EPISODE 659

[INTRODUCTION]

[0:00:00.3] JM: Stitch Fix is a company that recommends packages of clothing based on a set of preferences that the user defines and updates over time. The software platform of Stitch Fix includes the website, the data engineering infrastructure, there's lots of machine learning, and warehouse software; there's also plenty of other applications internally and externally. Stitch Fix has over 5,000 employees, including a large team of engineers.

Cathy Polinsky is the CTO of Stitch Fix. In today's show, Cathy describes how the infrastructure has changed as Stitch Fix has grown, including the process of moving the platform from Heroku to AWS, and the experience of scaling and refactoring a large monolithic database. Cathy also talked about the management structure, the hiring process and the engineering compensation at Stitch Fix. It's a wide-ranging high-level perspective on engineering at a large growing company that recently went public.

[SPONSOR MESSAGE]

[0:01:09.9] JM DigitalOcean is a reliable, easy-to-use cloud provider. I've used DigitalOcean for years, whenever I want to get an application off the ground quickly. I've always loved the focus on user experience, the great documentation and the simple user interface. More and more, people are finding out about DigitalOcean and realizing that DigitalOcean is perfect for their application workloads.

This year, DigitalOcean is making that even easier with new node types. A $15 flexible droplet that can mix and match different configurations of CPU and RAM to get the perfect amount of resources for your application. There are also CPU-optimized droplets perfect for highly active frontend servers, or CICD workloads.

Running on the cloud can get expensive, which is why DigitalOcean makes it easy to choose the right size instance. The prices on standard instances have gone down too. You can check out all their new deals by going to do.co/sedaily. As a bonus to our listeners, you will get a $100 in credit to use over 60 days. That's a lot of money to experiment with.

You can make a $100 go pretty far on DigitalOcean. You can use the credit for hosting, or infrastructure and that includes load balancers, object storage, DigitalOcean spaces is a great new product that provides object storage, and of course computation. Get your free $100 credit at do.co/sedaily.

Thanks to DigitalOcean for being a sponsor. The co-founder of DigitalOcean Moisey Uretsky was one of the first people I interviewed and his interview was really inspirational for me, so I've always thought of DigitalOcean as a pretty inspirational company. Thank you, DigitalOcean.

[INTERVIEW]

[0:03:16.6] JM: Cathy Polinsky, you are the CTO at Stitch Fix. Welcome to Software Engineering Daily.

[0:03:20.4] CP: Thank you. Glad to be here.

[0:03:22.1] JM: Stitch Fix is a company that gives styling and personal clothing shipments to customers for them to try. Explain how Stitch Fix works at a high level.

[0:03:33.0] CP: Yeah, sure. Stitch Fix is an online personal styling company. The way that it works is that a client comes into our website and they fill out a detailed style profile. We ask questions about your style preference, your fit, your size and even your cost preferences. Then we assign you to a personal stylist. What we do with that data though is we run it through dozens of machine learning algorithms to get very accurate recommendations that we give those stylists. They hand curate a fix just for you, send it to your house, you try it on at home, you pay for what you keep, you send back the rest, we pay for shipping both ways.

[0:04:15.2] JM: The first stage of that process is this customer signup, and then a surveying process. I want to go through some of the data gathering and then we can –

[0:04:23.8] CP: Absolutely. Yeah, it's a big company. Right.

[0:04:26.8] JM: Exactly. Then we can gradually dive into the algorithms and machine learning and infrastructure and so on, but give a perspective for the beginning of that customer lifecycle when you start gathering information in the survey.

[0:04:39.7] CP: Yeah. Ask people to fill out a survey, it's similar to a dating survey actually when we're getting information about different dimensions that really people care about when they're going shopping themselves. If you're like me, you probably do almost all of your shopping online, except for apparel. Apparel is one of the industries where it's really hard to buy clothes online.

The reason is because there's so many different things that you think about when you walk in to a store and probably they're slightly different than your brother, or your best friend, or your coworker. We really have a wide range of things that we're asking people. First it's around fit, so what size are you for shirts, for pants, do things usually run long? Do they usually run wide? We ask about style. What colors do you like? What materials do you like? What preferences you may have? We ask about other stores you may shop at, or things that you care about for quality and what your budget is. We use all of that information to get a really holistic view of what each person is and how they think about shopping.

[0:05:42.6] JM: Then eventually, that results in a set of clothing that is sent to them.

[0:05:49.6] CP: That's correct. Yeah. Because it's really about this how do personalization, how do we get a curated box just for you with things that you like. Fan of paradox of choice, in the sense that more choices are not necessarily better, you want the right choices for you and that's what we provide.

[0:06:07.5] JM: What's the right way to frame this? Can we think of it like a stable matching problem, where the customer has given some set of preferences and then you're doing a matching algorithm between the properties of those preferences and the properties of a set of clothing?

[0:06:27.1] CP: That's a really good question. It's not quite in the same as a stable matching marriage problem, so to speak, in that there are multiple pieces of inventory that can go to many people. It's not a closed system in that way. There is this dual side of the recommendations. There's information that we have about every single item of clothing, plus information that we have about every client. We are trying to get the best match between each and every – for each and every client.

