CR071

Bridging the cloud infrastructure talent gap with software with Cory O'Daniel, Massdriver

Capgemini

CR071

Bridging the cloud infrastructure talent gap with software with Cory O'Daniel, Massdriver

Disclaimer: Please be aware that this transcript from the Cloud Realities podcast has been automatically generated, so errors may occur.

[00:00:00] The meat is the same size. It's like this much rind. So they're very good. They're great for cocktails. Like you can just grind and grind and grind and grind and grind. You have just tons of rind. But it's funny the first time I saw when I was like, Oh dude, we're making lemonade. One of these. And I cracked it open and I was like, this is the dumbest thing that nature's ever made.

Unless you like cocktails.

Welcome to Cloud Realities, a conversation show exploring the practical and exciting alternate realities that can be unleashed through cloud driven transformation. I'm David Chapman and I'm Rob Kernahan and today we are going to deep dive into platform engineering. What it actually is, what the nirvana of it is, but what also the reality is and how you take Baby steps initially and then maturation steps to [00:01:00] get to something that that feels and looks like a platform But before we get to that Now Rob over the course of the last few episodes.

We've been doing Interesting stories about how you get yourself into befuddled situations, but I thought today you might want to tell one yourself What's wrong with your face? What's wrong with your face? David, let's start. I do have a fat lip today, David. How did that happen, Rob? It was via misadventure, where I might have been doomscrolling.

Uh, might have lost track of time, fallen asleep and my phone fell and hit me in the face. So yeah, lesson learned, uh, that it does actually, uh, it's, it's getting better, but yes, that's a personal injury, uh, through my own incompetence. Let's hope it gets better, not worse, because it's quite a, it's quite a swelling you've got going on there.[00:02:00]

Let's uh, fingers crossed it improves over the weekend. Fingers, fingers crossed. Now, what else has been confusing you this week? So, the one I've been pondering is uh, social media and the broadcast and response style social media. Has it reached its zenith? And the reason I think this or I'm, I'm a bit confused about it is social media is basically getting mold at the moment.

So we have these very large open platforms online. People are starting to rage against the terms and conditions. Yeah, they're starting to think about what's happening with my data on how it's being used. We have this angst and effect on society where you know it's starting to Harm us. There's social media platform addiction going on.

, you get the whole raging against the tick tock platform from, uh, you know, the U. S. Starting to legislate to ban it. And the whole thing seems to be coming in on it. And Mr. Musk at X has driven a very different Type of approach. I just wonder if it's peaked and then it's going to get mold [00:03:00] and then it's going to be down on the on the downhill and something else will come out of it.

And like closed community social media and things like that which you get works quite well and we appreciate that it's not as toxic or is it just we're addicted to it and it's going to romp away and it's just going to get bigger worse and just more impact and it's going to take a bigger part of our lives and I can't, I can't quite work it out because I can see both scenarios occurring and I'm a bit confused by it, Dave.

Well, so there is a, I'm going to get my numbers and facts completely wrong here, but the just a little. No, exactly. Let's not change the habit of a lifetime. But the gist of it, I think, is right, which is it was something like that. The human brain can only deal with sort of so many social interactions and people interactions.

And, you know, when we lived in sort of villages and small towns, That was scaling quite

happily. Thanks very much because it was scaling at a geographic person. It was scaling very [00:04:00] straightforwardly. Social media, to my mind, Is so profoundly beyond that, that I actually, it's almost like an evolutionary step for us.

Like we, we fundamentally can't deal with it. I have no idea how other than having social media teams, how people who have got many hundreds of thousands or millions of followers could possibly deal with that. Like that becomes an industry and a business in its own right, of course. Uh, it just seems completely mind boggling to me that individuals can wield that kind of influence and power in the world.

So it's your view. This is it. The decline has started the systems catching up with it, or do you think it's just going to keep rolling? So despite that point, I do, I think my point in there is human beings do struggle with this. And I think there is like, there's a real struggle with it in terms of like, What positive interaction looks like, what meaningful interaction looks like, all of those sorts of things.

That being said, I think the other dimension that you touched on in your [00:05:00] intro there was like X in particular and the opening, the very visible and transparent opening up of X to sort of all kinds of different dialogue and whether that needs to be mediated or edited and have a particular point of view and a particular thing.

And, and Musk is very clear on that. He's a free speech. Advocate make of that what you will I'm not saying that's good bad indifferent It just is what it is And that's how it's playing out and that feels different and it feels a bit edgy and it does feel toxic in places and it Feels difficult, but there's still also the same people if they haven't run for the hills and gone to one of the other platforms They're still in the mix there somewhere So there's that and I think that is going to be interesting to watch that play out You know media and reputage in this day and age has been so damaged by all of this I do think it's I think it's problematic and I don't know where that's going to, I don't know where that's going to end.

That's definitely a confusion. But then also I'm planning a trip to Japan and I'm finding Insta [00:06:00] really, really useful because I'm, you know, getting some decent intro in, you know, info about like some small little ramen places to go to in Osaka and, you know, like which temples to go and see and what time of the day to go.

So I think it is a legit confusion. I don't see it going anywhere anytime soon, Rob, I'm sorry to say. So, but do you see it, but you see an evolution of it, maybe? It's constantly evolving, isn't it? But maybe more dramatically when society gets a bit of a Yeah, I don't There's a lot of angst about it at the moment, that's the bit that I sort of struggle with about the, what does the angst drive, or do we just get used to it and move along now?

I think it'll level back out again and then move forward and level back out, you know, it's like one of those curves that's constantly going up but moving up and down at the same time. But, you know, like, I'm on it like a lot. You know, it's, it's a, a main source of media and information for me. So like, I, I can't see it going anywhere.

I think it is the replace, I think it is the replacement. I think it, it, it's given me a fat lip now, so I'm gonna chuck it outta the window. I'm not ruling out Yeah. That it was [00:07:00] AI did that to you, that they, that they're like, they did like one of those little buzzes and it just caused it to fall outta your hand.

They, they've been keeping track of what you've been saying. Yeah, probably. Anyway, look, let's get onto the main part of today's show. And I am delighted to say that Cory O'Daniel the chief exec and co founder at MassDriver is joining us to talk about platform engineering.

Corey, brilliant to see you. How are you doing?

And do you want to just say a word about yourself and MassDriver? Yeah, I'm doing great. Thanks for having me on the show. Uh, yeah. I'm CoryO'Daniel, CEO and co-founder of Mass Driver, the e changes, sometimes it's executive, sometimes I'm the chief elixir engineer. We write an elixir, so, sometimes I'm the Chief Excel officer.

Uh, I'm not very good at Excel, but I'll take it. You got to keep those ees fluid. You got to keep, keep those ease fluid. Sorry. A lot of ease. Yeah. So mass driver's a visual tool for managing infrastructures code. The idea is that. Your operations engineers can design Terraform modules, Ansible playbooks, Helm charts, et [00:08:00] cetera, any IAC tool.

Put all of your policy code in place and then publish it to our registry. And then as your engineers need cloud infrastructure, instead of them having to learn Terraform or click around in the cloud console, they can essentially diagram and as they diagram their infrastructure, it's using your IEC modules and policy as code to do all the provisioning.

So you can give them the feel that they have full control over the cloud, but you can make sure that all of your policies, configuration and best practices are in place. So let's explore this from a point of view then of the software development capability and the relative scarcity of it still, you know, the industry has moved forward very rapidly over the last 10 years.

