EPISODE 727

[INTRODUCTION]

[00:00:00] JM: Robotics, genomics and backend infrastructure, as an investor, it can be difficult to assess the viability of a startup that is on the cutting-edge in any one of these very technical areas. A robotics startup requires a team with an integrated understanding of hardware and software. A genomics company will not only have to develop a successful healthcare product, but will have to bring it to market through regulation. In the world of backend infrastructure, building a business that is differentiated from what giant cloud providers can offer, it's getting harder every day.

Amplify Partners is a venture capital fund with an emphasis on technical investments. The portfolio includes infrastructure companies like Datadog and Gremlin, as well as pharmaceutical and hardware companies. Sunil Dhaliwal is the founder of Amplify Partners and he joins the show to discuss the thesis of Amplify. The investments the Amplify makes are in technical companies, which makes the financing decisions around these companies complex enough to require detailed individualized research.

For example, the economics of a genomics company are going to be very different than a traditional SaaS company. So there's going to be some calculation involved in figuring out how much is it going to take to fund a company like this, and is the work that this company is focused on right now, is it even technically feasible?

So it would seem like the funding decisions across these different categories, like robotics, and genomics, and infrastructure, and SaaS. It would seem like there are no commonalities among these different companies and that you have to look at each of them in their own respective microcosms. But there are commonalities among the companies and commonalities among the founding teams. Sunil lays out a useful rubric for anyone who is looking to learn about venture capital investing, or Angel investing, and in particular, the strategy around building a venture fund out of technical investments.

Before we get started, I want to mention that Software Engineering Daily is looking for a number of roles. You can find these roles at softwareengineeringdaily.com/jobs. We're hiring journalists. We're hiring an entrepreneur in residence. So if you're looking to build a business and you're not quite sure what to build yet, Software Engineering Daily is looking for an entrepreneur in residence. So go to softwareengineeringdaily.com/jobs and check it out if you're looking for a job.

Now let's get on with this episode.

[SPONSOR MESSAGE]

[00:02:48] JM: Every developer uses open source dependencies in every software project. Whether you know it or not, your tools are automatically pulling in thousands of license obligations. You're also pulling in dependencies and potential vulnerabilities. This happens whenever you use a new component or simply run a build.

FOSSA is the tool for managing your open source dependencies at scale. FOSSA automatically scans your dependencies for license violations and vulnerabilities to prevent issues at build time and automate compliance obligations at release. