EPISODE 1487

[INTRODUCTION]

[00:00:00] ANNOUNCER: Mailchimp is an all-in-one marketing platform for growing businesses, empowering millions of customers around the world to launch, build, and grow their businesses with world-class marketing technology, award-winning customer support, and inspiring content. Eric Muntz is Mailchimp's CTO responsible for the engineering teams that design, implement, and maintain Mailchimp's production infrastructure. He joins the show to take us to the engineering behind Mailchimp's infrastructure.

[INTERVIEW]

[00:00:32] JM: Eric, welcome to the show.

[00:00:33] EM: Thank you. Thanks for having me.

[00:00:35] JM: You are a CTO at Mailchimp, and Mailchimp is, obviously, giant at this point. You've been there for 12 years, so you've seen a lot of the growth. It's quite a long time, and I guess I'd like to just start off by asking about – First of all, Mailchimp, obviously, very resilient service. I've used it many years. I want to know about the biggest outage. I want to know about the biggest problem that you've encountered in your history of working on the platform.

[00:01:05] EM: Wow. Yeah. What a question to start with. Thanks for being a user. Appreciate that. We'd love to hear what you'd like us to get better at and work on. Always want to hear that for sure. Yeah, 12 years has been quite a long time. It's been a great journey.

The biggest outage is what we affectionately call the SSDpocalypse. So we had moved to SSDs I want to say in late 2010 or 2011. The times around them blur pretty heavily, as you can imagine, in super high growth areas. So it was either January of 2011 or January of 2012. I remember it specifically because it started on January 2nd, and my wedding anniversary was January 3rd. So I was intentionally trying to get some time off.

My infrastructure partner, a guy named Joe, is a very, very calm individual and really handles outages super well. He's one of those like sort of like a duck people, right? Like his feet under the water might be really flying. But on the surface, he's always calm. So he sent this one email that was like, "Hey, these are looking really rough. Something's going on with some of the database servers. We're losing some databases. I'm keeping up with it, but just letting y'all know some things are going on."

Then his second message got a little more fearful and was like, "Oh, something really weird is going on. We're losing a lot of machines." Then his final message was just like, "Oh, all hands on deck. I need help. We're losing machines at just an unbelievable clip." Our hard drives were just dying on our database machines, and we were redundant. We had primary pairs, the whole deal, tons of backups, all of that. But they were all dying all at the same time.

When all of your hard drives just die all at the same time, there's not a whole lot of like extra redundancy that's going to do much for you. So we got in. This time, at this point, we were in managed hosting at the time. So we were running on Rackspace and a couple other manage hosts. Over the course of about 14 hours that day, we lost almost – I think it was around 150 SSDs all at about the same time. So that was really rough. I mean, we were seconds, minutes away, where we made a final backup and had to stop the service. That final backup completed just before the final bit of hard drives died.

What we ended up finding out days later, after 72 hours of just nonstop constant work, trying to figure out what was going on with these machines and restoring service to our customers, of course, we did end up losing a couple of hours of data for a small portion of users, so not all of our users but a small portion. What we ended up finding out was that we thought we were on professional grade SSDs. But we were actually on commercial grade SSDs, and there was a firmware bug in them, where after 5,000 hours of usage, they just turned off. Turned off and shut themselves down. Because we installed them at about all the same time, we lost all of those hard drives at about the same time.

We call it SSDpocalypse for quite obvious reasons. It was really gnarly. The team was maybe four or five total people at that time, so it was all hands on deck, a lot of work by a small group of people. The guy who I mentioned, Joe, has a pretty great catchphrase where he says he

almost lost faith in computing over those couple of days. So we were glad to get an official answer on that on what really was happening there. It was crucial. M4, I think, was the name of the drive if I remember it right from 11 or so years ago. That was a big outage. So it was a really, really gnarly problem.