University pipelines and workforces are sort of struggling to sort of keep a pace with. The radical technology change. Corey, perhaps to sort of open up the conversation. Have you got a perspective on that from what you guys are doing? Like the pace of change and whether workforces, human workforces are keeping up with it?

Yeah. [00:09:00] So what's really interesting is when you look at, you know, the stack overflow surveys from the past few years, like consistently, the number of operators has Gone down like sysadmins, DBAs, people with cloud experience, like that number is just continually going down, but it's a ratio, right? It's relative to the total number of developers.

And what's happening is we're making just tons of developers out of boot camps. We're making them faster than we've ever made them. And the people that have experience operating in the cloud just isn't keeping pace with it. Right? So you're learning about, you know, building react, you're building rails or something in a boot camp in 12 weeks, and then you come into a job like you can build features.

You're doing the thing that you know the CEO needs you to do. But then when it comes to like operating in the cloud, you see a lot of people kind of guessing their way through it because you don't have a big operations team. And that's one of the big gaps that we're kind of seeing in the workforce. And was, was the observation of this gap part of the sort of founding purpose of MassDriver?

Well, just maybe take us through your thought process and sort of how you got to the tool [00:10:00] set that we'll come on to talk about. Yeah, so when, when I first started in operations, um, my, my background was working in healthcare. And so I worked in data centers. And then when I first moved to California, Became a software developer.

And so my, my very first role in 2005 2006 was migrating a data center to easy to like when it first launched. And so I kind of got pigeonholed into like what we start to call like the DevOps role, right? Versus like, doing DevOps, like I am a DevOps, right? And so, you know, decade passes of me working at startups and kind of seeing this where it's like people, you know, they're familiar with writing software.

They took software development courses in college, but like nobody's teaching them the cloud, right? And then as I kind of progressed through my career, I started doing professional services. So I was, at the time, I was consulting with Google, doing a large-scale migration. I was moving a large customer of Google's to Google Cloud.

And this is a multi-billion dollar company. They've been in a data center for, I mean, they've been in a ton of data [00:11:00] centers for over a decade. And you see the same thing, right? You see this team of like 50 to 60 operations engineers that know the data center. They know switches. They know networking. They know Linux administration.

But now they're a cloud company. Right. And they don't, they don't know terraform. They don't understand the cloud. They're not traditional software developers. Like they're all of a sudden in this place. And, you know, as I'm working with this company, I kind of realized like this problem is pervasive across the industry.

Like we have people that know how to write software and know how to work in data centers, but the, the overlap of people that have, you know, This DevOps experience, cloud management experience that understand systems and understand like software and automation is a very small percentage of overall developers.

And so that's kind of where the idea for MassDriver originally kicked in was, can we make it easy for developers to manage the cloud and make it very easy for your smaller teams of operations engineers to kind of scale themselves? There was a point in that where, [00:12:00] you know, there's a few things in there where you got the unicorn effect.

It's. As people move to cloud where you got some a very small number of people who could do it full stack, but they're so rare then you saw the job description come out for people they couldn't find and then it pushed a focal point onto a small number of people became a massive bottleneck and it was a massive source of frustration as well.

And it's that the irony is, though, the discipline of codifying clouds is a developer style trait, which is you are codifying a pattern to achieve an outcome is very different to data center ops. But actually, it's the developer discipline creeping into the computer arena. Exactly. And then like the thing that's.

You know, difficult on top of that is like, there's one thing which is like thinking through the development principles, right? I'm writing software, I need to build good patterns, I need to build good interfaces for my customers, the developers, right? But then there's, I got to learn the tooling, I got to learn the CI, CD.

I've done all [00:13:00] that. That doesn't mean I understand the cloud service whatsoever, like there's just a massive hurdle of work and then it's like, okay, I have to go learn and understand how to manage, operate, secure, observe, monitor, backup, all of these things that all of our developers using, whether it's, you know, Postgres, right?

Like you all of a sudden have a ton of technology that you have to understand on this very, very small team that is, you know, you. Core to the entire business functioning now that you're on the cloud, right? And so like, how do you scale, you know, maybe the 1 to 10 ops person to developer that you have in your organization when they're constantly underwater, right?

They're backed up, they're drowning in debt, but they're trying to build new services and build new automations for engineers like that is a, that is a tough thing that a lot of companies struggle with. In your mind, what we've just been talking through fit with the concept of platform engineering. Is it like the totality of platform engineering in your mind?

Or is there [00:14:00] additional aspects of platform engineering that, that the tool set you've just been describing and the problem set you've just been describing, is that a subset of platform engineering? I think getting there, it's one of those things that's a, I mean, we believe in platform engineering like I think it is very important and I think it's going to be necessary to get a base level of platform engineering, whether it's You understanding the discipline and doing it or really good tools to kind of elevate the team that you have, I think it's important.

But getting there like saying like, Oh, I'm a startup or I'm an early stage company or I'm a small team. Like, do I need to do platform engineering and have a platform team? Like that's kind of a that's a reach for a lot of organizations like there's not a ton of those people, right? That's a great point.

Maybe let's just double click that for a sec before we move on. Sorry to interrupt. Yeah, no problem. Maybe just a basic definition. So like what in your mind is. Is platform engineering and why is it a reach for some of those organizations? Yeah, I mean, I think at the most basic, it's taking those DevOps principles [00:15:00] and turning that turning it into a product within your organization, right?

Like actually building a real good center of excellence, like actually doing the DevOps work, but then taking that next step to start to automate your automation. It's not me doing Automation for one off task. It's me starting to take the concepts of this business, the things that we need to operate as a team and starting to build abstractions that engineers can use that make that a bit self service, right?

So if I'm an engineer and I need a new database, like, let's say Let's say we've standardized on Postgres. We use Postgres. Everybody knows how Postgres works. But we grabbed an off the shelf open source tool that runs on MySQL. Should the entire, should that engineer that is going to be running some specific tool that runs on MySQL have to stop and learn everything about operating MySQL?

I'd say no. Like, that's, like, that's not a great use of their time. They might need to use that service. That's a dependency of that service, like, requiring the overhead of me going and figuring out how to run MySQL RDS, right, as the engineer, and also running this tool, like, that's a lot of heartache [00:16:00] just for me to be able to integrate with that tool, right?

And so, if I can make those abstractions as the operations team, they make it very easy for that person to get it, right? Like, if you think about what we do locally, let's say I add a new tool to my stack locally, I, I go, oh, it needs MySQL. I brew install MySQL, I'm done. Right? Like platform engineering locally is fantastic, right?

Oh, I need, I need this construct. I need this abstraction brew install or a docker run and it's done. But when you get to the cloud, there's like 35 hurdles before like the thing's going or you can get the thing going. And is it secure? Is it compliant? Like, who knows? Right? Like, but it works. So the PM's happy.

I'm happy. I've moved on. Breach in six months. Right? And so like, can we get a base level, Okay. Of functionality available in our organizations just focused around the stuff that we need so they can move quickly, right? So it's like, okay, my sequel comes onto onto the road map. The ops team's like, Okay, we got to figure out how to do this, but we've already got good abstractions built around how to do I am based authentication and databases.

We've [00:17:00] got good abstractions around like how to add compliance and security modules to like scan things, right? We've got a good baseline as the operations team. We've

done a lot to automate it. Right. the tooling and stuff that we use. And now it's just okay, let's figure out like the best practices for monitoring and scaling and backing up my sequel, get that shipped and ship it.

Not as like I'm making you your database and you're happy developer. We have a construct now for deploying my sequel. So if anything comes along and we need my sequel, it's just as simple as like adding that module. Right. And so I think what's really important for platform engineering is we think of it a lot.

