Google Prodcast Season Three Episode Thirteen

[JAVI BELTRAN, "TELEBOT"]

STEVE MCGHEE: Welcome to season three of The Prodcast, Google's podcast about site reliability engineering and production software. I'm your host, Steve McGhee.

This season, we're going to focus on designing and building software in SRE. Our guests come from a variety of roles, both inside and outside of Google. Happy listening. And remember, hope is not a strategy.

[JAVI BELTRAN, "TELEBOT"]

STEVE MCGHEE: All right, welcome, everyone, to season three of The Prodcast. I'm your host, Steve McGhee. This season, we're talking about software engineering in SRE and production software.

We have two guests today that are going to talk to us about resilience and robustness and other words that start with R. We have our two friends, Casey Rosenthal and John Allspaw. And of course, I'm also honored to have Jordan. Jordan, say hello, please.

JORDAN GREENBERG: Hello, everybody. It's nice to be here.

STEVE MCGHEE: Cool, and then why don't we have our guests introduce themselves?

JOHN ALLSPAW: My name is John Allspaw. I work at a company called Adaptive Capacity Labs here in Brooklyn, New York. That's probably enough, I think.

STEVE MCGHEE: What is Adaptive Capacity, John? Tell us a little bit more. Come on.

JOHN ALLSPAW: I used to work at a company here in Brooklyn called Etsy. I was there for about seven years. While I was there, as CTO, I did a Master's degree in human factors and system safety.

A little bit of time before that, which kind of dragged me to that program-- it was continually nagging and still does to this day, the belief that the types of work that we do in software engineering, and operations as a whole has much more to do with other historically known, colloquially known, as safety critical domains.

So it's places that there's high consequence, high tempo, aviation, power, control rooms, and nuclear

power plants, air traffic control, medical and clinical environments, all of these things that you think of. If you hear the term human factors, that's what you think of.

And so I reached out to a couple of people that I was reading so much about. And it turns out I said, come look at this. Am I right about this? Am I in the right direction?

And both of them said, nope. You are unfortunately very in the right direction, I'm sorry to say. So let's get to work. And then I worked with them.

And then I left Etsy, and I got to work with the heavies who have founded many, many important things along these lines, including entire fields such as resilience engineering.

STEVE MCGHEE: Nice.

JORDAN GREENBERG: That's awesome.

STEVE MCGHEE: We're going to get into that word, one of the R words, in just a minute. Casey, why don't you introduce yourself?

CASEY ROSENTHAL: Sure, I'll try to be similarly brief with my intro.

[LAUGHTER]

So I'm the founder of a startup called Cirrusly. Cirrus like the cloud. Cirrusly.

STEVE MCGHEE: I get it.

CASEY ROSENTHAL: Yeah? And generally, I'm known as the chaos engineering guy. I built and managed the chaos engineering team at Netflix, evangelized it, started the conferences, wrote the definition, wrote the book on it, stuff like that, and been trying to follow in John's footsteps.

I followed him into the Human Factors and Safety Systems master's program at Lund, but I just can't keep up. Life of an entrepreneur, I keep getting distracted by running companies and writing books and, occasionally, kids and things like that. But yeah, I'm trying to walk the righteous path that John has pioneered in tech.

STEVE MCGHEE: Awesome.

JOHN ALLSPAW: It's a mutual feeling.

STEVE MCGHEE: Cool.

JORDAN GREENBERG: My mom calls me a chaos engineer when I leave stuff all over the floor.

CASEY ROSENTHAL: [LAUGHS]

JORDAN GREENBERG: Is that the same thing?

CASEY ROSENTHAL: Are you doing it on purpose to test the limits of your mother's patience?

JORDAN GREENBERG: Yes.

CASEY ROSENTHAL: Then, yeah.

JOHN ALLSPAW: There you go.

JORDAN GREENBERG: All right.

JOHN ALLSPAW: There you go.

JORDAN GREENBERG: I'll tell her that. She'll love to hear it.

STEVE MCGHEE: So in the past year or so, I've learned a little bit about the role that you guys-- or the realm that you guys inhabit a little bit. There's a group called Learning From Incidents.

There was a conference. Was it last year or the year before? I can't remember. What is time?