[00:04:46] JM: Great anecdote there. So I guess that brings up an interesting question of picking storage mediums for the vast quantities of data that you have or maybe just talking about vendor selection more broadly. A company like Mailchimp, you're 12 years old. So you're kind of pre-cloud, I believe. So maybe you can talk about vendor selection and how the scale of Mailchimp affects your vendor selection.

[00:05:19] EM: Yeah, yeah. It's a really great journey. So when I started, we were in managed hosting, and we moved around managed hosting providers. So we use shard approach, where a master shard is – If you log into your Mailchimp account, you will see US and then a number of .admin.mailchimp.com, when you're going to the URLs, and that number is what we consider to be a main global user shard that holds about a million users. We've got about 20 of those we have stood up today.

When I started, we were on US1, and it was only US1. So when we first moved to US2, we said, “Let’s try out a new managed hosting provider and see how that goes.” So we started using those global shards to test out managed hosting providers, until we found one that we were really happy with. It was called SoftLayer. What we were most happy with with SoftLayer was they really understood that we just wanted access to manage hardware, and we wanted them off of the machines and to let us provision and go and run with the machines from there. That we were going to be spinning up and down and around machines really quickly.

So we could stand up a full shard of dozens of machines in 24 hours with SoftLayer, and it was phenomenal. It was just phenomenal. The turnaround we got back then, they were super responsive. They did a really great job with their network, which was super important for us, obviously. We saturate 10 gig lines outbound on our heavy email days back then. So we really needed to make sure we had a good partner on that.

We started getting to the point where we were like their biggest customer or one of their biggest customers. Then you run into biggest customer problems, where they don't see the type of cross shard connectivity or cross machine connectivity issues. So we're running into things that are sort of new for them, and that started to get us to a point where we said, “We believe it's time to take this into our own hands.” So from there, we went to colocation, and vendor selection around colocation is also quite important, right? You need to make sure that the folks running your colo environment really understand what you're doing for your business.

Again, you don't want to be the biggest customer, right? So we picked a spot in Atlanta that houses places like Twitter and Facebook and has a hallway long enough that you can almost see the curve of the earth in it as you're looking out. We got really great support from them and really built out our own network and hired a phenomenal network engineering team. Then, obviously, on the procurement side, making sure we could go really fast and getting a vendor who deeply understood how Mailchimp works.

We had a great relationship there. Honestly, here we are today, where everyone's in the cloud, right? A lot of people are like, “Oh, a colocation. Why?” But there are some things that are easily missed in that story, where right around the time we moved from managed hosting to colocation, some nefarious actors on the Internet were going around DDoS in companies at about our size, and they were hitting companies with really heavy amounts of bandwidth. It was really, really rough. If we were a managed hosting, they would have essentially just no routed Mailchimp, and we would have just been hard down for days, weeks. Some of our competitors were down for weeks at a time.

Luckily, we were able to go behind a DDoS mitigation service and be okay. Maybe a funny anecdote, the people doing the attacks were sending emails saying, “We just hit you with a test. We’re going to hit you with something harder soon.” They hit us with something harder and say, “Pay us X dollars, and we will stop attacking you.” Eventually, they sent us an email and said, “We're impressed.” We never responded to a single one of those emails. I wanted to respond, “So are we. Can we just call it quits, please?” But, honestly, without having been in colocation and building that network and being able to really control our own environment, we would have been in really, really rough shape.

Now, I know the cloud providers are in a much stronger position and have better ability to manage all of that. So we are moving to GCP to the Google Cloud Platform and vendor selection there. We looked at which services we really want to adopt and not actually just manage VMs, but really run with cloud native services. We really liked what Google had to offer. We make heavy use of BigQuery. We have six petabytes in BigQuery and are just super satisfied with the performance.

Also, lastly, and my name is monologue here on this. Lastly, one of the big important things for us is that the Google team treats us like partners in a way that I just haven't had with a ton of vendors in this space before, where they provide team members and office hours, and there's oftentimes people on our team don't know who's Google and who's Mailchimp. They really view our success as their success in a way that I think is truly unique when you're working with vendors. So it's been really great so far.