[0:06:57.6] JM: Okay. Let's talk a little bit about the data modelling. You get the customer in the door, they give some set of preferences and you also have a large database of pieces of clothing that are schematized with their own pieces of data. Can you give me a framing for the data platform? Where the data is stored, what kinds of databases you're using to store that data and how you're getting the properties of those pieces of clothing and of those users?

[0:07:27.4] CP: Yeah. We have two sides of our technology platform. One is through applications. Our applications for our engineering systems, our merchandising systems and our styling systems, those are relational databases and store transactional data that we get and receive from clients and statistics from inventory that's coming in and out of our system.

We also have a big data platform that is run by our algorithms team. We stream events to them. They have a lot of different sides of that data architecture, but what they're doing is continually taking events and the data to create match scores for each client and each item pairing that we have. Those get fueled back into our styling applications, so our stylists can see up-to-date information about what's right for each client, as the inventory is changing on a moment-by-moment basis, and it really gives us this ability to have at the moment data for each client.

[0:08:34.9] JM: We've had a couple shows recently where we talked about this process of moving data from the transactional stores to the long-term storage, or the data platform storage. We did a show with Uber for example, where they have their transactional data store, where it's using – I think they're using Mongo, and I think maybe MemSQL for the online transaction processing. When a user is accepting a ride for example, or they're involved in a ride, you want the responses to be quick and the data model and needs to reflect that. The system of data availability needs to reflect that.

Then they have these periodic jobs where the data is pulled from the transactional stores and then put into storage systems that are easier to access for large columnar queries, for example. What's the process of getting the data from the transactional storage systems into the analytic storage systems? Do you have ETL jobs that pull from the transactional store into the analytic store?

[0:09:45.8] CP: It depends on the type of data that we're pulling, but we have a number of different connection points. Some of it is just database snapshots and just a simple ETL process that's going through into the other systems. Some of these systems are throwing events that we have listeners on the data science teams that are running through their Kafka bus. It is a collection of systems that are all working in unison together.

We don't have some of the same real-time problems that Uber does. We do have a little bit more predictability into our systems, and a little bit more time for really having a great experience for styling. It's not this on-the-moment we're sending out a fix every minute after you order, but it is important for us to really think about how we can tighten those times and process all that information reliably.

[0:10:35.0] JM: Okay. When the user has given their set of preferences and you've got this large database of clothing, you need to be able to match some set of clothing to the user. How time-sensitive is that process? Do you need it to run quickly, or can you give it some time, or does it doesn't even matter? Is it a process where you can take advantage of having more time and do more of this?

[0:11:05.2] CP: Yeah. I mean, historically it was a nightly batch job that was running and that was fine for a while. Now we've realized that we want to tighten that time for delivery, and so we've been decreasing the frequency that we do those runs multiple times per day. We can also see this happening in the moment as well. We're continuing to evolve as we want to get faster and faster delivery out.

[0:11:30.2] JM: Right. It's interesting, because it doesn't matter – I guess in the original model, you could just have the user sign up, they take their survey and you say, "Okay, in the morning we're going to send you an e-mail with what your order is going to be," and then – so you take out some of the time sensitivity.

[0:11:45.5] CP: What's amazing is we never send an e-mail out to the client of what we're going to send them. There's no peeking of the system. This I think is just a really interesting story, where Eric Colson who leads our data science team was at Netflix when he was first talking with Katrina our CEO and Founder. At Netflix, they were really thinking about how if – what a personalization company Netflix is. If they really trusted their personalization algorithms, they wouldn't give you lists and lists of movies to watch. They would just show you the top three. Or if they were even better, they would just show one.

At that point if it was just one, they might as well just put a play button there and deliver the exact movie that you wanted when you were opening up the app. He came to meet Katrina and was shocked to hear that she was doing just that. She was sending a fix, a box of items to clients without showing them what was in the box. We trust our recommendations and our personalization so much that we have our own skin in the game. We're paying for shipping both ways for the clients, so we have to get it right every single time. The fact that we can do that with confidence without even having the client see those items before their ship, I think is pretty remarkable.

[0:12:58.7] JM: Definitely. We can talk a little bit more about the feedback loop there and how you can improve that process over time. I'd like to get a little more color on is the inventory and the data model for the inventory. You've got jeans, shirts, dresses, socks, how do you develop the schema for that data and what are the different fields? Are they all human readable fields, or are some of them latent fields that are developed through just feedback loops? Tell me a bit about the data model for the inventory.

[0:13:32.9] CP: Yeah. The data model really maps to what the merchandising team thinks about when they're ordering merchandise. We think about a merchandising hierarchy of characteristics about clothing and how it fits into different categories. A lot of it is key value types of information and about size, about color, about print, but then we augment that with additional information that we get on the ground with when we receive the clothes, so we have a point of measure application that is running our warehouses. When we receive clothing, we measure each item of clothing on these different dimensions; how wide are the shoulders, how wide is it the chest, what about at the waist, how long is the shirt?