And one of the things that has come up that I've noticed is there's a bunch of words that have definitions that are all very close to each other, but they're all very importantly different from each other. And a lot of them start with R, so I'm just going to list them really quick.

We don't have to define them today, but I think it's just fun just to point out. We have resilience. Some people say resiliency. There's robustness. There's rebound. There's reliability. There's probably more than that, and they're all slightly different from each other.

So one thing that's important about this set of words is that sometimes people use them to describe a computer system, and sometimes it's a computer system with the humans that then control that computer system. And as far as all the R words go together, it's that last point that it seems to be the most important to me in my head of making sure you include those humans in the system. And the

phrase, sociotechnical system, has come to be pretty important as well.

So I wonder if you guys can tell us the stories that are necessary for people to understand this at a high level to get into it. Why is this important? Why is it a big deal? Why can't we just talk about bugs and features and things like that?

Do we really need to think about humans too, John? Is it really that important? Casey, do you think so? What do you think?

CASEY ROSENTHAL: Yeah, if you actually care about the outcomes that should matter to the business, that drive customer satisfaction and probably correlate with revenue, then you should care a lot about the humans and the role that they play in the technical systems, because they're the most important part. And generally, focusing on the most important part is a good idea.

JOHN ALLSPAW: I would concur with my friend. I might even go a little bit further and say that the only reason why any companies are successful are because of people.

Sometimes we will do a thought exercise with groups that we-- in conference talks or groups that we work with. And we say, imagine if tomorrow-- thought experiment-- at 10:30 AM, everybody is instructed to stand up, step away from your computer, and do not answer your email. Don't answer a page or an alert. Don't commit any code. Do nothing to this technical stuff.

And then I want you to think how long will the system continue to run exactly the way you hope, expect, and intend. Would it last 24 hours, 36, 72? Would it last a week?

Even those who say it would last a week are pretty hard pressed to say much more than that. That's actually, I think, a bit stunning. It means that people are doing things to keep the system working. So yeah, that's the strong agreement that I've got with Casey.

STEVE MCGHEE: So clearly, what I was asking was tongue in cheek and leading and all these things. But the reason why I ask it is because, also, this isn't what is always focused on, specifically around when stuff breaks or when we're planning the future or whatever.

Some companies, sure, but most companies, they focus on the other thing, not on the people thing. So why is that? Why do you think that that's the case? And is it something we should think about?

CASEY ROSENTHAL: Well, it's the case because it's easy. [LAUGHS]

STEVE MCGHEE: Yeah.

CASEY ROSENTHAL: Well, there's two reasons. One is habit, right? So we've got a hundred-plus years of habit, of thinking of software the way insurance companies taught us to think about manufacturing.

And industry titans. Scientific management, framed up how we think about producing things and where mistakes are made or where outcomes that are undesirable for the company-- how to think about placing blame for those outcomes.

So we've got a lot of history and baggage that we can drop. And then we've got a lot of stuff that's just-- it's easy to track numbers like MTTR, at least in theory. It's easy to track numbers like that, and then point at a number and use that to judge whether or not something's good or bad.

I'll go ahead and jump forward a bit and highlight why this is important. We're recording this about a month after a huge CrowdStrike outage that cost probably billions of dollars and had a catastrophic impact, globally. So this is a very serious outage.

And what we as an industry are facing is an increasing call, particularly here in the US-- they've already made more headway in the EU on this, but particularly we're going to feel this in the US-- to regulate things like uptime. And once regulation comes in to regulate things like uptime, then a cascade of bad things are going to happen.

A, it's not going to improve uptime, because it's asking to measure the wrong thing. But it opens the door to regulating our industry and putting gatekeeping in our industry that won't be to the benefit of innovation or software engineers, and certainly not to-- it'll be a setback to DEI efforts because getting degreed and certified in something is going to favor people who already have privilege in our society.

So our industry is facing an existential crisis that most of us, I don't think, are even cognizant of as the government is asked to get increasingly involved in regulating our industry because of things like the CrowdStrike outage.

JORDAN GREENBERG: Right.

JOHN ALLSPAW: I wanted to add. Paul Reed, I think in 2018, I think, and then 2019, put together a conference called Redeploy. And at the first, the very first one, he said at the beginning-- and this is something that I continue to believe. I'm glad that he said it.