[00:10:08] JM: Have there been any large initiatives to reduce costs that have led to any interesting stories?

[00:10:17] EM: No, nothing that I'd say is really led to any super interesting stories. Obviously, we've had cost reduction along the way, right? Mailchimp, before being acquired by Intuit, was privately held and never taken investment. So we were profitable, and margins mattered. So we had to really pay attention to what we were spending, and we were very efficient with our hardware use.

In a lot of cases, where you may have over provisioned or not made quite the same use of hardware, we oftentimes had vendors expecting our usage to be at a certain level, and it was 10 or 20, 30 times that level because we just made really, really efficient decisions and efficient use of hardware and made sure there were no extras. But no like super fun stories that came up about maybe how we would build technology or anything to reduce cost.

[00:11:12] JM: Today, the platform is so big. I imagine monitoring is an interesting challenge. Tell me how you surface and combat issues that come up across the platform.

[00:11:27] EM: Yeah. It's really interesting. Back in the days, when I first started and when the aforementioned guy, Joe, first started, we didn't have a whole lot of monitoring. We stood up our first NOC wall with TVs that we just ran to the store and grabbed, and Joe sort of mounted to an Ikea table. So we've progressed heavily from there. We are making heavy use of monitoring tools and alerts.

I think, really, the big thing, everyone sort of does sort of the same approach here, where they look at uptime and availability, and then are also looking at reliability, and then starting to look at where bottlenecks are coming in and where machines are slowing down or where alerts are coming. I think the only sort of novel thing that teams can do is how they handle their on-call rotations and how they really look between – Make sure there's a difference between what you might call a war room, where there's like a real honest – There's now an active incident going on and something that's impacting users, versus we're in warning stages, right?

We're in early alert stages and designating who folks are in the on-call rotation for a certain week or a certain day or period of days, and making sure that pre-warnings are coming in and that teams are taking those seriously and looking at those and seeing if they're red herrings or they're just spikes in certain types of usage. Or they are really pre-outage warnings, and teams can start to get ahead of those and look and see into what's going on.

Especially as we are sort of pre-cloud, where we are at the moment in our migration to the cloud, we have statically over provisioned hardware in our shards. So we have to make sure that as usage is really spiking, we get folks in to see what's going on and where those things might be coming from. So really, it's around getting folks to make sure that there's social contracts around releasing. We deploy about 100 times a day, which is pretty good velocity for a team of about 400.

With all of that, we have to make sure that people take the framing of production ready very seriously. So our framing around that is that everything that's deployed is both observable and observed. Meaning that as you are working on your code, and you're getting ready to deploy it, you make sure that you know how to observe how it's working. Then that you will observe it as you are launching it and deploying it, and then making sure that everyone else knows that you're deploying a thing. Great.

If things start to spike, and teams get called in and have to go and see whether this is pre-outage or just something going on, they have something they can cross reference as to what changes were made and where exactly those changes were and how to get in touch with you along the way.

[00:14:17] JM: Do you have any architectural decisions around monitoring data pipelines or log management data pipelines? I find that the monitoring infrastructure often leads to interesting data engineering questions.

[00:14:33] EM: Yeah. From a data engineering standpoint, we are, and hopefully I'm answering your question right, but as you're monitoring and marshaling data around monitoring where that data goes, it's important for teams to pay attention to the final source, right? So like where exactly the data is going and then where it's going to be accessed from there. We use BigQuery as our data warehouse, so we're making sure that teams know that there's a good data taxonomy and that things are tagged properly and that then all of the events of the system and everything that's happening is able to be stitched together and tell the same story.

So that when analytics are looking at it or engineering teams are looking at it and trying to figure out what's going on, that there's good taxonomy and good correlation between the events and the data that's being tossed around. Hopefully, that's what you were getting at? If not, clarify and we can keep going.

[00:15:25] JM: That's reasonable. I was just wondering about if there was any pipelines around, I don't know, like Prometheus monitoring or if you use data dog or just wanting to get a window into how you do monitoring in more detail and how much of the data engineering you defer to underlying vendors.

[00:15:44] EM: I see. Yeah. So we do use Prometheus. I don't know enough to walk you through exactly how it’s configured.

[00:15:50] JM: That's okay. Let's zoom out of this game. Email delivery is, obviously, core to your business. If your email delivery is suffering from things being somehow marked as spam, or there's a variety of ways that the fuzzy nature of what is spam and what is not spam can

really bite you. Can you tell me about the team structure or the infrastructure you've put around email delivery?

[00:16:20] EM: Yeah, for sure. So it's a dark art. No matter how much you've really dug into email delivery, it is essentially a dark art. It's because every ISP is different, and there's no set of standards, right? There's no like do this, follow this, and your email delivery will be great. There's a few standards, and there's a couple of standard bodies. But every ISP sort of chooses how they detect spam and how they monitor and manage or how they do or do not communicate back to ESPs.

We have an email delivery team that's engineers and system analysts, and then work with a bunch of folks in customer service and in other orgs across the company, really monitoring and watching this. Again, heavily monitored, we have built our own MTA. So MTA stands for mail transfer agent. It is the thing that does the sort of final bits of sending and receiving email out on the Internet.

Email sending reputation really starts with IPs, with your IP addresses. So if you're going to send 100,000 emails, and you're going send, say, 30,000 of those to Hotmail or Outlook and 50,000 of those to Gmail, you need to make sure that those IP addresses are things that those ISPs actually understand. So there's a whole process of managing and monitoring those IP addresses.

Then if you are a large enough email sender, so you come to Mailchimp, and you're like, "Look, I'm going to send hundreds of thousands to millions of email on a regular schedule," you will get a dedicated IP address. So the email that goes out for that user is really specific to that user, and we've automated the management and proliferation and handling of all of that if, otherwise, you will be on a shared IP address. So a couple of thousands of your closest Mailchimp friends will be sending email together.

The part of that that is really important for an ESP to manage, and I think we do extraordinarily well, is if you're on a shared IP address, well, one person can literally ruin the fun for all the others, right? So you need to be really careful about what you allow it to go out. It's really interesting and gets really difficult because 10, 15 years ago or even longer, it was really all

about content. But the downside is that, unfortunately, bad actors on the Internet know how to build really great content, so nefarious content tends to look very, very similar to legitimate marketing content.

Managing it from a content standpoint isn't going to get you all the way there. It'll get you a start but not all the way there. What you have to monitor is behavior, things like time it took to upload the content, to send the content, time it took to manage the contact and where the contacts are coming from and all of that and then also knowing a lot about the email addresses. So Mailchimp has records on over four and a half billion unique email addresses. That doesn't, of course, correspond to four and a half unique individuals or four and a half billion unique individuals.

But going back through Mailchimp's history, we can see all of the things that have happened with all those email addresses, whether they're bouncing, whether they're actually being received, when they're being received, how they're being interacted with all of those things. It's easy for us to do some data science modeling around how all of that works and then help us make really great decisions for the rest of the environment. It's a team that doesn't get a whole lot of attention because it's just sort of considered table stakes for ESPs.

I'm extraordinarily proud of the work this very small team that was only about 10 engineers. Managing all of that and monitoring it and watching it all day every day is just – It's a lot of work because the rules change all the time, and the rules are things that you basically just have to reverse engineer because an ISP is not going to tell you, "This is how we detect spam," because then, of course, the spammers will know how to get around it. So it's a lot of work, and it's really important, and I think it's a huge differentiator for Mailchimp.

[00:20:29] JM: Do you guys run your own email servers?

[00:20:33] EM: Yeah, yeah. So our own MTAs, yeah. So we don't for like inbound or for – We don't provide people with – You can't get like a jeffreymyerson@mailchimp.com email address. So we're not an ISP, but we do manage our own MTAs, our own mail transfer agents that handle all of the outbound and feedback loop processing and do some inbound processing for things like, what's it called, Email Beamer and inbound email processing for our transactional service.

[00:21:03] JM: What's the DevOps setup for managing email servers?

[00:21:08] EM: For managing email servers, specifically, it is mostly infrastructure. So when you ask DevOps setup, it is in our collocated environment. Right now, I think it would be pretty difficult to move to the cloud because we're managing tens of thousands of IP addresses, and they need to go specifically to pieces of hardware, right? So making sure our cloud provider is doing that right is going to be a little bit difficult.

So managed in colocation, we've got a bunch of tools around managing that hardware. On the actual MTA itself, there's not a whole lot of like day to day code deploys. There's a system that sits above that that manages all of those IP addresses and manages all of the communication back between Mailchimp in our transactional platform, and those MTAs has all the agents running and is doing all of the mail coordination.

That is really just a typical sort of deployed software platform, right? It's generally software engineered. We are actually a LAMP Stack. So it's running PHP and MySQL and all of that. It's just generally deployed like a standard software project.

[00:22:16] JM: Can you give me an overview of the frontend infrastructure? I'm guessing you're on React at this point, but maybe you could just talk through what front end engineering at Mailchimp looks like and what your input is on that side of things?

[00:22:34] EM: Yeah, for sure. So we are mostly on React. We have been moving to React from dojo, which we adopted fully probably in 2012 or something. So the older version of our editor is fully built on Dojo. We built a bunch of tools around it to make it work better for us. So maybe we called it Mojo because it's Mailchimp Dojo. But we are in the process of finishing migrating from Dojo to React. Yeah. That stack is just sort of a standard React stack, building custom components and making sure that the frontend teams can really fly around.

We've got some just phenomenal frontend engineers who did some really great work to make our antiquated framework work as a single-page app so that we can marshal requests around in ways that are going to really work cleanly for a React environment. So that's at a high level

really where we are today. We have a PHP framework that we use called Avesta. That's homegrown and has been really adopted and adjusted lately to sort of be modernized and work cleanly with React. So, yeah, that's where we are.

[00:23:48] JM: Can you walk me through the life of an email blast being sent on Mailchimp. Kind of how scheduling works and, yeah, maybe you could put it in the context of the larger infrastructure. I mean, I assume you have lots of email blasts that get scheduled throughout the day, how those get executed and cued up.

[00:24:14] JM: Yeah, yeah. So I'll start high level, and you can ask me to dig in more deeply if you want. So, really, at the highest level, Mailchimp has two things that it's doing a ton of. It has application servers that are working with HTTP requests. So customers in the product working, getting things done, clicking through the UI and doing all the things, and then also processing events that are happening with the emails and other materials that we've got out on the Internet, right? So with our webpages or our beacons that are on webpages or any of that.

A lot of people think about Mailchimp as like an email platform and that like sending the email is the big piece. Sending the email is the start, and then the analytics journey is really what Mailchimp's platform is doing. So a lot of that is ingesting HTTP requests from emails or websites or beacons and then running through the event-driven aspect of all of what happens from there, right? So we kick off an automation. We tick up a reporting element, all of those things. That's all, basically, frontend HTTP.

Then the majority of the work really happening is background jobs, right? So we either ingest something or a user takes some action. Then we go to a scheduler and we say, "Hey, schedule this job to be run." Then we've got job processors in the backend picking those jobs up and then running them, right? That's what's really happening. All day, every day, the vast majority of what our compute is responsible for.

Email sending, obviously, is one of those background jobs. So someone says, "Schedule an email for 5:00 PM." It's going to wake up a little bit before 5:00PM and get started processing and get started working on stitching together all of the elements for that email, right? Because with email customization, it's not create one payload and then just send it a whole bunch of

times. It's create one payload that's customizable, and then it gets customizable however many millions of times or anywhere from one to many millions of times for all of the recipients.

That job gets picked up, it starts building a ton of payloads, and then marshaling those payloads out to the aforementioned MTA instances, which then route it to the right machine based on IP and all of those things, and then get it sent out through the Internet. So that's a very high level sort of view of how it all comes together. I'm happy to dig in any of those specific components if you want more info.

[00:26:49] JM: Maybe you could talk about the failure modes that can occur across that series of steps.

[00:26:58] EM: Yeah, lots of them. Well, in 2011, SSDs could die [inaudible 00:27:03]. But, yeah, so failure modes across those steps.

[00:27:07] JM: Again, I'll say like Mailchimp, obviously, extremely resilient and –

[00:27:12] EM: Yeah.

[00:27:13] JM: I have not had a failed email.

[00:27:16] EM: Yeah. Thanks for that. Knock on wood. Yeah. We send a billion a day, so not a lot of failures happening. But when they do happen, one of the things we have to make sure that we do is we have to be really careful about sanitizing the content that our users can provide. So a big failure mode could be that we get some content that is unable to be parsed or unable to be dealt with and stitched together properly.

That happens extremely rarely these days. It was more prevalent 12 years ago, before times when I'd started. Other modes are like we're running on hardware that we own and run. So, obviously, there are some hardware failures. We do rely pretty heavily on third parties as well. So those background servers might be talking to third-party services to get content for emails. If those servers are down or those servers are slow or experiencing high latency, those are things that can start to clog the pipes and make things a little difficult to deal with. We've got some

built-in monitoring and some built-in code that will just sort of pause jobs that they know are relying on third parties or hit eject switches.

One interesting failure mode, not specifically about sending email, is that we deliver web hooks. So if you have your own service and you want to be notified of activity happening within Mailchimp, either with your subscribers on your list or with reports or with emails that are going out, we will send you web hooks, notifying you that those events have happened. A really common failure mode is that someone has a service that's not ready to ingest millions of requests. But then they take an action in Mailchimp that is going to then send out millions of web hooks, and they actually sort of DOS themselves with their Mailchimp account.

So they might like stand up a web hook and have a web server that's not super highly provisioned and then upload a list of two and a half million contacts maybe. It’s configured to send a web hook for every single new subscriber added to the system, which then queues up a ton of background jobs. Then we fire it off to those systems, and those systems collapse. So we have to be pretty careful about that and make sure that when that happens, we don't ruin the fun for everyone as part of the system. It's not something that's super obvious.

Another thing that came up, a failure mode is – This is actually one of US for outages. This wasn't a huge outage, but it was an interesting one. Around that same time, probably 2011, maybe 2012, we just had a random one day some of our app servers were just totally being consumed with requests and falling over. We looked at it, and they were all the same request. A user had accidentally found a way to set up a campaign type where the content for the campaign is retrieved from an external URL and then made that URL the Mailchimp campaign itself somehow.

We just had recursion error. We were just requesting ourselves over and over and over and over and over and over and over again. So that was a fun one to find and try to figure out what in the heck was going on and protect ourselves from that type of outage happening, right? Dear nefarious actors who may be listening to this podcast, we've already covered that. Please don't try to attack us with it. Well, you can, but it won't go anywhere.

[00:30:37] JM: What about detection of bad actors on Mailchimp, either people that are using it to sell illegal items or whatever, do all kinds of illicit things that you can do over email?

[00:30:54] EM: Yeah, yeah. Yeah. I mean, that's the whole thing, right? Because we're a freemium platform and we allow completely free users with up to 2,000 subscribers on their list to make very high use of our platform, we have to make sure that we are protected from the bad actors. So, again, we have an abuse prevention team that's super great and also super small, roughly less than 10. They're just a super phenomenal team. We've got some automated systems that can do content detection and content scanning.