What's amazing is we found that immense button-down shirt actually has more points of measure that matter to fit than a woman's shirt, which I find pretty surprising. I think women are pretty concerned about fit and they also have lots of different body shapes, but when it comes to a men's shirt, even how far the first button is down from the collar really matters to a guy. What we've been able to do is collect a lot of different data and then figure out the data that really matters when we're trying to match with each client.

[0:14:46.7] JM: Right. Immense medium. That doesn't really even tell you anything. I mean, you’ve got medium from so many different platforms and companies and medium doesn't mean anything. I can imagine if you have a lot of information about my actual human body dimensions, if I give you those human body dimensions and you have done the work of actually manually taking your own measurements, you can have a much more accurate representation of what shirt will actually fit this body type in a photogenic fashion.

[0:15:24.7] CP: Exactly, exactly. I think that's really amazing as we have this continuous feedback loop for us to get better and better over time. Not just to get the sizes that fit you, but for what brand is that size really relevant. Then what does that mean from a latent sizing for either a brand that we're covering, or from a person that we're seeing. We can use that all through into our recommendations for that person.

[0:15:52.1] JM: Some of these features are much more important than others. The sizing for example, you probably want that to supersede things like does it have stripes or not and does this person like stripes. Do you have any idea of how that data model prioritizes the different features of a piece of clothing?

[0:16:11.2] CP: It all matters. One of the things that we really think about when we're asking questions about a client is is this information that we can use and can we share this information to the stylist? Because the biggest way that we can disappoint a client is if we tell them – if they say, “I don't like red and then we send them a red shirt,” they are going to have this really bad mismatched expectations and disappointed fix through that experience.

Whenever one asks a question and then not, either you use that data in the algorithm, or send that to our stylist for them to be thinking about the recommendations. Yes, fit is extremely important and that's every single item has to fit. If they say that I want plaid and they feel strongly about that, we also want to listen to that. Unless, there's some new type of plaid that you think that they may never have heard of, and may be more likely to try, but we also really are careful about making sure we tell the client when we're doing that and send them a note and saying like, “Hey, I know you mentioned that you didn't like this. Even though I hear you, I think I'd like you to give this a try and to stretch someone out of their comfort zone.” The algorithms help us do that. They help us surface things that someone might like, but we also have to be pretty intentional about how we talk to a client about it.

[SPONSOR MESSAGE]

[0:17:36.8] JM: Leap.ai matches you with high-quality career opportunities. You are more than just your skills and a job description and a resume; these things can't fully capture who you are. Leap.ai looks beyond these details to attempt to match you with just the right opportunities. You can see it for yourself at leap.ai/sedaily.

Searching for a job is frustrating and Leap tries to reduce the job search from an endless amount of hours, days, weeks, to as little as 30 seconds trying to get you matched to a job instantly, by signing up based on your interests, your skills and your passions. Leap works with top companies in the Bay Area, Unicorns and Baby Unicorns; just to name a few; Zoom, Uber, pony.ai, Cloudera, Malwarebytes and Evernote.

With Leap, you are guaranteed high response and interview rates. You can save time by getting direct referrals and guaranteed interviews through Leap.ai. Get matched to jobs based on your interests, your skills and your passions instantly when you sign up. Try it today by going to leap.ai/sedaily. You would also support Software Engineering Daily while looking for a job. Go to leap.ai/sedaily. Thank you to Leap.ai.

[INTERVIEW CONTINUED]

[0:19:15.6] JM: You're the CTO, which means that you have a hand in all of the different technical teams. Can you give me a picture for the org structure of the engineering organization?

[0:19:27.5] CP: Yeah. What's really amazing is that when I started, the vast majority of the team was working on internal technology. We build all of the technology to run the business, whether it's tools for the merchandising organization to buy the right product and the right size, right quantity, tools for the styling agents to be able to – the stylists to be able to style each and every fix. We have five warehouses around the country and we run all of our own software for those warehouses, and also have tools for finance and customer support.

I'd say about 80% of the team was working on technology across each of those departments. Very product-centric engineers, people who really cared about making business impact. If you look historically with the company, we had gone through this period of time where we decided to focus on profitability, really to not worry about going through funding rounds again and owning our own destiny. As we were scaling the company, this was important for us to invest in that technology.

I'd say though when I arrived, I also noticed that we had tackled a lot of the big scale issues that we needed to hit profitability. We were profitable and we hadn't really built a muscle around conversion, or client engagement. The actual website and mobile app hadn't had that much investment in it. Over the last year and a half, we've really been scaling out that side of the business. It went from less than 20% of the organization to almost 50% of the organization now.

We're seeing a ton of momentum there, where we're focusing on first on features for acquiring new customers. When we're asking new profile questions, we want to make sure that it's not confusing and it's not going to decrease the number of client signups. We're doing constant experimentation on that. Then we're also building a lot of engaging features for long-term engagement and retention of our clients.

Then thirdly, we've been launching a lot of new businesses. We started out as women's business. About a year and a half ago, we launched our men's business, then we launched Plus. After that, we launched Extras, which is ability to put in basics into your fix, so bras, underwear, socks, shapewear that you can put into your fix. Then this month, we launched kids. It's been a constant flurry of amazing product launches, the teens been able to work on them.