And I think in Google's terms, And in Spotify backstage terms, those are two companies that are operating at a level that I'm sorry, most people aren't operating at right. And so I think that as operations engineers and DevOps engineers looking at that and thinking, how do how does my team of three ops people?

Get to Google level. It's like you're not. And we have to bring it down just a little bit for them. And it's like infrastructure level platform engineering [00:18:00] that makes it easier for your developers to feel like they have control of the cloud, like they have control of the services with Docker run or brew install.

Right? That I think is the promised land for most organizations is getting that DevOps team to be able to be shipping components that are very reusable, makes it easy for engineers to self serve, whether it's a UI like mass driver or whether it's a GitHub repo. Yeah. Right, like it doesn't have to be some amazing piece of software like you can do this on fumes as an operations team and they what I've observed that be interesting to see if you you have seen the same thing is people go to clouds, they embrace the sort of new way and they go, we're going to be full stack and then they get this as we described the world in the front where they just configure all themselves, they push it out the door, they're just Focused on the functionality.

And then they live in that world. I call it life after CICD, whereas you actually went live and then you had to live with this behemoth monster, which is a new type of tech debt. And then you see the emerging thinking of they've had to deal with it. And then the platform thinking rises [00:19:00] and they understand that they need the power of the platform.

So, and I see that a lot with organizations that think we'll do it all ourselves, it'll be fine. And then about two years later, they go, Oh, Yeah, we really should have built a platform first to allow all this stuff to work greatly because they have the security breach, or they have the toil, the cost of maintaining 50, 60 different things in cloud.

Have you seen that as well, that curve of that learning where the realization of the power of the platform arrives in the mind's eye as they've just been in the quagmire for so long? I have, and I've seen it like kind of surface in a few different ways, right? Like, it's funny, like thinking about the f full stack, like, and how full stack has changed.

Like, if you were a developer 20 years ago, full stack meant something very different. You were LAMP, right? And you're like, oh, I got, I got a VM from Slicehost that's running Linux. I put Apache on it, right? Like, there was very little in your full stack, and you're probably writing PHP that generated JavaScript, right?

Like, that was full stack years ago. Now, full stack, it's funny, like, when you see it, like, you know, there's frameworks [00:20:00] where it's like, you can do full stack and it does everything, but like, You know, you see some of these roles out there where it's like looking for a full stack developer and it's like experience.

It's like React, Next. js, Vercel, EC2, Redshift. And it's just like, Oh my God, like the amount

like you've got a little data scientist in there. Like you've got a little like data engineer in there. You got front end engineer in there. And like the reality is like, Software is one of the only places where we will just like stack responsibilities on top of people.

We don't do this with law, like your accountant and your person defending you in court, they both study law, but it's not the same person. But like, we're like, oh, the person's got a keyboard. We'll just throw more responsibility on them. He's in your discipline, he's in your discipline, he's in your discipline, he's in your discipline.

Keep poking them. And it's funny because like we kind of ask questions. As developers, we're like, Oh, this is our hobby. We love to learn. But then we get burned out. We're like, Oh, shit, we're burned out. Like, we got too much stuff going on. It's like, I don't know. We need to just like say no at some point in time.

But, you know, as far as like, you know, people kind of getting to the point where they realize that they, that there's [00:21:00] yield in the platform, whatever the platform is. And I want to be clear, like, I want to sell my product, but you, you can do, you can do really good platform engineering and infrastructure platform engineering on just Git.

That's it. And I, like, if you, if you're like, how in the hell do I do this? Corey, like send me an email. Like I will pair program with you and walk you through it. But if you want to buy my stuff, like, please do that too. But like you can, you can get really far without investing in products and, you know, open source tools and whatnot.

You can do, you can do a lot with very little. Just before we sort of delved into that, you, you were talking about how. Most organizations, and I've been both guilty of this and seen this multiple times in organizations where you're looking at the single two best platform engineering organizations in the world and saying, I want to leap straight to, you know, boss level on this from a maturity level of zero just because I've moved lifted and shifted a few things into the cloud.

I guess what do a sensible you've [00:22:00] touched on it a couple of times with the use of things like, you know, you can just use GitHub and you can just do this. But what say, do the next steps look like from that point onwards? How does scaling this out work? And how would something like a visual you I help with that?

Yes, I think, you know, kind of scaling it out. Is, you know, first, like, starting with what you have, right? So, you know, a lot of organizations, and I'm talking from, like, the people that I'm seeing today. So, like, we work with a lot of seed stage and series A companies, right? And so, and this, every company is here at some point in time, right?

Like, you may be well beyond that. You're series D. You've figured everything out. You've got the money to hire a thousand ops people. Like, you've got it figured out. But everybody goes through this phase where they start out with a small team and they're trying to scale up, right? And when you're building a startup.

Seed stage series a like you're looking at that first ops hire, right? And this person's making, depending on where you live, like you're probably making like 1 80 to 300 a year. Somewhere in there, like all in comp wise, right, as an ops engineer. And [00:23:00] so as a startup, you're looking at this person, you're like, okay, do I have 40 hours of work for this person to do?

And the answer a lot of times is no, right? And so like, they're looking at it and think, okay, do I spend 180 and like find busy work for this person? Or do I look at my engineering team and go, I'm going to make all them do some cloud stuff, right? Or I'm going to pick one and that one's going to be like my guy or, you know, my person, right?

And so, You know, when you're, when you're first starting to scale this out, you got to think about around like what you have, and one of the things I see people stumble across a lot in startups is, and I think this is startup. It's just a symptom of being in startups. I think it's a symptom of like just VC culture, but like we're just trying to bum rush to anything somebody will buy, right?

And so, and so a lot of times it's just like, oh, well, you know, yeah. Is that a best practice? Is that secure? Okay, I actually had a health care startup once tell me that that HIPAA wasn't a revenue driver. And I was just like, Oh, you're fucking killing me. Like, my master's in health care [00:24:00] information systems.

Like, it just crushed my soul. I'm like, it's not, but like, you might have my mom's data. For the love of God, don't say that to me. Right. And so like this bum rush where we just like get software out the door. That right there, that pace, it's very hard for you to stop if you don't have like a really good automation, IAC experience, infrastructure as code experience, like your engineers aren't stopping to say, I should, you know what, I should terraform this production environment in case we need to like, make sure that we have parity between staging and prod or like, make it easy for people to follow it.

They're not thinking about that. They're like, you know what, like, We're being rushed to get this out the door. I'm just going to click right. And like me, I can move faster in Terraform than I can in AWS. But I've been writing Terraform for like a decade. Right. And so like most teams don't have that. And so what you have is you get to this point where like you're at a boiling point.

And now you're like, Okay, we have to hire this first ops person, or we have to go to an agency or, you know, we have to look for a tool that can reverse generate some Terraform, right? Like you're stuck in a spot. And now is that first ops hire? That's a [00:25:00] terrifying role to walk in. You walk in like you're our first ops hire.

You're like, okay, we've got 15 engineers and you've been in business for three years and you have no IAC. You're telling me my job's going to suck day one, right? Is that because they have to kind of unravel all of the spaghetti? I call this the new. tech debt. Yeah. And all the legacy, which is you've double clicked on so many things and not maintained it.

Now it's just a, uh, uh, a rat's nest, an explosion, escape factory, wherever you want an explosion, a spaghetti factory, you know, it's like just stuff everywhere. You're using the technical, I thought you're using the technical term there. And it is the fear of somebody just going poking them going, do you mind having a look at That and maybe fixing it.