And it fits with what Casey said, which is, our field, our community-- especially the SRE-adjacent sharpened practitioners-- we are enthusiastic more than other domains or more than other communities about real, concrete, messy-detail observations that are bottom up, right?

Frankly, you could make an argument, that's not too far, extreme programming, and agile and DevOps and continuous deployment-- These were not things that came from somebody who doesn't have close contact with the real messy details. It was starting this enthusiasm. The fact that they're learning from incidents, Slack had grown. The fact that there are more people doing their Master's degree.

The fact that Casey even begun the program is notable. And what Paul said is this momentum-- I'm paraphrasing —ut this momentum is important because if we don't have a chance at helping set the agenda for what genuinely productive, humane, and just pragmatically good the future will be, then someone will do it for us, and we will not likely be happy about the results.

JORDAN GREENBERG: Very good point. Switching gears a little, I've heard that people avoid declaring an incident because it means more paperwork. You have the incident itself, the action items, writing the postmortem and the retrospective, reviewing the postmortem and retrospective. And sometimes this means getting named and shamed for something that you were responsible for. Nobody likes that.

JOHN ALLSPAW: No.

JORDAN GREENBERG: Do either of you have any advice on how to approach retrospective reviews of postmortems after an incident?

JOHN ALLSPAW: I do.

JORDAN GREENBERG: Go for it.

JOHN ALLSPAW: It's a great question with a number of potential hypotheses. Jordan, I would say that there are many reasons-- we've certainly seen many-- amongst those that you counted out for reluctance to declare an incident.

Now, of course, it's a label that is negotiable. It's always negotiable. And there are lots of reasons why people-- even outside of those-- yes, of course, if I call this an incident, jeez, man, that means I'm going to have to do all this stuff. And sometimes it's just a hassle.

But it's only a hassle because experience has told them that it doesn't matter. It doesn't do anything.

I'm going to go to a meeting that people won't pay attention to. I'm going to write up a document that nobody's going to read.

And so there's a cycle that's based on this expectation. In that situation, it's just a hassle. In other situations, it is another thing that you said, Jordan, which is it's a hassle, and I may open myself up for unwanted attention.

JORDAN GREENBERG: He would say that.

JOHN ALLSPAW: Yeah. Yeah, because for multiple reasons. Because if there's an incident, and something's broken, that draws attention from the people who are super far, a.k.a. leadership.

And then because it has that label, they're really concerned. But if we don't call it that, the same reality can play out. But they won't have that much, right? So I'm incentivized to not do it.

The third reason, which I see more often, is neither of those dynamics. We've seen plenty of organizations-- not plenty. We've seen a number of organizations. We spoke at the first LFI conference. I gave a talk with one of our clients, and this is one of them.

They're not afraid of doing the paperwork, the bookkeeping, because, actually, they've seen that it is a huge-- people are psyched for it. People will cancel other meetings so they can go to the group review meeting.

They look forward to it. They also aren't afraid that anybody is going to yell at them or there's going to be table pounding. The reason why they don't is because they're not actually sure at the time whether or not it actually is an incident.

Incidents don't show up with a big sticker on its forehead and says, I'm an incident. Start paying attention, right? Partial, weirdly degrading, but only a little bit can be an incident.

The reason why people will demonstrate this-- if you've ever had an incident or experienced an incident, and afterwards you realize that it started way before, then you noticed it.

JORDAN GREENBERG: Yes.

JOHN ALLSPAW: Right? So declaring an incident in that situation is only-- sometimes you did know it, and you say to your cousin, hey, this looks weird, right? Not weird enough like, oh, my god, but this looks weird?

And they're like, yeah. Huh. Is it something? I don't know. See what happens. Maybe it'll balance out. Maybe it'll clear itself out. All right, we're going to go to lunch. We'll come back.

Some time will pass. And then they'll say it's not looking better, and it looks like it might be getting worse. Now let's decide. We're going to call this a thing now for some reason. That's a judgment, a judgment call.

Hindsight will only be the one that tells us the outcome. So it can be hard. Some of the most ambiguous parts of responding to an incident is even working out whether it's an incident or not. Also, same thing, sometimes it can never be clear when it's over.

JORDAN GREENBERG: True.

