimes, you don't even have the technology yet to pursue an idea. And doing that sort of feasibility study of do I have of everything that I need to execute on this idea? Are teams ready to hear that idea? I think you get it out during that time because a feasibility study is what are all the questions we would need to answer? Who's all the people? Where is their input?

What would have to be true for us to push this through? And sometimes, when you put that in front of a project, you're better able to estimate when could you launch. We've all seen people set launch dates. But if you haven't done a feasibility study, you're just putting a date out there. And that date is going to slip. Or you're going to take a bunch of compromises just to hit that date. I've been in all those situations. And that's really why this is something I prefer to teach people how to do early on.

MP: If you think about how SRE has changed and evolved over the last 10 years, what do you think the next 10 years of the evolution of SRE is going to look like?

SABRINA: When I think about how we do our work now versus how we used to do it, I'm very motivated by change. So I love change because I think change brings opportunity for us to really challenge what we think we know. I think SRE-- especially SRE at Google, there's a lot of best practices, and rules of the road, and how do you build teams, and how do you lay out your team, and how do you do on call? There's a lot of things that are very standard.

And I think that in order for us to evolve, we really have to take all of those things and question the assumptions in which these decisions were made. Just because we have a policy does not mean that's going to serve us into the future. It's only served us up to where we are now.

So one of the things is-- the question that I mentioned earlier, what would have to be true, is a question that I'm trying to challenge my team with. Which is we have an outage, all these things happen, we postmortem, we find all the AIs in there. We really think about how do we avoid this happening again.

But oftentimes, the postmortems are very specific to that service, the teams involved, and what happened on that day. And I think what I'm thinking about when I'm thinking about the future is not how do I avoid that again. But it's like, what would have to be true for this to never happen again? Or what would have to be true such that I don't have to get paged for this, or I don't have to observe it, or a user doesn't have to discover this?

I think that's the question that's really going to lead us to the next generation of infrastructure and the next generation of how we build teams and how we build our services. Because there's a lot of technology now that can really enable us to have the systems be a lot smarter than they were in the past. So what would have to be true for the system to self recover?

I think in the past, we might put some wrappers around our systems, or babysitters, or things like that, something looking out for what's happening. But how can the systems understand how it could fail, circumvent that, automatically say, hey, this is a hot shard, I should move things around. I should move user traffic around automatically. How do we build a system that can see that instead of building on top of it?

We're really good right now in creating layers on top of the system. We're working around all the problems. And what we're doing is building solutions on top of that. And I think we really need to start being like, how do we build this system to have these features, have this intuitive understanding of how it can exploit the risks of the system such that it can self repair?

And so I really like that way of thinking. I think as we work through these things, we're really discovering that if you built this again, you could build these features or those features to help the system be smarter in the future. And I think our products are getting very advanced. They're getting a lot smarter, very intuitive to what the user wants to write. I love when I'm responding to an email, and I'm like, yes, that's exactly the answer I want to say. And I think more technology is going to do that. And I think our infrastructure can do that too.

RITA: So in addition to improving the system so that it can continually evolve and find these kind of risks, what are your thoughts regarding the engineering experience working as an SRE on these systems? What are some areas that you would like to invest in over there?

SABRINA: There was a separation between what the developers did and what SRE did. And there was always this wall between the teams. And I think the first thing I do when I engage with a new product is I find out how high that wall is and I just start knocking it down. Because work should be moving in both directions between SRE and dev.

And SREs are so much more valuable to the dev team to have them on board at the beginning. Really being intentional about the resilience you want to build in the system, and what reliability are you targeting, what the user experience that you want to achieve, asking those questions up front and bringing SRE into the design phase is a much smoother path to success for your users and achieving what your goals are for the users.

It's so funny that we have this history of SREs coming in late. And I think that doesn't work going into the future. There's too many things coming. And so I want SRE to be comfortable in the design phase. I want them to be comfortable taking on the resilience part, engineering part of a new product, knowing how to do it, engaging in that, and collaborating across the aisle, so to speak, between dev and SRE.

We should be working from the same set of priorities, the same business objectives, and targeting the same experience for the user. I think SRE at Google, we have an additional mission to bring some consistency to production, to bring cross-functional viewpoint to the solutions that we're recommending. But I think we all need to act as a single team on behalf of our users. And so I think that's a much critical future for SRE.

RITA: In some ways, I can see that's also a cultural paradigm within SRE. And it can be a cultural thing of whether sometimes you feel like you don't have enough knowledge about the system to get involved early. Well, I guess nobody really has that knowledge. It's the concerns that we need to be addressing in terms of reliability and failure domains. Nobody has really thoroughly thought about it. So that really is the point where SRE should feel empowered to come in and get involved in the design early.

And so talking about the cultural aspect of SRE, one of my favorite things about being an SRE at Google is just the culture of transparency, of blameless postmortems, of not being afraid of failures, and learning from that, often, in a public way. So you can model that behavior for other people. I'm wondering on the culture and people aspect, which of these initiatives are you the most proud of? And what impact have they had?

SABRINA: I think that there's a lot about the SRE culture that I love. And it's evolved so much in my time here. It's almost been 18 years. And I think there was this well-articulated tension between dev and SRE. And that we would use budgets to negotiate between each other. And I think that's historically how we've acted. And I would say now, I find that the teams are much-- you can't really tell that there's a line between them. That's how I can tell we have a really healthy SRE-dev engagement.

And I think that is-- this collaboration, this look left and right, which we don't always get correct, and what I mean by that is before you start pursuing an idea, look to see who else is thinking about it and join forces when you can. I think we do that a lot more now than we did in the past. And I think that that is so critical for us in the future as well. So I think we still have some work to do there.