Yeah, and it's fine. It's wild because it's like, you know, like getting started like So it's like that's that's the baseline for a lot of companies, right? Like your your ops person you come in and you're like wow It's on fire and like you can see this on reddit You go to reddit slash devops and it's like every day It's just like you can see the people that have been working at a [00:26:00] company for like years.

They're series d They got a good team. They're like, I don't understand why your life's a miserable person They're like i'm the first hire at a series a company It's like like they've just lost like what how bad it was like back when they were that small. Right. And it's funny because like people come in with like The debt that you come in with sometimes, like we had a customer that when they joined, we were like, well, what's like, what's your what's the biggest problem you're facing right now?

And they're like, like a 10th of the Internet can't even access our site. And I'm like, that's a

weird one. And so like, we start digging in and they used a public account. IP block for their virtual network, and the public IP block they picked is like a tenth of the public Internet. And so it's just like they can't even like traffic will come in and then like their servers have no idea what to do.

And so they're like, how do we fix this? And I'm like, unfortunately, You have to move everything to a new VPC, right? Like, you can't change this CIDR range in AWS, right? Like, you're going to have to move all of your stuff, and like, to them, like, that was terrifying. For us, it was, you know, we, we, [00:27:00] we helped them move the databases.

We helped them move the workloads. Like, we got it done in, like, less than a week with no outage. But that's, like, the experience. That you get from somebody who really knows it. And when you're trying to grab that one person for your first hire and then you put them in the middle of the spaghetti factory, how do they get it?

So does your stuff in the gooey sort of automagically discover through the spaghetti and then create a series of patterns? Or is it more of a manual lift to get it to a point where it's kind of understandable and it's got reasonable components and things like that? Yeah, so the way we, we operate is, so the platform doesn't do most of that stuff, so the platform at its simplest way to think about it is it's a visual infrastructure as code management tool. So what's really cool about it that makes it really easy for these smaller teams to kind of scale themselves is, as an operations person, you get to think about the services that your teams are using, or the services your teams are going to be using.

You can write your infrastructure as code in OpenTofu, etc. You [00:28:00] publish it, and instead of your developers having like the full view of the cloud, they essentially have this GUI that is a diagram. But as they diagram stuff, it's pulling from your Terraform registry that you've built. You've built your registry, right?

So now, as a developer, I go and I say, Oh, I need this new service. We use Redis. I see Redis on the sidebar. I drag it in. I connect my apps to it. And when you connect your apps, it does the I am behind the scenes. When you connect your apps, it does the secrets management behind the scenes like you don't have to, you know, Try to copy and paste secrets around or whatnot.

And then when you provision, it runs your terraform, it runs your policy tools, right? And so as an operations person, I know that every time my developers can deploy Redis Postgres, they can make clusters and networks, they can make as many things as they want that we support. And I know that all my policies and procedures and best practices are codified into those what we call bundles, and they can deploy them.

Now where this like networking stuff happens is Something that we do as a part of, you know, some of our plans is we do statements of work when people are coming onto our platform and we break even on them. So [00:29:00] like we we literally do like migrations and statements of work and we break even because like the platform for us is so like we have such margins on the platform that we can afford to just.

Sell at cost to do work. And so when people come in a lot of times, it's like they're, they're to this point where they're like, we have three people on our ops team, but we've got like 100 developers. Like these three people are outnumbered. We need them to feel like 10. We don't have the money to hire 10 more people.

Like, how do we scale this group? And so a lot of times we'll come in and say, okay, like you three people, like you need to be focusing on like how you can better serve your developers

and you've got some debt. Yeah. Give us your debt, give us the stuff that you three people do not want to work on, and you focus on the stuff that's gonna make your developers more efficient.

And we will handle, we'll handle the painful part because we've got a team that's got experience in doing application migrations, like database migrations and stuff like that. So it makes it very easy. It's not like three people that might not have done like a Postgres live migration before. Right. They don't have to figure out how to do it and hope it works.

It's like, oh, [00:30:00] we've got people that have done that before. Like, we'll just do that for you. One of the things you touched on there, which I think is is really interesting in the sort of cultural. Change that's gone on in the world of digital engineering is this notion of serving your developers that you just that you just touched on like previously in the past when you'd have say an old school infrastructure team it's almost calibrated entirely the other way where the infrastructure team is, you know, creating You know, walled gardens and firewalls, and they're gonna give you a series of rules, and you're gonna have a series of SLAs and actually, a lot of the time, you know, perhaps intentionally, and certainly in hindsight, those things are restrictive. Like, how would you characterize this notion of serving the developer? And why is that important? Yes, I mean, I think that's the tenant of DevOps that we all I'm not we all but like there was Missed a lot right and like it's really funny like where you look at where we are today and you look at when like the term DevOps first came out for a lot of organizations.

We're in that exact [00:31:00] same spot and that's not just like my gut. Like I see that a lot and I hear people say that a lot. I read it on Reddit a lot, but if you look at the Dora report, the Dora report on page like 37 just admits it. It's like 50 percent of people, 50 percent of the people that are keyed in.

To the door report DevOps in the title, right? They know about it. They're thinking about it. But then when you look at their answers on the survey, it's like 50 percent of people still are in that when we have an outage, it's multiple days. We deploy once a month, right? Like we never got to that cadence of that, that infinity loop of CI CD.

Like, that's the dream, right? And like, I have this blog post, like DevOps is bullshit. And people like, Oh, how dare this guy? And it's like, we need to look at the industry. We didn't do it. Like your company may have but you're an outlier like most of us never got there in the first place Right and like and that's the hard part is it feels like we act like we do but for a lot of companies We're where we were 20 years ago.

Yeah. Yeah. Yeah, you get a lot of there's a lot of [00:32:00] Enthusiastic talk about it and there's probably a lot of people that have put themselves out there and got business cases and things like that to try and achieve these things, but you're right The reality is some is somewhere short of the dream at the moment Yeah.

And when you look at like these teams, like a lot of them are still at odds with each other, right? Like again, the operations team's job is to keep things stable. The engineering team's job is to introduce change to make more money, right? Like that, that's, that's it at the end of the day. But that shift for a lot of operations people to their customer is the engineer is hard to do when there's three of you and there's 100 of them and you're drowning under debt.

It's really hard to you. Think about that other team when you're trying to think about your own team and like your own morale, right? And that problem right there, that's not that three person team's problem. That's not that hundred-person engineering team's problem That's a management problem like the like the management of or the organizations where

you see these Operations teams struggling to get to DevOps and to serve their customers.

It's because you're not [00:33:00] putting enough effort into into the operations team and their time and letting them get their head above water, right? At some point in time, like you have to take that Google like SRE mentality of like the error budget. Your entire error budgets blown. Like you're blown. Like when you're three DevOps people and a hundred engineers, it's blown.

Stop. Pump the brakes. I know we got to make more money for the business. But if we can get that operations team's head above water, they are the unsung heroes of your business. And if you can get their head above water, Operations engineers are your 10x engineers, but when their heads below water, they're your 1 10th engineers.

They're slowing everybody else down. And like that right there, like getting them above the watermark is absolutely important to getting DevOps to work. And so many of those teams are just struggling. They got their nose. They're just like, Oh, I'm above water. But it's just like a, it's a, not as a single nostril.

They've got, they've got like, they've got like an air bubble at the top of the cave and they're just like pushing their face into it. Trying to, trying to get this. [00:34:00] Better than it was last week. Oh my. And like much, everybody talks about automation and particularly AI coming along as, as being the panacea to solve.