STEVE MCGHEE: Yeah, one question I ask customers who are confused by this is, do you use severity levels, and do they ever change? And why do you change them? And when do you change them? And who changes them?

It's all over the map, the reasons and who does it and why. The severity, to me, is just madness, right? It could be anything. And to me, that tells me that this is not a science right now.

This is pure vibes, which is fine. That's where we are. But we don't want to delude ourselves into thinking that it's pure science.

CASEY ROSENTHAL: The part of that I don't think is fine is that it's treated like science.

STEVE MCGHEE: Yeah, yeah, good point, good point, for sure.

CASEY ROSENTHAL: And that does get us in trouble because MTTR is a similar thing. MTTR, I'll just go out on a limb and say, if you're tracking that, at best, you're wasting your time, because MTTR is complete bullshit. I'm not sure I can--

JOHN ALLSPAW: No, no, it's Google, so you say horseshit.

STEVE MCGHEE: Horseshit, OK, thank you. Thanks for the clarification, John.

JOHN ALLSPAW: No problem.

CASEY ROSENTHAL: And let me just back that up by saying we have the math. We have the studies.

We have the scholarship. We have proof that MTTR is horseshit. So if you'd like that proof, we can give you access to that. The proof exists. It's a waste of time.

So when companies spend time and attention focusing on bad things or wrong things like MTTR or severity-- severity is another one-- you're exerting effort in a way that can only lead to bad outcomes because businesses that are poorly aligned don't function well, right?

So when you try to align around something that's false or misleading, then you can't highly align. And that causes resentment, and then people start not liking things. So then you've got a business that isn't functioning well, and people are miserable there while they're doing it.

So I think a lot of the negative attention around incident management comes out of that. A lot of people intuitively know that some of the processes that they're following are irrelevant or inconsequential or the wrong thing to be doing for the better sake of the company.

But they do it because it's process, because it's mandated by leadership. And that's part of the problem. If we don't address that and change that, change how the industry thinks about those things, then we will continue to have those negative spirals and poor outcomes and conversations that don't help but rather hinder the industry moving forward, like the majority of the conversations that came out of the CrowdStrike incident that were just not helpful.

JOHN ALLSPAW: Yeah. I have two very small things to add to what Casey said. I like to think sometimes as severity as a GROUPBY for everything else, right?

It's like, oh, MTTR. The other thing-- the most cogent, eloquent take on, in particular, severity, M group at SREcon this past Spring gave a talk with one of the best titles ever, which is What is incident severity but a lie agreed upon?

And the take that they had is absolutely spot on. It may have been the best talk I saw. It certainly tied with the best talk I saw. So if you haven't seen it, it's very good, and it's just all laid out.

STEVE MCGHEE: Yeah, that really landed it for me, as well. I remember seeing the same talk. In fact, I think this is the second time it's been mentioned on this podcast, so that's two votes for M. Way to go.

JORDAN GREENBERG: I think that's true, yeah.

JOHN ALLSPAW: So if you haven't seen it, listeners--

JORDAN GREENBERG: Now's the time.

JOHN ALLSPAW: --you're behind.

STEVE MCGHEE: That's right. OK, so we can't MTTR. Got it. We can't just blame the people for not trying hard enough. Got it. We can hope that people will take the time to decide, oh, yeah, that was a thing, and maybe we should write some things down about it. Maybe, right?

JORDAN GREENBERG: Maybe.

STEVE MCGHEE: Postmortems are a thing that we talk about in SRE, and sometimes we review them and try to learn from them. But what comes out of that a lot tends to be, fix this bug. Do this thing. Add this monitoring. It's a bunch of to-do items.

And I've heard from a lot of people that we actually want to focus a little bit more on learning about our system, as opposed to doing things further to our system. But those seem harder to count, right? Those seem harder for spreadsheet wizards to agree that it's a good thing about.

What I'm getting at is, how do we push people towards the activities that actually make a difference, as opposed to those that are easily accountable, if you know what I mean?

JORDAN GREENBERG: Absolutely. As a spreadsheet wizard who ran the postmortem events, I have never seen an action item given to SREs that said, keep watching this and see what affects the system over time.

I've never seen, capture if this is an isolated incident or if this is something that has been ongoing and will continue to be ongoing. I've not seen, fix the thing, and then capture it, and then report back later. I've only seen, here are your action items. When these are done, you can close this postmortem report and carry on with your day.