[0:22:01.3] JM: With those subsequent product launches, I imagine there is some ability to leverage pre-existing infrastructure. There's probably some ability to reuse.

[0:22:12.7] CP: All of it. Yes.

[0:22:13.7] JM: Yeah. Platform engineering, some front-end reuse, some reuse of machine learning systems. Of course, these things were originally built I imagine with – we're building this for in an MVP fashion, and it wasn't originally built to be replicated into other verticals. What was the process of repurposing that infrastructure, or rewriting it, or was it actually built to be modular in the first place?

[0:22:47.1] CP: I think that we have really taken this refactoring mode of figuring out what we can reuse and how we can continually to extend and componentize each areas as we've been scaling and growing to new business lines. One is the style profile was a pretty rigid profile when we first created it. There was a lot of work to modulize a startup profile make it easy to add and remove different questions, but also have different profiles for each vertical, whether it's kids, or men's, or women's, so that we can have a different experience for each of those business lines.

The components are the same, but we still need to get those same questions into the algorithms, the same data platforms exist that also needs to surface up in our application for our stylist, so they can see information about each of their clients. There's a lot of customization, so that we can have a really great profile for men that doesn't look like the woman's profile and ask questions specifically that are going to make sense for men.

[0:23:51.5] JM: Do you have a platform engineering team, or these independent business lines fending for themselves and setting up this infrastructure and getting monitoring and alerting and everything and figuring out the best practices?

[0:24:05.2] CP: When we started, we were on the Heroku platform. That actually got us by for a very long time without really needing a platform engineering team.

[0:24:13.7] JM: I love Heroku.

[0:24:14.9] CP: Yes. Yes. It was really amazing to get us up to scale. After we hit a certain point, we are now going our own way directly on a cloud platform right now in AWS, and we created a platform team before that transition. That platform team has really enabled all the other teams to get economies of scale around some of those engineering things that you talked about, especially around our deployments and monitoring.

[0:24:41.7] JM: Definitely. Tell me more about that.

[0:24:44.1] CP: I mean, the first thing we did is that was the team that really focused on the migration from Heroku to AWS. We really thought about what are the key platform features that we got for free from Heroku? What are the needs that each of the engineering teams are going to want to have? We have one of the most product-centric engineering groups that I've ever worked for in my career. These are people who really want to solve problems.

We have the platform leader there who just reached out to all the different engineering teams and asked, “Okay, what are your biggest needs? What are your biggest pain points?” Every quarter, they've been going through that list and really making sure that – not that we have parity, but that we're focusing on the things that are most important for our teams.

That's been great to see the evolution of that team as we've been scaling it up. It's still small, but I think it's a superpower type of team there. You invest in that and it increases the productivity of everybody else.

[0:25:43.2] JM: Now when I use Heroku, using Heroku versus using AWS for example, this is a bad analogy, but I think of it as like iPhone versus Android, where you have best practices that are rigidly set up in certain ways, which can really help you, but over time it's a little bit too rigid. If you have 25 teams, these teams may have very granularly different continuous delivery processes, so you don't want everybody on the same Heroku continuous delivery model. Maybe you want a much more granularly defined continuous delivery platform that you can present to each team and they can configure however they want, and the same would be true for monitoring and deployment and queuing and all these different things. Is that an accurate representation of what you had to – the things you had to reconfigure when you were going from Heroku to AWS?

[0:26:47.1] CP: Yeah, similar. I mean, the interesting thing about the origin story of the Stitch Fix architecture is that it started as a Ruby on Rails shop and never had a monolithic code base. If you think about all of the components that I talked to you about, tools for merchandise for merchants, tools for stylists, tools for the warehouse, tools for apps, for our website, each of those was standalone applications that had its own deployments. It was very independent.

When you think about a team that's very modern, TDD, continuous integration, continuous delivery team, so what happened was we could focus the migration from Heroku to AWS in a piecemeal way. We could go app-by-app, team-by-team to do that migration. Teams could configure things in different ways as needed based on their needs. We've got the platform team to help with things that are consistent across the board. Because of the creation of this team, I've – it's the only team that I – company that I haven't had to deal with breaking up a monolithic codebase, which is really nice evolution.

[0:28:03.6] JM: Did that platform engineering team, did they just go from team to team around the company and act like consultants?

[0:28:12.0] CP: They did. Yes, exactly, exactly. It did require a lot of work across different teams. For the most part, it was a pretty smooth migration. The teams were really thoughtful in how we migrated and it's all said and done, it didn't. I think it took a couple of corners, but no team was impacted for that long because each of these could be done independently.

[0:28:35.2] JM: That's cool. What about machine learning? If we talk about these different verticals that the company has now set up for kids and Plus and so on, these different business verticals of Stitch Fix, how do you get machine learning in each of those organizations? Do you have a machine learning platform team, or do you have individual machine learning sub-teams within each of those verticals?

[0:29:05.4] CP: Yes. The data science team reports into Eric Colson. We have a chief data algorithms officer. In his organization, he's got a data platform team that is really thinking about tools that are used by all data scientists. Our teams and our platform teams work closely with their platform team to make sure that we have good data integration points and clean handoffs for different technologies.