I mean, most of humanity, these problems. before it starts to create a whole new set of bigger problems. Or destroy humanity. Yeah, exactly. Don't forget the arc where it destroys us eventually. Exactly. Robots will rise. Exactly, Rob. I notice you've been reining in your, uh, your negative commentary about AI over these course of the last half of the season.

It's unpopular, Dave. It's unpopular. I'll wait until we're all in servitude with the robots, and then I'll be able to say, I told you so a lot. So it, you know, I'll have in a glee from the, uh, the, you know, I was right. You could maybe start the Rebellion I mean, you could be the leader Freedom Fighter. Yeah.

I'm on, I'm on the Freedom Fighter side. I am, I had a, I had two funny things happen to me the other day with ai. I am, I have always been, it's funny, I had a conversation with a buddy [00:35:00] four or five months ago and I'm, I'm mean to ai. Like, I'm just like, Oh, you piece of shit. You don't work, like you got to, you got to watch that.

You got to keep track. You're on the list. This is funny. So like, I mean, like, you, you know, you're using a tool and it kind of sucks. Right. Yeah. A screwdriver. It's stripped. You're like, ah, this needs to be shit. I need to order a new one. Right. And so like, I'm asking chat GPT. Something I was using it for something that, I was familiar with, but I just didn't feel like doing the work.

So I'm like, yo, do this for me and it did it. And I'm like, that's not right. Now I literally, I didn't tell it what was wrong. I just said, that's not right. And it was like, oh, you're right. It's not right. And it fixed it. And I was like, I didn't even tell you what was wrong. That's pretty funny. And so I'm like, you're such a piece of shit.

And it was like, I don't appreciate the language you use with me. And I'm like, you aren't an, I, you aren't, you are not an, I, you are a thing. I can use whatever language I want with you. And then the other day I was, I asked it something and it was like, well, as the CEO of mass driver, and I was like, This is a new chat.

I was like, how do you know that? And I was like, I maintain, [00:36:00] I can look at our other conversations. I was like, Oh God, you're going to be, you're going to be in a court and

there's, you'll be held by two robots and they're going to bring up your chat history and they're going to go down it and you'll be number one. That's this is it. This is the end. Oh my gosh. You better start groveling now. Yeah. I told her the other day that it was as dumb as a Tamagotchi.

So I think, I think, I think that's like the most offensive thing you could probably say to an AI, like that'll, that'll be me on the, uh, on the, on the stand, they'd be like, did you call Chichipate a Tamagotchi? And I'll be like, I did not like we, we actually don't delete your chat history. Yes. Here's where you said it was all recorded.

It's all recorded in perpetuity forever. Exactly. Anyway, going back to how it might be helpful in the short term before those sorts of repercussions start to kick in, where do you see automation fitting in? This seems, it feels a bit mundane going back [00:37:00] to this now, but remember when we were in the cave and we were trying to gasp for air, how do you, how do you see AI and automation sort of helping with that?

Like. Turning the team of three that you described into a team of 30 or whatever that ratio actually ends up being. So I think, I think there's potential. And I think one of the things that's hard with the potential is there's a lot of context that's the AI is missing. Right. And so it's funny, like when I feel like when AI started first, like happening and everybody was panicking, like there was this like moment in like, That was like December or January.

It's like, Oh, 84 years ago in January, uh, where it feels like it has felt like that. I was like, we were talking about this season. So we're coming up to the end of season three. Right. And we were, we were doing a little bit of work on what the season And episode is where we reflect on the season. Right.

And we were like, Oh yeah, the first episode that we did was like with Irving Visser. We're like, was that just the season? It's like that was in September last year. [00:38:00] And it literally feels like about three years ago. I didn't have forehead wrinkles before AI. Yeah. So like, it was funny. There was this like panic moment where everybody was like, I remember when, what was the thing, Dolly?

Yeah. Right. When Dolly came out, it was funny. And then, you know, illustrators were like, ah, it's going to steal our jobs. And people were like, no, like the art that it generates is like bad. Like the hands are silly looking like noses are on foreheads. Like it's never going to get art. Like we got to worry about developers jobs.

And now six months later, people are like, Oh, actually, we have to worry about the artist's jobs. And what's interesting is the, the context, right? Like when you think about us as humans, like we have experience and exposure, right? Like there's, there's like the, the, the experience that we get, uh, going to school or reading a book.

And then there's like, or sorry, the exposure. And then there's like the experience of like you going about the world, doing things. Oh, I was robbed or like, Oh, I was on this plane. And like, uh, you know, it fell 10, 000 feet and it scared me. And like, there's things that shape your life that, AI doesn't [00:39:00] have.

Right. And so then when you start thinking about like what it does have, this has access to everything on the internet. That's what it has. Access to everything on the internet. Great. Unless you put it in your robot's txt file, then it won't. Uh, So, yeah, But then it has access to the internet, right? So now, we look at it, it's like, okay, well, is it going to take developers jobs?

I'm sorry, I don't believe in Devon, like, okay, guy. No, I don't think so. And here's why. Authors

and artists put their absolute best work on the internet. Art authors have to go through an editor and get published. That's what it's trained on. Oh my gosh. It wrote star Wars. It can write, uh, uh, you know, a follow-up to like a star Wars episode.

That's amazing. That's great. But could it have written Star Wars if it was never written? No. It didn't have the experiences and the exposure of the writers of these, you know, episodes of the original movies, right? Like, it doesn't have that, like, it can only be based off of [00:40:00] what it knows, right? And then looking at art, Behance, Dribble, like, people are putting their best art on the internet because they want to attract customers, right?

Or they want to show off their art. Like, like, our arts, we've trained it on our best arts. But then when you look at software, it sees A snapshot of something on GitHub or a snapshot of somebody stumbling through some shit that they didn't understand in the cloud on medium. com, like the software it's trained on is our absolute worst product, right?

Again, like we're being rushed through stuff. We have PMS that want us to move faster. We're not sitting here like trying to write the Iliad of software. We're trying to get something to work to get out the door because we got 40 hours and we want to get back to our families. And that's what we're training on.

So we're training on low quality Content for software. But then we're also not showing it how production works. And that's where I think the big gap is. It's easy to say, Oh, look at this. This A. I. It totally wrote a for loop really, really efficient. It's like, okay, does it understand the context of my [00:41:00] cloud architecture?

The request rate of my customers? The average like throughput? the average request size, like my actual network diagram and all the cloud services that I'm using and my metrics and like, what are KP eyes to my business? No, it's not trained on any of that, right? And so the idea that I can sit down and be like, okay, generate me a business is highly unlikely.

Right. It business logic also isn't something that's really trained on. Right. And there's like one thing. It's to find a software bug in like a logic error, like off by one or something like that. But it's another thing where to get to understand your business rules, right? Like I worked at a company once that had you could only put one item in your shopping cart.

You can only buy one thing. That was the business that made sense of the business actually made a money. You put more than one thing in the car. People got distracted. All the price is high. Do I want to buy this? And so they're like, we're so one thing. That's it. The software. Look at that and be like, that must be a bug.

It's like, no, it's not a bug. That's that. That makes the business more money than putting more things in the cart. Right? And like, that's the kind of context that I think that it's missing. And when it comes to operations, [00:42:00] most of the stuff that we do isn't in code, right? It's me looking at the metrics of the system.

It's me looking at health checks failing, right? It's alarms going off, and like, sure, like, there's different people that can train on a little bit of that, but like, if if PagerDuty trains on the alarms, it's like, okay, well, they know, like, when alarms go off, they can help you with some alarm stuff, but they're missing the metrics that Datadog has.