I think probably the best thing that I like about SRE, which is kind of weird because it's also the part that makes my job hard, is that SREs feel very empowered. They're very passionate people. They rarely don't have an opinion on something. You have to manage a lot of different opinions. But I love that. I love that about SRE. I never want to lose that.

I love that they're passionate. They'll fight for what they believe in. And so I don't want that to go away because I think we get better outcomes from it. I just think we need to make people understand that you can do that in a safe environment. And I think SRE at Google has been very good at, especially from leaders, is whatever question comes to us, we know we should answer it.

We don't have to like it. But we know we have a responsibility to answer it. And I think that that is something I want to never go away because it means people can ask us anything. That they will be safe to ask us really hard questions. And that's only true if we live up to that as leaders.

RITA: So for those folks who might be interested in engineering or SRE in particular and are not yet fully in the workforce, in your opinion, how can they best prepare to become an SRE?

SABRINA: I think that understanding what drives you is probably the most valuable lesson you can learn, regardless of what role you go into. Because I think SREs are problem solvers. They are curious. And they understand that you're never going to know 100%. And so I think in SRE, it's a constant learning.

So if you're comfortable learning new things, and change, and you see yourself as a problem solver, you should really consider SRE as a career. Because I think I am so appreciative in college I learned that I was super motivated to solve problems. I loved writing code. But code was a means to the end for me. And for me, it was that problem solving.

And so I've never been bored in SRE. There's always problems to be solved. I have seen people, they're only comfortable when they really understand the problem end to end. And I think, oftentimes, you might be better as a full-time SWE, where you have the one thing you're working on and you evolve that one thing. In SRE, there's no one thing you need to know.

You need to know the full stack. You need to know how everything works together. But a good SRE appreciates that you can learn any of these things along the way. You don't need to know them up front, mostly because we also have a community of people who really support each other.

So even if you're on call, you're rarely on call by yourself. Your whole team is there behind you. And we're always thinking about the user. So no one's by themselves, typically. And I think that that is another thing people should know. Because we'll talk to people and they'll be like, that sounds super interesting. But I don't know enough yet. And I'm always like, no, none of us know enough. You need to understand that we're all going to work on this together.

MP: What advice would you offer to those who are already in SRE and are looking to grow their scope and their leadership potential, either as ICs or on the managerial ladder? What advice would you have for them?

SABRINA: I think that I'm going to use the term networking, even though that's a weird term sometimes. I do think that talking to people, really challenging-- I'm doing something, it's going well, can I make it better? I think that's always been my guiding light, a little bit, is, do I think I can make this better?

And then how much time would it take? And then learning that decision of pursue this one or pursue another option. I think that it's really valuable to learn. So if you think, oh, management is in my future, you should really understand-- and I wish I had done this early on-- is, what does management mean? What's involved?

Because it's not really that you're now the boss and you get to tell everyone what to do. I think I thought that early on in my career the first time I became a manager. And I went back and forth, IC, manager, and then I was like, oh this sucks, I'm going to IC, just my work, because as a manager, yeah, you get to tell people what to do. But you also work for all the people who are in your organization.

So I think about this all the time: that's a lot to take on. You need to know that you're ready to do that. And I think that spending time learning about the role before you jump into it is really, really valuable. I was not so intentional, I think, with my career. And I think if you look at some of what I've done at Google is over time, I created these career planning guides for people to understand what's expected at the different level.

So that I'd have team members say, hey, I want to get promoted. And I would be like, OK, this is what your new job is going to look like. Is that what you want? Because sometimes, it's not. And you don't want to find out after the fact. Because then what happens? You have to change directions. And that's a lot harder.

So I think talking to people, learning about what's required at the next level, what does it look like, and then making intentional choices is really, really valuable. The other thing I think, is understanding what support you need. So if you have an idea and you want to generalize it, I said, often, I'll get design docs because people will want my sponsorship for that because they want to put my name on their idea.

And it's like before you submit that to me, you should learn, how do I make these decisions? Talk to people who I have sponsored their efforts. I think it makes the experience much smoother for you to learn from someone else than sending me a design doc and realizing I'm going to read it. And I'm going to give you comments all through it, which I think some people think are harsh. But in reality, I'm actually trying to tell you how to get my sponsorship along the way.

MP: Rita, did you have anything else you wanted to ask before we wrap up?

RITA: I think we've heard some really great insights about what the job of an executive is like, and how leadership at that level works, and also some great advice for folks that are starting out in SRE or looking to advance their careers in different aspects about how to make sure that, yeah, you are getting what you expected once you make that transition.

MP: This has been absolutely wonderful. And thank you so much for your time today. Did you have anything else you wanted to share with our audience before we say goodbye?

SABRINA: No, thanks for having me. Thanks for evolving this field. I do think that there are great engineers in SRE. I think it's important to know that it's not an operational function. At its core, we have operational work we do, but it's really about what engineering are we bringing to our products and what are we doing for our users. I love it.

MP: Well, thank you so much again, Sabrina, for all of your time today. A lot of planning went into making this episode possible. So special thanks to all of those on your team as well. This has been an absolute blast. Thank you.

RITA: Thank you for letting me cohost on these episodes. It's been absolutely a great time getting to know fellow SREs more.

SABRINA: Thank you. Thank you so much for having me and doing this.

MP: And thanks to our audience for listening.

[VOICEOVER] "Prodcast," the Google SRE Production Podcast is hosted by MP English and Rita Lu, and produced by Salim Virji. The podcast is edited by Jordan Greenberg, engineering by Paul Guglielmino and Jordan Greenberg. Javi Beltran composed the musical theme. Special Thanks to Steve McGee and Pamela Vong.

[THEME MUSIC]