The organization structure is pretty similar. We've got an engineering team around tools for our merchandising. We've got a data science team around algorithms for our merchants. In there, we've got over 80 data scientists of the company. There's not a single piece of the company that is not impacted by data science. We use data science for everything. It really helps extend and make very intelligent decisions around different parts of our business by having this partnership model of how we work together.

[0:30:05.5] JM: Earlier in their show, you talked a little bit about eventing. I guess the process of queuing up different events that happen on the customer side, or internally on Kafka. Can you tell me more about the eventing architecture and what role Kafka plays in your infrastructure?

[0:30:24.0] CP: That is led by the data platform team. Within the engineering team, we do not have the eventing platform. It's mainly used – the Kafka infrastructure is used within the data platform team for the algorithms. We have eventing systems on the engineering side to communicate between different applications. When it comes to the data science applications, they are able to create the infrastructure that works for them.

[0:30:48.9] JM: Okay. It's more about inter-process communication. Kafka is more like an event bus for communicating between different services. I'm curious about some higher-level concerns; management style concerns. As the organization has scaled, how have communication processes between different teams changed? What has broken? What are the technologies you've introduced that have been useful? Things like Slack, Asana, other tools.

[0:31:19.4] CP: The team is a distributed team. We have a remote-friendly culture. About 50% of the engineers are local here in San Francisco at our headquarters. Then about 50% is remote distributed all around the United States. Nice thing is we don't have to deal with a lot of different time zones, so we do have – are just focusing on these three.

There are a lot of things that we need to focus on to make sure that we have great collaboration and communication across teams. We use Slack and we have from the early days of the company, I think that's been great for collaboration, and even asynchronous communication as people can go back and read up on different channels about what's been going on.

We use Github. I think that that is also great for remote teams, for how we can have asynchronous development and really have visibility into what's going on. Then the other thing that we do is we have a summit every quarter. Twice a year, we have a whole org wide summit, where we fly everyone out to California. It's four days, where we focus on getting everyone together in the same room, aligning on our key objectives, focusing on building relationships and doing continuous learning. Then twice a year, teams have their own summits and they can do this anywhere they want. Sometimes people will fly out to a warehouse, or work with a customer support team, or fly out to do a ski trip somewhere, but it's focusing on really getting the teams together for intense collaboration and partnership, so that they can build their relationships to make it possible to work together when they are remote.

[0:32:57.7] JM: As CTO, how much of your work is about having one-on-ones and setting OKRs and KPIs and these more human-level, cultural and managerial processes that you're putting in place? How much of it is getting in the room with an engineering team and helping them work out how to refactor the deployment platform, for example?

[0:33:25.7] CP: I'd say it's a mix. I'd say this past year has really been about scaling the team. I'd say, when I came onboard, I was not getting paged every night. The architecture was actually a sound building block for us. All the key indicators of how the teams was doing testing, how teams were doing on-call rotations, how we had our – just metrics about availability, about performance all were trending in the right direction. Those are things that I would know to focus on if I felt like they were the most burning needs.

I'd say the biggest burning need that I saw was the team had not kept up pace of growth with the rest the company. We were really understaffed compared to other departments within the company. We more than doubled last year. A lot of my focus last year was on how we scale and hire on team, how we put in good hiring practices, how we create a new career ladder for the individual contributors in addition to managers, how we put in manager training. These people things I think are really important to make sure that you've got the right setup for growth.

I feel like as my role, you just wear the hats and you look at what your team and org needs. Now that we – I feel like we have gotten into a good state on the team health side, I'm looking at what the next big challenges are. A lot of it right now is we have a micro-applications infrastructure, but we're not quite microservices. We still have a monolithic database, and that has had contention points at certain points of our time and we're looking at ways to start breaking that up. That has been a new shift that we're focusing on now.

[0:35:11.3] JM: There's a lot there. I want to revisit some of the stuff on management, scaling, hiring.

[0:35:17.5] CP: Absolutely.

[0:35:18.4] JM: Career ladder. Those are all interesting subjects. Since you mentioned that database, the monolithic database issue, so is that a monolithic database in the sense that you've got all of the clothes in one place, or what exactly is in that monolithic database and where is it running?

[0:35:36.8] CP: I mentioned that we have micro-applications. We've got 80 plus applications that are running for merchandising, tooling, for stylists. We've got dozens of apps that we run in our warehouse, plus we've got our main core website and our iOS application. If you think about each of the schemas that each of those applications need, they were all on a monolithic database.

We had one relational database that served all the needs for all of our applications, to make sense in the early days. There's no sense to have lots of different DBs. There's actually a lot of efficiencies that you can get for how you can join tables and look at data across these different applications when you're running in single DB. The problem though is when you've got hundreds of engineers and different apps running against the same table. You get this contention, so from a simple standpoint just from a number of connections, it can become a bottleneck.

From agility and speed of development, where it gets in a problem is the ability to keep your teams agile. A microservices application, instead of a micro-applications standpoint allows an abstraction layer between the database that all of the applications can use, and it also allows us to start to untangle from the monolithic database into separate smaller databases for these services.

[0:37:01.4] JM: Can you give an example of contention that can occur?