And maybe Datadog's doing some stuff with your metrics and doing some AI training on it. If so, I'd Oh, my gosh, they might actually be able to take us like take away our jobs, but then our codes and get hub and our stuff's on and like it's it's distributed all over the place. Our context is scattered in operation.

Are you thinking about how it how it applies specifically in the platform engineering and

mass driver space? Like how how it could help it maybe small ways? Or actually, do you just think it's, you know, it's something that is untrainable at the moment. So I think that a couple of things. So we do use AI for very like Tedious tasks that are, you know, [00:43:00] cloud solutions, architects are like experts understand and can evaluate.

Right. So like going back to that, I think I said earlier where I was like, I asked it to do something and it did it wrong. It was actually for a webinar I was doing a few months ago on online replication of Postgres. So like doing a database migration of Postgres without downtime between it. Yeah. Two networks, two versions, whatever.

And I was like, hey, can you like do the replication config for me for an instance? And it gave me some stuff about MySQL bin logs. And I was like, And that's not, that's MySQL stuff. That's not Postgres stuff. But I've worked in MySQL and Postgres for years, so I recognize that immediately. Somebody else may have just copied and pasted that, and then been like, why doesn't that work?

Right? Like, they wouldn't, they're missing that, that context, like, to be able to solve that problem, right? And so where I think it's really useful is when you absolutely understand the domain, but you don't have time to do, The tediousness, right? Like that, that tweet a few weeks back is like, Oh, like I wanted AI to do my dishes and laundry, not to do my art and writing, right?

Like, like we wanted to do the tedious stuff. And so like one [00:44:00] place that we find it very useful is we actually use AI for doing Terraform generation from cloud resources. Now, if you hopped onto chat GPT today. And you said, Hey, let's say you run the AWS command to list all the resources from your AWS account.

And you just paste that Jason blob into chat GPT and say, give me give me the terraform for this. It's going to be it's going to give you a bloodbath. And it's it's it's not going to work. You can copy and paste it won't terraform be like, not even a terraform. But when you're really familiar with Terraform, and like you're really familiar with like how you've manually done These like reverse terraforming of cloud resources before I can start to automate my steps in prompts.

And so like we have probably 20 or 30 different prompts that a set of cloud resources go through to generate terraform to back it out to variables to back out all the resource referencing to look at the staging versus production and back out the abstractions. And so something that used to take one of our ops engineers like eight hours to [00:45:00] like back out maybe 10 resources to some Terraform, they're now able to do in like 30 minutes.

And it's a lot of like, you'll run, you'll run a prompt, you'll see it and be like, okay, that prompt that that output looks good for the next prompt, go, go, go. And sometimes we'll just like, let the whole thing run. And like code will come out. And you're like, that's great. But then we'll like back it up. And we have to figure out, okay, this third step, it's getting something wrong.

And how this works, right. And so like, that's not it's not magic of AI. It is magic of experience. Transcribed Right, like we've seen this so many times, we're able to give an extremely exhaustive list of like do's and don'ts for every single step of the way. And if you don't know that, it's hard to get it right.

And so that's one of the places we're using it. Now, one of the other places where, I don't know when this episode goes out, so I don't know how far along we'll be in this, but, and, and this is good. When we roll this out, this will be opt in, uh, because we're not evil bastards like that make you opt out via pigeon mail.

Sorry, people. If you're listening to this, your company's horrible. Uh, but one of the things [00:46:00] that's interesting about mass drivers, we actually have that full context. Right? Like we have access to your, Sorry man, you don't have any friends that talk to you. My Twitter is Cory O'Daniel. Feel free to yell at me but seriously, that that email they sent out was crazy. I don't know if you guys know what I'm talking about, but one of the things that we have is we have, we actually have a massive amount of context. And so we are starting to work on some AI stuff, and it's going to be opt in. But what's interesting about MassDriver is we have access to a lot of the information That a bunch of these companies have in small silos, but large volumes, right?

But for our customers, we have access to their entire volume and all this information. So, like, when you publish a Terraform module or an Ansible playbook to MassDriver, we understand your code, right? We understand that you've got three environments that it's in. deployed in, and it might be deployed seven times.

I've read us deployed as a caching cluster. I've read us deployed as a sessions database, right? So we understand the [00:47:00] code. We understand the use cases of the code. We understand the environments their configs. But we also see the costs. We see your metrics. We see your alarms because the platform kind of handles all this stuff.

So we have the full context of your operations, whereas most companies have snapshots. And so we're starting to do some interesting AI stuff. The first stuff we're rolling out is just kind of essentially infrastructure as code assistance. So it's like you've deployed something. One of your developers went to use it.

There was some sort of fault in Terraform as it was rolling out. And we'll look at, okay, here's how they used it. Here's how other people have used it. Here's your code. This is what we think is wrong with your code. Right. And so it's now as an operations person, you come in, you're like, Oh, wow, what am I, we have redis deployed 15 times, somebody that just deployed it brand new, it didn't roll out.

Why? That's that sucks to troubleshoot, right? Like that. Like if you're if you're sitting at a GitHub repo Terraform modules, and you're looking at a dashboard and data dog trying to figure this out, you're like, sucks, that sucks. But like we're able to use, we're able to [00:48:00] pull all this stuff together and say, Okay, what's what's different about this use case versus these other 15 use cases.

And we know this is in staging. And we know And we know these are three prod use cases, and these are seven staging use cases. Like, what is wrong with this? This configuration specifically. And so, you know, suggest a little diff. Like, hey, this to change about your Terraform to make this work without breaking all of that.

Like, that's powerful, right? Like, and that's the thing that like, that's tedious work. There's a point there about software developers have been using things like Stack Overflow to source code. You're creating a community that's similar. There's two sides to it. One, the opt in and be part of the community and help others and make your own day easier versus the others who might go, no, I don't want others to see what I'm doing, but are you actually losing any real competitive advantage by being part of a community?

I think development over the past 10, 15, 20 years has shown the community helps everyone for the greater good. I'll just say it'd be interesting to see how many opt in and become part of that. I would hope it would be a lot because you can see a lot of benefit from joining what is already a scarce resource together with that knowledge.[00:49:00]

Exactly. Yeah. And I think that's one of those areas where. It's again, it's that like unexciting part of the job, right? Like troubleshooting, like a very specific configure, like one in 15 deployments is not working well with a specific Terraform module. Like as an ops engineer, like when that pops up, like that's one of those things.

It's like it's probably almost immediately an emergency, right? And it's taking you off of whatever else you're doing. And now you're trying to troubleshoot it. And it's just like, at the end of the day, it's like, okay, I'm going to fix it. What else is it going to break, right? Like that, like you're always thinking about, like, what's this change?

How is this change going to affect, like, the rest of the, you know, the uses of this Ansible, Ansible playbook, right? And so, like, that's the part of the job where I feel like it's not super exciting. And I feel like that's where, like, AI fits in well. It's like, I, I have so much other more important work that I do better than AI.

Like, let's let the AI do these simple parts for us. So pulling ourselves back out of the AI discussion and maybe just to bring our conversation to a bit of a conclusion. Let's return to the [00:50:00] explosion in the spaghetti factory and the basics of getting some of this right, which is still quite a big industry problem for us.

So maybe to wrap us up, Corey, what are the first one or two steps you would suggest to one of those organizations or individuals that's stepping into a massive explosion at the moment and just wondering what to do? Yes, I think if you're, I think if you're, I mean, If you're a company that's ended up in this space.