CASEY ROSENTHAL: Yeah, that's ripe for the government to step in and say, you know what? We asked the Fortune 50 companies what their best practices are for action items, and here they are. So now you have to do these same action items.

So let's take a completely different perspective and say, have you ever come out of an incident review with the understanding that we need to figure out if customers cared that this thing happened.

JORDAN GREENBERG: Huge, huge.

CASEY ROSENTHAL: Did this impact customers? Or here's another one. The outcome that we all think needs to happen is the CTO needs to allocate more resources to this. That's not an action item.

JORDAN GREENBERG: True.

CASEY ROSENTHAL: Right, but compare that to an action item. Which one is more likely to result in a system that's more reliable? Telling an engineer who's working on the thing, oh, yes, if you monitor this thing, then it's less likely? Or getting the CTO to invest more resources?

I think we all agree just off the cuff that getting the CTO to invest more resources is immeasurably more impactful to the reliability of the system. And that's never going to come from an action item list or-- I don't know-- spreadsheet wizardry.

JOHN ALLSPAW: Yeah, I completely agree, unsurprisingly. There's a lot of topics. And Steve, this question is load-bearing in multiple ways.

CASEY ROSENTHAL: I'm so sorry.

JOHN ALLSPAW: It's fine. It's fine. I'm going to try to keep my thoughts about this topic-- sorry, topics-- coherent. I don't want to be all over the place.

Lots of organizations-- and again, this is mostly in the realm of conventional and typical, which is really the only time I'll be abstract. What we see often is that, along with counting things-- I'll get to that in a second. I don't think counting things is inherently bad. Counting the wrong things turns out to be terrible, which I think we've already covered.

As a practice, sometimes, the more severe the incident or more impactful, more significant-- let's just say that-- the less time people are given to produce an explanation and a description of the event, right? Or sometimes they'll say you need to have-- whether it's a norm or a mandate or a policy, whatever you want to call it-- within n number of hours, you have to have some description.

Some of that, I completely understand, are dictated by or led by contractual obligations with customers. But those are different communications and different activities than helping people inside the organization, right?

It is ironic, or it's reversed, inversely proportional. Though, if it's a significant incident, a surprising incident, it was probably because-- part of it is because it was really hard to figure out. It was really

hard to figure out what to do about it once you figured it out. And it was hard to figure out whether or not that worked.

So in other words, the most complicated, difficult, uncertain, ambiguous event gets the least amount of time to describe, which is, I think, backwards. As a result of that, people, they'll put in what they think. It's a bit of a cynical take, but they will put in items in a template sometimes. Templates are really good for giving you an excuse for not to be curious.

So as a result-- and we only booked this conference room for an hour. How many times did we say, well, listen, everybody, listen. It's great that we're talking about this. And we're blameless, by the way, thanks, super blameless. I appreciate that.

But listen, we've got five minutes here, and I only see three-- let's come up with-- I don't want to come out of here without five. Is there something that we did already that I can put down?

It's more of a demonstration that you're doing something. And so if you've had any experience as a software engineer, then you have had the experience where you're banging your head, you're really having a tough time with this particular bug, and you're exhausted, right?

You've already tried the afternoon coffee trick, and now you're wired, and you're scattered. It's like, screw this. I'm just going to put this down. And it's in the shower the next morning that you realize what the thing was, right?

So don't be surprised if, during some mandated short period of time, you come up with ideas for action items that, given some time and space to reflect on it, you realize is a terrible idea, right? You want to force people to go ahead and do them when they know that there's a good chance that, if you were to do it, it might make things worse. Or it might fix that thing and introduce new things, a new vulnerability.

So yeah, that's why you see Jira tickets that say it won't fix. Thank God there's that field you can drop down.

JOHN ALLSPAW: Because it's not a good idea. That phenomenon, by the way, where you're going to intend, whether it's an action item or not, some post-incident remediation item that turns into yet a new set of latent conditions or enablers for a different type of thing to surprise you-- colloquially, we've attributed that to Lorin Hochstein. I think someone had coined that Lorin's Law. And now that

you have a term for it, you're going to see it.