[0:37:04.6] CP: I mean, the one is we were running out of connections. We definitely had to do some work. You can have different replicas to hit that contention. You can add different caching layers, so we did some improvements there that really hit, address the common issues of contention that we were seeing. I think that now, even getting past the points of contention, it's really from a developer productivity standpoint, a reason that we're shifting to the microservices picture.

[0:37:35.8] JM: When you were starting to run out of connections, what were the layers of caching that you put in – This is this is just a big MySQL database, or PostgresSQL?

[0:37:44.6] CP: Yes. MySQL.

[0:37:46.2] JM: MySQL. Then, so what caching infrastructure did you put in front of it?

[0:37:50.5] CP: We just looked at where the highest number of connections were coming from from the different applications. We added some throttling and we added some limits. Then we looked application by application and added some caching layers; for instance, our styling applications, so that it was not having – when it was fetching data about our clients, we could cache that data rather than having as many hits to the DB.

[0:38:18.2] JM: Okay. Do you cache the response to a query – I don't know anything about database caching, so I –

[0:38:23.6] CP: Yeah, exactly. It just depends on what the each of the applications needs, but you can have a caching layer. Therefore, reducing the number of times that it's doing a DB call, rather than doing a hit to your cache.

[0:38:36.4] JM: Not to get too into the weeds, but do you remember what were the super frequent queries that were – that had the caches and returns all the women's jeans, or something?

[0:38:46.6] CP: No. There wasn't any real like, “Aha, gotcha,” that really took all of the work. It was just teams were constantly running reports on what are the longest running queries, what are the most common queries that are running? Okay, how can we block and tackle on this? We had a couple of engineers who were just in addition to the on-call rotation, who were on this database SWAT team of sorts that were picking away against this problem. I think that we were able to make significant problems there as a Band-Aid, but really when you think about it, how do you think and take a step back from that to architect in a different way?

[0:39:26.4] JM: You've mentioned this micro-applications framework. When you say micro-applications, are you talking about –

[0:39:32.3] CP: They’re just apps. They’re just Ruby apps.

[0:39:35.3] JM: Right. Okay, so and would that be women's and kids and the plus and these different business verticals?

[0:39:40.4] CP: No. I mean, one app we have is for our people in our warehouse to pick clothes. They've got a mobile device that they go around and say, “Okay, I'm going to around the warehouse and picking these clothes for these fixes.” That's one app. There's another app that prints out the style cards that gets put in each shipment. There's another app that's running our returns process. These are all applications that are important for different parts of our business, but they can be deployed separately, run separately on different instances. We've got a lot of ability for isolation and speed of development.

[0:40:20.5] JM: Then you're saying and the same is true for the databases that those different services access, or the style card you should be able to break out the requisite data pieces from that huge monolithic database and just set up a database that's devoted specifically to the style cards?

[0:40:39.0] CP: Yeah. I think that we're looking at abstracting things, so that we won't quite have that one-to-one model for every application would have its own DB. The style profile survey, we're having a survey service and that's data that's used a lot of different ways, but there's a primary owner which is the team that's creating the profile. Many teams like the styling team needs to use that data when the stylists come in, so we can abstract that data into its own DB, but it's a higher-level service that can be used by many applications.

[SPONSOR MESSAGE]

[0:41:19.0] JM: Thank you to our sponsor Datadog; a monitoring platform that unites metrics, distributed request traces and logs in one platform. Datadog has released new log management features and trace search and analytics to help you explore and analyze your APM data on the fly. Plus, Watchdog is a new auto detection engine that uses machine learning to alert you to performance anomalies without any setup.

Datadog includes out-of-the-box supports for more than 250 technologies, including PostgresSQL, AWS, Kubernetes and more. With rich dashboards, algorithmic alerts and collaboration tools, Datadog can help your team troubleshoot and optimize modern applications.

Start monitoring with a 14-day trial and Datadog will send you a free t-shirt. You can go to softwareengineeringdaily.com/datadog to get that free t-shirt and to try out Datadog with all its new features. That's softwareengineeringdaily.com/datadog.

[INTERVIEW CONTINUED]

[0:42:35.8] JM: All this refactoring and software developments that's taken place since you've been CTO, has there ever been an issue of infrastructure sprawl, or overspend where you've had to reel in cost structures?

[0:42:48.3] CP: That hasn't been the primary motivation for us. I think that we've been able to be pretty lean for what we're doing. It's really about making the right technology decisions that allows us to move fast and serve our clients.

[0:43:00.8] JM: Has there been any desire to move to – have you done containerized refactoring, or Kubernetes migration, or has that not been a concern for you?

[0:43:10.4] CP: That hasn't been a concern for us yet.

[0:43:11.9] JM: Interesting. Why is that? Because I mean, that's just – so many companies I talk to, that's top of mind for them.

[0:43:20.0] CP: There's more containerization done on our data platform team and the algorithm side, but I think when we're focusing and we've done some work on this on the platform side, it just hasn't been the primary motivation, or focus right now on our teams.