You know, where you're, you're like, oh man, like we, we waited too long to get an ops person. I think first off, like you're not the only one, like don't give yourself too much of a hard time. And, and the easiest way to fix it is like in probably, I mean, honestly, if we spent more time inventing time machines, that's like, that's, that's easier than a lot of the debt you're going to go through to fix, but like, I think, as that organization and the person coming into that role, like I feel more for that person coming into that role because I know it's going to be a hard job, but I think the first thing you have to do is you have to take a step back as an operations person and not come in and say, Okay, you know, how can I get in here and use some [00:51:00] cool new tools?

You know what? Thousand things do I need to learn about? I think the first thing you need to do is like go back to that mindset where these developers are going to be your customers. You're going to need to have a good relationship with them. There's a ton of debt. There's a ton of stuff that's probably going to be broken.

You know you're going to be in hell for a bit. And so putting yourself through additional hell of making animosity with this team is not going to help you, right? And I feel like that's one of the things that's hard when you come into this role is like figuring out where to start. And like your manager might have an idea of But the real answer is those developers have an idea.

And so day one as a DevOps person or an operations engineer coming into a company that hasn't traditionally had it, I think one of the most important things is to kind of put on that product hat. And it sucks because operations people are consistently the furthest from product, right? Like it's hard to put on that product hat and think through like, how do I build an MVP, something that really accelerates this engineering team when you've never been in a product place before.

[00:52:00] And so I think like some of those more when you've Soft skills as an operations

person is more valuable than walking in saying, I absolutely know. Kubernetes inside and out, right? Because what I want to do is I want to get in. I want to talk to these 2030 engineers, figure out what trips them up the most.

Across the entire team and I want to focus on that and like that is my MVP and it might be a platform probably isn't you're probably not deploying backstage and I probably not even buying mass driver at that point in time you're finding something where it's like, ah, we consistently have 20 to 30 minute builds because none of us knew Docker.

And we just threw some stuff together. Great. If I can go in and focus on just getting the builds of Docker down to sub five minutes, everybody's builds are faster, right? Like, I just, I'm a hero at this point in time. CI is flying, GitHub Action Bill is dropping, right? Like, coming in and like finding where the engineers are frustrated, like that.

That's your job, day [00:53:00] one. Not to come in and have, you know, the CEO or CTO or director of engineering say, this is the thing that you need to do. It's like, no, let's see what the engineers are struggling with and how I can help them and then just stay in that loop. And if you're just continually solving the problems of the engineers, you'll get to the point that you have a platform.

Eventually, it might not be super sexy looking. It might not be this beautiful thing written in React. It might just be some Terraform and some GitHub repos, but if those engineers are happy, Your relationship is going to be great. You're going to start to feel DevOps. Like you're going to feel yourself moving up that maturity ladder.

And like that is the part that is extremely hard that we lose sight of I think when we walk into waste deep debt.

So David, interesting twist, but what have you been researching this week? So I thought I'd have a look at the context that exists around [00:54:00] platform engineering. It's something we've talked about on the show a lot, this notion of products and platforms based organizations, but actually what are the sort of five core elements to, to putting something like that in play?

Is everything okay? You alright over there? Cory? Sorry, my network is so bad. I'm going through my phone and I set it on top of a picture frame to get it higher. Because the higher my phone is I go from like 1 megabit to 200 and it just fell off the picture frame and hit the back of my desk. Uh, yeah, at least you didn't drop it on your face like I did, because it seems to be working better at the moment.

It's actually a dramatic improvement. That's great. All right. So let's, let's talk about then maybe the five ways or the five things to consider as you're building out a product and platform organization. First is build the product teams around the end user experience. [00:55:00] So focusing on the end user experience to, I think, prioritize.

And keep a track of the direction you're moving in. It's very easy to get lost in the weeds with all of this stuff, because there's always good stuff to do, but what creates prioritization in the backlogs user experience is a good place to start second one. I think we'll appreciate this from the conversation we've just had, which is don't forget the platforms.

Why are platforms important in this situation? Well, they're important because of all of the things we talked about. They drive the performance of the rest of the organization. If you get that wrong, it's going to come constantly drag back your ability to perform and your ability to hit new levels of performance.

The third one is around funding. So how does funding and funding frameworks become

important in this situation? Well, how the product teams function and how the platform teams function and how they use, use and consume finance as they go forward is [00:56:00] quite different to an old Portfolio organization and how portfolio organizations function.

And I would add that to number four also, which is about accountabilities. So ensuring that the product leaders and the engineers have got delegated accountabilities to allow them to make decisions. And they understand the differences between a good and bad decision based on clear principles that help guide them and not constrain them.

And then I would have a look at then also the interface between. The platform and product organizations and the business organizations and to get that as close as possible and where possible, frankly, lose that boundary, particularly when you're talking about the product teams and then bringing all of that then together with with the underlying platform engineering conversation.

We had Rob. So What do you think? Does that resonate with you? And are you spent like a lot of time [00:57:00] thinking about these sorts of organizations? Yeah, the , what's cloud done? It's allowed us to decouple and build organizations a different way. I'm a really strong believer and it's part of the conversation we just had is taking away the pain.

Hmm. Uh, from the product team so they can focus on the outcome and get it right and let the platform do the heavy lifting. I mean, it's the undifferentiated heavy lifting point, which is why do it yourself when somebody else has already done it? We still have to get out of this mode of i'm a creative engineer I want to go and play with all of this stuff and create my own I suppose once you get past that point of pride it allows the platform to do its job and let the product Thrive and then the person at the end of the experience or the business process that's attached to it gets great results.

But there is a discipline in here, which is you have to, you have to understand that through that shared responsibility model to use the hyperscaler phrase. You have to do your role and do it well and understand how you support the others around you so that they get what they [00:58:00] need so they can go and do what they need.

So it is a, you know, a continuum from that perspective and don't get lost in your own domain. But remember, you're there to support the domains that you touch and support. So I suppose, yeah, that's, that's how I see it. So yes, broadly agree. But there is a, there is a lot in what you've just said. In making sure the organization operates in the way that you say this, it's, it's five points, but it's a lot of work to get there.

It was a lot of research, Rob. Was it? Yeah, I, I know I distilled it down, crystal clear. It was value adds, it was huge value, that's the human in the loop still adds value as, as Corysays, you need the human for the specialist analysis. And a special big thank you to the author of the big product and platform shift paper that's on the, on the internet that was extremely useful and very well clarified.

Although there is a, there is a really interesting point about the products and platforms organization that people think. It's the full edge to edge operating model, and it's not. It's an operating [00:59:00] model that lets you get value out on a product out. But don't forget the wider ecosystem that's required within a bigger organization to actually do the other jobs that are still critical.

I think some people have become consumed by we're going to be a purist product and platform organization. They forget the rest, and there's big gaps. And I think some organizations have almost destroyed themselves in the pursuit of it by forgetting the other

things that are still still must exist within it.

So that research. Not not that particular paper, but that the research has to be taken with a pinch of salt because there is always a pragmatism that has to kick in with deep very much exists with an organization. I think very good point. Very well made. And I think it's Coryaligns to your point, I think, which is you look at you look at examples like the spotifies in the Google from a platform engineering point of view and think that's the Nirvana I'm shooting for.

And you so naively you. You. Trying to create the moonshot did point Rob's point resonate with you. It's the same thing. I think when you expand that out to all of the products and platforms are going to do style of working, isn't it? [01:00:00] Yeah, it's really funny. I know you as you're saying that like it get high.

It triggered some some mental trauma This this team that I worked It's like that's the thing with like platforms It's like I don't think like you look at any platform out there right like any any platform as a service etc alike None of those are going to work for 100 percent of organizations, right?