We also, again, have to really look very heavily at behaviors and patterns. There's enough services that you can run through bot detection and bot awareness and all of that. So that'll cut out the bots but then finding the bad people. It's why if you've used – You said, you've used Mailchimp. It's why you have to do things like verify your domain. So 12 years ago, before we did that, people would come in and set up a Mailchimp account with wellsfargo.com as their domain and then send a Wells Fargo email to someone and basically fished them to click a link and go out and connect to their Wells Fargo account.

We have to take all of that extremely seriously and make sure that our trust and safety procedures are really well thought out and really well considered. I don't really like using sort of violent metaphors, but it is an arms race. So we bob, they weave, and we just keep going back and forth. Our phenomenal team is always looking at it and always on the lookout for what's going on and able to make really quick adjustments. That's sort of one of our engineering mantras is to be rapid, flexible, and reliable.

From a rapid standpoint, we need to be able to see what's happening, make changes really quickly, get them deployed really quickly, see how that's going, and then keep adjusting from there, and it's worked out really well. So that and we have just an unbelievable huge dearth of data that we can lean on, and we have great data scientists that help us do propensity and modeling, and help us do prediction of bad behavior based on all of those activities and sort of features and attributes of users as they come in. So it gives us a bit of pre-defense.

[00:33:16] JM: Can you tell me more about how the teams across Mailchimp are structured? Maybe you can walk me through some of the division of labor and management strategy.

[00:33:30] EM: So how the teams are divided. So at the highest level, we have product engineering. We have infrastructure engineering. We have data services. Data services manages the data pipelines, right? So marshaling all the data from its source, be it our MySQL databases or web analytics or wherever else to the data warehouse, so the BigQuery, and then adding governance on top of that, and then providing analytics tools for the rest of the org.

They also do some ML eng work for our data science team. I also have enterprise IT, and there's a really small architecture team. Across that, and then we have department operations. So with 450 people, we need to make sure we're operating quite well. So our PMO org and folks that help with all hands and all of that. The main teams in there when it comes to engineering discipline is the product engineering team, infrastructure, architecture and data services. So we have a chief architect, and that's the entirety of the architecture team.

Then what we do is use a distributed model, where as we break into squads and squads focus on areas of the product, there's a lead engineer in those squads, and that lead engineer joins a community of architects. So they're not actually labeled architects. They do architecture work together. So they will architect and document and design the work they're working on, bring it together with the community of lead engineers who help make decisions, sign off on those decisions, and then go from there.

What we like about that model is that it provides people, who may not have like the most senior title at all times, be able to participate in that process. So it gives our team lots of exposure and lots of growth and really keeps folks together. Then that way, there's sort of not just like one anointed architect who's running around making all the decisions. It's also the people who are implementing the work that are responsible for that.

Then on the product engineering side, we take the approach of mostly generalists. So our stack is common. We're not polyglot. I'm a huge believer in monoglot environments so that there's elasticity across teams. People do specialize a little bit between the backend and the frontend, but people do a lot of full stack work as well. The specialization starts to come in for the areas of

the product that you've spent the most time on. So you might be someone who's worked on reporting or list forms, and that's really where you stick around for a while.

But if you need to help someone on campaign editing for whatever reason, it's all on the same stack. You can jump in quickly. It's just as easy to deploy as anything else you've been working on. It's easy to understand the code base. It all works the same. It's all written in the same language, all those things. So the product engineering team is probably about two-thirds of the org, and it's a large team of generalists who do cross functional work in squads with design and product management and all of those teams.

Then on the infrastructure side, we specialize much more heavily. So there's a security team doing OPSEC, pen testing, endpoint security, all of that, and they are pretty specialized on security. Then same thing sort of all the way down the stack, right? So we have folks who go to data centers and work on that and manage the data center infrastructure, do system administration, manage our puppet configs, all those things. They are really specialized within their areas, including email delivery as part of our infrastructure team. On that side, we've got a lot of specialization. Then within data services, it's sort of like an infrastructure team from that standpoint, lots of specialization around that.