[0:43:33.2] JM: Does it have anything to do with the fact that you have perhaps a lower amount of transactions, but those transactions are much higher significance, as opposed to some company like Uber, where they've got all – they've got tons and tons of smaller transactions and smaller interactions, so maybe the infrastructure just doesn't require as much high utilization, like the containerization thing really gets you higher utilization of your platform?

[0:43:57.9] CP: Absolutely. Yeah, I think that's probably spot on.

[0:44:01.0] JM: Interesting. Revisiting the management stuff; you come in and you realize, we need to ramp up hiring. Of course, the classic phrase that I keep hearing at least right now is hire slow and fire fast. You need to hire slow, because if you hire too fast, then your culture is going to fall apart, you're going to hire substandard people. How do you scale hiring?

[0:44:24.9] CP: First off, we made it everyone's job. We really talked about how it was important for us to focus on scaling the team. The problems that we were seeing were actually related to not having the staffing that we needed. I think it was really getting everyone's buy-in and this was important and this was in order for us to make the impact, we needed everyone to have this as a focus.

Then it really came to consistency of how we looked at hiring, making sure that we're hiring with our values in mind and having a consistent process. We really focus on sharing candidates, so we had a centralized process across every team, so every manager wasn't doing it all on their own. I think that helped us for, "Hey, this person's not quite right fit for this role, but I think they would be really good over here." That allowed us to have a better candidate experience, as well as also move a little faster.

Stitch Fix is a company that really cares about partnership and product-centric engineering. We have a product interview as part of our flow. This is having an engineer talk to a non-technical person and collaborating on a business problem. What I love about this is that we tend to get people who are very inquisitive, who really care passionately about solving problems, but can also talk with non-technical people about technical problems. I feel like we tend to have people who are stronger communicators, who have more higher EQ than most engineering teams that I've ever worked with.

[0:45:56.6] JM: I can imagine, that would be pretty important for Stitch Fix, because I imagine as your – you're CTO, you probably get pulled into a lot of meetings where the business development VP is saying, "We're thinking about launching one of these five different product lines. Which one of these would be easier to engineer?" You can probably help them understand what's going to be the cost of launching this business line from an engineering standpoint, because there's obviously tons of opportunities for Stitch Fix to expand into and it's a really a matter of what's the highest ROI, lowest amount of work thing that we can do right now.

[0:46:34.0] CP: Yeah, absolutely. What's amazing is I don't have to be in every single one of those rooms, because I have such a strong technical team who can be working with our strategy team, or a product team on different aspects of that.

[0:46:47.1] JM: In the hiring realm when you were deciding to scale that up and you said it becomes everybody's job, does that mean every engineer has to be doing some interviews, or they have to calling some people that they know and inviting them to come work at Stitch Fix? What does that look like when you do an all-hands-on-deck for hiring?

[0:47:05.4] CP: I mean, everything. We do an all-hands every other week and we talk about all the open roles and ask people to post on their networks to talk about hiring when they're at different conferences. Then we also have a very large number of people who go through our phone screening, training to really help scale out the number of people doing phone screens. We very much have a large portion of every manager's role, but then they're leaning on the leads of their team, as well as other engineers to fill out the hiring panels.

[0:47:36.8] JM: You also mentioned this career ladder thing earlier. There are some companies for which the career ladder is prioritized, or it's a big deal. I know Amazon has their career ladder and Microsoft has a career ladder, and it could be SDE1, SDE2 and then you become a senior engineer, then you're etc.

There's other companies that don't prioritize this as much. I did a show with Stripe recently and Stripe, it seems like they're really figuring out do they want to have a career ladder, or are there more advantages to just keeping things flat lead to find? You're just a software engineer, or you're an engineering manager, instead of having a – there's some trade-offs to defining that hierarchical structure. How have you resolved that problem at Stitch Fix?

[0:48:28.1] CP: I agree that if you can get away with not having any titles in any of the career ladder, you should try to maintain that as long as you can, because it is complicated as you start adding different titles. It can pull people away from what's really important for their career growth. That being said, we already had a career ladder when I came. What it did was the individual contributor role tapped out at the manager role.

There were no individual contributor roles past a manager for someone to progress. What I was seeing is that we had great architects who were swayed to be directors, in roles that they weren't happy and it wasn't using their best of their skills, but it was what was seen as the right thing to do in order to progress. That is a problem and I think no strong technical organization should have. Every strong technical company that I've worked with has a separate individual contributor career ladder that is on par, if not beyond the manager track, so that you are valuing your top-level individual contributors as much, if not more than your manager roles and you're not encouraging people to move to management for the wrong reasons.

[0:49:47.0] JM: Right. Historically, there was this thing it sounds like, where compensation, the only way to get at a certain point, the only way to get past a certain compensation level was to move into management, which doesn't make sense, because there are individual contributor roles, where you can have just if you're a superstar engineer/architect, you can have just as much leverage business impact –

[0:50:11.1] CP: Not more. Yes.

[0:50:12.9] JM: If not more as a really good VP engineering.

[0:50:16.8] CP: Exactly, exactly. That was a big problem that we needed to fix. We had an early engineer here who was in this director role. When I was having conversations with him, he decided that he wanted to move back into an individual contributor role. That was definitely a good sign in the right direction that we had a model for this for the rest of the teams that they could look up to and see how they could progress. That started as the basis that we were able to use as we were flushing out the rest of the career ladder for technical leaders.