And when we look at organizations like team to team, like teams are pretty different, right? It's like the idea that you're going to build this perfect platform where you're going to buy this perfect platform, and 100 percent of the company is going to use it. Like, that's bonkers. Like, things are just too different.

Like, your front end team's doing JAMstack stuff. Do you want to, like, jam them into Kubernetes? No, like, Vercel is probably better. There's probably, you know, something on AWS Amplify that's better, right? And so I remember this team, like, We'd gone in to help this company and they were like, we do everything in the cloud with Terraform.

100%. And I was just like, that's cool. But then I saw their data science team trying to manage big query tables with [01:01:00] Terraform. And I was just like, guys, like you're, you're just, you're just torturing these people. Like this is a tool, like they don't even know what Terraform is. Like they know how to move data.

They know how to write queries, and now they're trying to manage big table or big query tables like with terraform. And the funny thing was, like when I went in to, like, help this team, like, look at what they were doing. They weren't using Terraform. They were using the G cloud command line and they were just calling it with a null resource.

Like they appeased, like the team that was like, we, this is how, this is how we manage infrastructure. We do with Terraform. And they're like, okay. I've got to compliment you, Corey, on a, on a, on a first class developer gag there, by the way. You don't get, you don't get very many of them, but that was like, it was peak.

Yeah. Yeah. But it is that, it is that horses for courses to understand when you do need to deviate and do something different and not be dogmatic about a principle just because [01:02:00] you're principled and it's like, you know. Yeah, exactly. And the thing that's interesting is, like, when you see that team that needs to, like, diverge, let them.

Because what you might find is, like, you might find an abstraction or a way that they're doing things that you can start to incorporate now into the platform that other teams might need in the future, right? The skill is, I think, allowing the diversion and then bringing it, bringing it back to something that then becomes repeatable, not dogmatically pushing forward on what you thought was the, you thought was the track.

I, I always value those digressions. Yeah, they're important. You can get some great stuff

out of them. Yeah, exactly. Look, Corey, thank you so much for spending some time with us today. It was, I've got deep envy looking at the back of your patio doors there into sunshine, given the appalling summer we seem to be having in the UK at the moment.

But it was, it was wonderful, wonderful talking to you. Thank you so much for your time. Yeah, I appreciate it. Thanks for having me. Apologies for the network. It's always the network. It happens. It happens. And it [01:03:00] was all fine until you started dissing the AIs. That's all I'm saying. Just to be clear, I'm the databases and Kubernetes expert.

I'm not the networking one, so apologies networking people for putting my iPhone 15 feet up in the air. Like, it seemed to go very well. Anyway, so look, we end every episode of this podcast by asking our guests what they're excited about doing next. And that could be, you know, something fun. You're going to do it the weekend.

We're recording on a Friday, or it could be something that's coming up in your professional life. So Corey, what are you excited about doing next? Mine's a combo. So, I am, I'm a Florida boy who lives in Southern California now, and there's nothing I love more than just tanning my alligator skin, uh, sitting in the hot tub, but I'm also a founder, right?

And so like juggling these two lifestyles is very difficult. I want to lay in the hot tub. I need to send emails. And so, uh, my son and I, my son's almost [01:04:00] five and I'm, I'm a big believer in like, I don't like the idea of like treating kids like kids. Like I like introducing like work and tools to him. Like he built his own little tower for, for standing in the kitchen.

Like he didn't use the saws, but like he got to do the screws and drills and whatnot. So next big project is we're building a quarter round, uh, teak deck for the hot tub that sits right at the, so it's, it's an above ground. It's like one of those like Nordic wooden hot tubs that we have. So we're going build one.

So dad. On Friday afternoons. Cause I got Friday afternoons off. We do half day Fridays. So dad can have a little margarita in the hot tub with his laptop doing his emails. So I'm going to do all my like weekly planning and my emails. Sorry, VCs, if you're hearing this. Yep. I'm going to be doing it from the hot tub with a margarita and enjoy living in San Francisco guys so that's the next big project. I'm really excited. My son's like, do I get to use solves? And I'm like, no, you don't, but you can use the [01:05:00] hammer. And you can paint it or not paint. That is like a microcosm of work life balance right there, Corey. You've painted a picture. I think you've invented a product, right?

That's it. We need to industrialize that. I need some of this. Yeah, it's just like, just make sure that, you know, secure the laptop so it doesn't go plop into the water and all's good. And is your Is your Swedish hot tub, is it wood fired or is it electric? So it's, it's electric, but yeah, it's uh, so it's, it's electric.

It's actually, it's amazing. It's electric. And then I don't know, I should read more about the technology and how it actually does the water, but it's a no chemical hot tub. So there's like, Like when we bought the thing, they're like, there's like the guy, it's fine. The guy that was selling it has no idea what it is either.

It's just like a guy selling hot tubs. And he's like, yeah, it's my wife's like, we can't have chemicals. Like, I don't want like chlorine on the kid's skin, you know, just like crazy California thoughts, so again, bleach is fine. Throw them in it, and so we're talking to the guy, he's like, Oh yeah. So this thing's got like [01:06:00] lasers and it just like shoots germs and kills them.

And so like, you never have to put chlorine in it. And I'm like, I don't think that's how it works, but like, we're going to buy it, so we have, I just tell everybody I type, I have a, the thing has

lasers in it and it shoots the germs and we don't have to put chlorine in it. It's great. To get a light show late at night.

So you sat there doing your emails. There's a great big laser show going on and then a band pops up and then the music starts and off you go. Yeah. Yeah. Pretty much. Brilliant, Corey. Well, look, enjoy that, man. I look forward to some pictures for that. That, that sounds like it's worth documenting. Yeah. Yeah I think that, I think that if you go on, uh, if you go on my Twitter and you look back a few days to June 10th, you can see why this is important. As you see a picture of my laptop on its side, cranking music, pushing rock, pushing water out of it. I, uh. Don't balance your laptop on the side of the hot tub.

That's all I'm saying. It doesn't work. Necessity is the mother of invention there. We [01:07:00] have the answer right there. So, uh, good. Anyway, a huge thanks to our guest this week, Corey. Thank you so much for being on the show. Thanks also to our sound and editing wizard, Ben and Louis. Our should have had a different career producer, Marcel.

And of course, to all our listeners. We're on LinkedIn and X, Dave Chapman and Rob Kernahan. Feel free to follow or connect with us and please get in touch if you have any comments or ideas for the show. And of course, if you haven't already done that, rate and subscribe to our podcast.

See you in another reality next week

#CloudAssessment #CloudCheck #MovetoCloud #HybridCloud #MultiCloud #CloudEngg #CloudOps #DevOps #CloudApps #CloudMgmt #CloudSecurity #Hyperscalers #Native #CloudNative #DCT #DCMigration #DCT #HybridDC #ManagedServices #Enginering #CloudPlatform

About Capgemini

Capgemini is a global business and technology transformation partner, helping organizations to accelerate their dual transition to a digital and sustainable world, while creating tangible impact for enterprises and society. It is a responsible and diverse group of 340,000 team members in more than 50 countries. With its strong over 55-year heritage, Capgemini is trusted by its clients to unlock the value of technology to address the entire breadth of their business needs. It delivers end-to-end services and solutions leveraging strengths from strategy and design to engineering, all fueled by its market leading capabilities in AI, cloud and data, combined with its deep industry expertise and partner ecosystem. The Group reported 2023 global revenues of €22.5 billion.

Get the future you want | www.capgemini.com

Capgemini