JORDAN GREENBERG: I see. OK, so if postmortems are not the way to turn an incident into more headcount for your team, what should we want to get out of them? What are some things that have genuinely come from postmortem reviews or post-incident system monitoring that have impacted reliability positively in your sphere?

STEVE MCGHEE: What does work, John? We've said all the things don't work. What's left?

JORDAN GREENBERG: What does work?

CASEY ROSENTHAL: Yeah, so this is the most exciting part of this because it's blue skies. We can give you a couple of ways that have been positive and fruitful things to do. Chaos engineering is a whole field that improves reliability and understanding of complex systems. You can put a number on anything, so it's very easy to go from qualitative to quantitative measurements.

But I generally suggest people don't start with the quantitative stuff, because you're probably measuring the wrong thing first. Start by measuring qualitative things. And there's a bajillion frameworks to measure things qualitatively, and we can propose a couple.

But there's no limit on the number of systems that you could use to measure things qualitatively and understand a system in qualitative description that would get around your problems with, for example, severity or what counts as downtime, these descriptors, MTTR, things that just don't work. So it's a very young seed for a field that we can open up and define if we choose to pioneer this field.

STEVE MCGHEE: Without putting you on the spot, Casey, can you go into that a little bit further in terms of-- can you provide a made-up example, even, of starting with a qualitative scenario or read on a subject and then describing some quantitative outcome?

CASEY ROSENTHAL: Yeah, sure. So we would have system degradations-- So when I was at Netflix, there was always some part of the system that was degraded, take that for granted just at the scale that it's at. There's always some part.

So declaring an incident started at time x and ended at time y. It's just not helpful, regardless. John gave me the great example of incidents that have a negative resolution time because by the time you noticed it, it had already resolved itself. [LAUGHS] So those numbers don't help.

One of the scales that I've brought over from business instructional settings is the Kirkpatrick method, which just has four tiers of ways of qualitatively measuring the outcome of a program.

Now, I took it from an educational background, because it's describing learning, right? So it's saying, we applied, in this case, a training, and then we want to, OK, at level 1, did the people who got the training think it was useful, right? It's just asking them.

And in the case of an incident, just ask the people who are part of it, hey, did you learn something from this review, right? Or we introduced chaos engineering tools. To the people using the tools, is the tool useful to you. And then going up the stack, OK, now you ask the people that they report to, hey, are there things that you're responsible for that you think this is actually contributing to the success of your team?

And then you go higher, and the fourth level is-- at that level, do you have some evidence that introducing this practice or conversation or tool or whatever impacted something that the business cares about?

And that's not instrumentation. That's not necessarily measuring something. That can be a, quote, unquote, "gut feeling," but it can be somebody's context and their understanding of their world in their given context, which, in this case, if you're hiring an executive to manage cloud operations or whatever, you're paying for their expertise to understand their context.

So you don't have to put a number on it. Just ask them. [LAUGHS] Is this thing improving how your organization benefits the business? And that's a completely different question, and so much more interesting to get that answer than to say, were we out of SLO? Or how many high severity incidents did we have this quarter because I want to tsk-tsk at somebody if it was higher than last quarter.

That's useless. Anybody who operates systems at scale knows that has no reflection on the success of the system or the effort that was put in to make the system perform the way the business expected it to. So just don't look at those things. Instead, go to the people.

JOHN ALLSPAW: Yeah, I have an answer that's complementary. I actually do have-- we're in the middle of writing a book, and I can't believe I just mentioned that, because now it means that we'll have to finish it.

STEVE MCGHEE: Gotcha.

JOHN ALLSPAW: Right, and it's been going on for years.

Think about when you're not having an incident. How did that happen? You're making changes on every dimension, statistically, mathematically, conceptually. Organizations that we're talking about are astronomically more successful than they are down, right? It's an industry where we talk about it in the number of decimal places after 99%.

JORDAN GREENBERG: Statistically significant.

JOHN ALLSPAW: Right. What goes into making that successful? People are putting stuff in like timeouts and retries. And this thing, we should have some redundancy for. This thing over here. How do they know that? Experience is a cop out answer. Expertise is the answer.

And they understand that because either, one, they have first hand experience of what it looks like when they don't do that stuff. And number two, they have heard, they have benefited from an experience that somebody else has had. This is known as vicarious learning. And so the way that systems improve is by creating, number one-- I'll start with this one.