[0:50:48.8] JM: I always wonder how the comp structures are determined for those. Do you go out and find some compensation market research company that helps you figure out how to do this?

[0:51:00.8] CP: Stitch Fix have really unique compensation model, different than any other tech company that I've worked. Stitch Fix plays the same amount for every single role, so if we have two lead engineers in San Francisco, even if they're on different teams, they will make the same amount of money. For anyone in the same level, the same role, they have the same compensation. This is so different than any other place that I've worked. Everywhere else you have a range and it depends on what you may have been making before, or how well you negotiated when you came in. What we wanted to do is take that out of the equation that it shouldn't – if you're doing the same role, you should get the same pay, so equal pay for equal role.

What we do is we do market-based research for every single role, not just for our technical roles, but every single role within Stitch Fix. We have a compensation team who does that and they've got a lot of third parties that they use to get competitive data there, but we really make sure that we've got a competitive role, competitive compensation for each role and then we comp to the role.

[0:52:02.8] JM: Are there bonuses, or extra RSUs, or anything like that, or there's really just no variability?

[0:52:08.4] CP: No variability at all. We expect everyone to be performing in role and the senses that we've got very high performers here, and that we pay – this model works if everyone is performing in role, and it doesn't work if you're not. Therefore, we also encourage managers and train them on how to manage performance and make sure that everybody is operating at their best.

[0:52:32.0] JM: That sounds really awesome and idealistic in some regard. It sounds like it avoids through certain managerial conflicts. In reality, I've seen – I feel like even at the beginner engineer level, even at the intern level, there's a power-law distribution of who actually does the most work and who puts forth the most performance, but I guess you could just adjust for that in saying, “Okay, you accelerate through the career ladder much faster.”

[0:52:59.2] CP: Right, right. Because then you're not playing the same role. Your role is more equivalent to a senior engineer, or just a software engineer. We do have internal – I don't love the idea of having too many titles, and so we’ve tried to reduce the number of titles, but we have internal levels within those titles for different – for step-level progression, so that we can have a trajectory of growth for each individual.

[0:53:27.9] JM: You can maybe accelerate faster through the career trajectory, because of that egalitarian salary notion?

[0:53:36.0] CP: I think it's just based on your role. We're really looking at the role that you're playing on the team and making sure that we have monthly promo meetings right now, so there's no limit in how fast we can get someone to progress. There is an expectation that's not just one project that you have to show consistent delivery at a specific level. It's not that you'll get to promote, promote, promote over every week, but there is the sense though if you are really performing at a higher level and that we compensate you based on that role that you're playing.

[0:54:06.1] JM: Okay. I know we're running out of time here. Tell me, what are the biggest frictions right now for weaving machine learning into an organization and how are those things going to change in the coming years?

[0:54:18.0] CP: I mean, I think that the biggest friction that we see is when we don't have equivalent staffing on both sides. If the data science team has focused more staffing in one area than we have, or the engineering team is focused higher investments, and when you get into contention points where one person is depending – one team is depending on the other team, or blocked on it, then that's where we get some problems.

We're trying to really have more upfront conversations earlier on in the year, as well as each quarter to make sure that we're not going to hit these blocker points. It's harder as a team scales to make sure that you stay in that alignment, because there's always these really interesting opportunities to go after, but we have to make sure that we stay in lockstep so that we set up the teams for success.

[0:55:00.7] JM: You mean lockstep between the data science team and the specific application engineering teams?

[0:55:06.3] CP: Yeah, exactly. I mean, sometimes you can run pretty independently, but there are other projects that really you need to have investments on both sides to get the full benefit from whatever innovation is happening on either other side. Eric and I are focusing on more of that communication to make sure that we are setting up our teams for success in that way and staying in sync.

[0:55:28.7] JM: Cathy Polinsky thanks for coming on Software Engineering Daily. It's been really fun talking to you.

[0:55:31.8] CP: Thank you very much. Really fun talking to you as well.

[END OF INTERVIEW]

[0:55:37.2] JM: If you are building a product for software engineers, or you are hiring software engineers, Software Engineering Daily is accepting sponsorships for 2018. Send me an e-mail jeff@softwareengineeringdaily.com if you're interested.

With 23,000 people listening Monday through Friday and the content being fairly selective for a technical listener, Software Engineering Daily is a great way to reach top engineers. I know that the listeners of Software Engineering Daily are great engineers, because I talk to them all the time. I hear from CTOs, CEOs, Directors of engineering who listen to the show regularly. I also hear about many newer, hungry software engineers who are looking to level up quickly and prove themselves.

To find out more about sponsoring the show, you can send me an e-mail or tell your marketing director to send me an e-mail jeff@softwareengineeringdaily.com. If you're a listener to the show, thank you so much for supporting it through your audienceship. That is quite enough, but if you're interested in taking your support of the show to the next level, then look at sponsoring the show through your company.

Send me an e-mail at jeff@softwareengineeringdaily.com. Thank you.
[END]