That's generally how we're structured. Then you could sort of break the product apart and into swaths of what the product does. That's how we squat up around what teams are working on. Then from a management philosophy, starting at the highest level, we have an engineering mission statement. That is we give marketers production-ready software designed to help them grow, and we succeed through togetherness, momentum, and pragmatism.

The four words that really pop out of that are production-ready, togetherness, momentum, and pragmatism. That's really the core of how everything is run. Our engineering levels reflect those four terms. So if you're looking to move from a senior engineer to a staff engineer, you're generally examined across those things, as well as what we call focus areas, which are things like industry impact like public speaking or team mentorship. But that's sort of like a choose your own adventure, so you don't have to do public speaking. But you have to do something that makes the team or the environment better. That's our general philosophy.

We also really believe heavily that individual contributors are leaders, just like managers are leaders. So the leadership tracks that go up across each of those roles stay in parallel, and we are really cautious about and really invested in because it shouldn't be the case that to make more money or get more autonomy or authority or mastering your craft, that you have to go into the management route.

People who really want to manage and want to make sure that they are taking care of the business and employees together really should go into management. Folks who don't really want to do that and want to specialize and continue to lead and mentor and help grow engineers, but focus on hands-on keyboard coding and really honing that craft should stay the icy route and still make great money and have a great career. So that's at a high level. Again, happy to dig in any of that.

[00:39:18] JM: Well, as you begin to wind down, maybe you can talk about your engineering responsibilities for the near future. Any replatforming efforts or other big engineering efforts you're working on?

[00:39:31] EM: As you know, we were acquired by Intuit, and big vision there is that Mailchimp and QuickBooks together are just really amazing partners for what we can do for small businesses. I wish I could compare it to other acquisitions. It's the first one I've ever successfully been a part of. But when I think about it, it's hard to find an acquisition where we have two companies that view customers in such a really similar way but have very little product overlap. So we just sort of seamless lead up to each other. There's no like, “Which part of whose product are we going to shut down and keep the winners version of?” It's really just how do we meet right in the middle and combine and unite and provide a ton of power for small business.

Obviously, one of the big things is connecting to Intuit and connect into QuickBooks specifically and getting our data flowing between the two products and starting to make really amazing recommendations across what you can do if you're an existing QuickBooks customer who's using Mailchimp to do some marketing and help grow your business or you're vice versa. Or you're new to both platforms, and you want to come in and figure out how you’re invoicing, and management of that side of your business can really tie into your marketing and help grow your business. That’s super exciting and something we're undertaking now.

Another big thing, we're in the ending stages of sort of liberating our core data architecture from the email address. So as you can imagine, starting as an email marketing platform, if you looked at an ER diagram, it would just be like email and then a big spider out from there. So we have sort of service size our approach to managing audience members for our customers and being able to manage customers who don't have an email address, right? So that we can have folks who just SMS with their customers or just a few website visitors or people who are starting the checkout process for a shopping cart but don't yet have an email address. We can figure out how to connect to them. That's a pretty big replatforming and undertaking.

Then, obviously, as I mentioned, we're moving to the Google Cloud. So the containerization of things and making sure that we're properly adopting GCP services and making sure the system is working around that and that we've got our deployment mechanism working well, and we're able to maintain our velocity across teams with that is a really big undertaking. So those are the big exciting things we've been working on. As well as, as you asked, on the frontend platform, landing the React plane, so really finishing the move from Dojo to React. As I noted, I don't like polyglot environments. So if we've got both Dojo and React code out there, I'm not really comfortable. So as soon as we are 100% React, then we'll be better. Of course, teams will have higher velocity.

[00:42:16] JM: Awesome. Well, Eric, thanks so much for coming on the show. It's been a pleasure talking to you.

[00:42:20] EM: Yeah. Thanks. Thanks for having me. Appreciate it.

[END]