Number one is exactly what Casey said, which is let go of the possibility that you're going to be able to measure this in some sort of way, like humidity and temperature and pressure and that sort of thing.

Once you let that go-- because here's the thing. You're already doing that. There are many expensive activities that organizations are engaged in that never get economic scrutiny, that never get any sort of-- they don't even show up in a spreadsheet. Is there a spreadsheet by how many hours engineers engage in code review?

JORDAN GREENBERG: Good point.

JOHN ALLSPAW: I don't know of any group. I'm sure there might be. But I think we can all agree it is easily one of the most expensive activities in a software engineering organization, right?

When I'm talking with engineers, we usually use the example of brand redesign, which is incredibly expensive and incredibly difficult, if not impossible, to measure. So what makes-- if you can-- I'm trying to impress Casey by not saying a couple of really academic words.

Instead of looking at incidents as these disconnected, disembodied distractions in a category called cost of doing business and looking to reduce those negatives, how about figure out-- in spite of

whatever policies and norms you have, people are way more successful.

What goes into that? Increase that. Find out what that makes that work. Increase success by understanding it closely, rather than the other. Because with all of the shallow metrics stuff-- my colleague never really liked this framing, but the denominator is missing, right? Oh, you've had six incidents this week. Out of how many?

STEVE MCGHEE: Yeah, we're measuring all the red stuff, but we don't measure the green stuff very well, do we? Even though there's way more green than there is red.

JOHN ALLSPAW: Way more, way more, yeah.

STEVE MCGHEE: Nines, right?

JORDAN GREENBERG: Mm-hm.

STEVE MCGHEE: We've taken up way too much of your time, and we have one closing question, which you can either take seriously or not. It's up to you entirely. Which is, in the future, in the distant future, my personal belief is that more people are going to depend on big computer systems than they do today, for good or ill.

It's just going to keep getting bigger and stuff. And this is even life-saving things or really important things. Does this freak you out, or are you optimistic about this? How should we consider the future when it comes to society depending on computers?

JOHN ALLSPAW: So I've got an answer that's succinct, I believe. And it's something that my colleague Richard Cook and I wrote. It's his words, because you can tell-- you've known Richard-- his vocabulary was quite exceptional.

It's a chapter in the Seeking SRE book, and the title is SRE Cognitive Work. The introduction goes into saying, things become more complex as they become more successful, which fits to the basis of your question, Steve. He says we're involved in exploring how this is kept running and handled when things are not going well.

What we find is both inspiring and worrisome. It's inspiring because the studies reveal highly refined expertise in people and groups, along with novel mechanisms for bringing that expertise to bear. It's worrisome because the technology and organizations are often so poorly configured to make this

expertise effective.

The beginning of the chapter that we put a quote, and that I think-- actually, there are scientific papers written about how the work that this quote has come from, which was written in '83, has only become more relevant. And Ms. Ann Bainbridge points out that there is an irony that the more advanced an automated system is, so more crucial must be the contribution of the human operator. Machines aren't responsible for anything, but people are.

JORDAN GREENBERG: Absolutely, that's a great summary.

JOHN ALLSPAW: And there's an opportunity. I remain optimistic.

STEVE MCGHEE: Yeah, this is the Ironies of Automation paper. I think it's oft cited.

JORDAN GREENBERG: Well, thank you both to our guests, John and Casey. You've had some really awesome notes for us. I think we can summarize by saying MTTR bad, and the human factor of how we use our tools to end up with something and be responsible for something is most important here. Thanks again for showing up and talking with us today, and have a great day, everybody.

CASEY ROSENTHAL: Thanks for having us.

JORDAN GREENBERG: Bye.

JOHN ALLSPAW: See you all.

[JAVI BELTRAN, "TELEBOT"]

JORDAN GREENBERG: You've been listening to Podcast, Google's podcast on site reliability engineering.

Visit us on the web at sre.google, where you can find papers, workshops, videos, and more about SRE. This season's host is Steve McGee with contributions from Jordan Greenberg and Florian Rathgeber. The podcast is produced by Paul Guglielmino, Sunny Hsiao, and Salim Virji.

The podcast theme is “Telebot” by Javi Beltran.

Special thanks to MP English and Jenn Petoff.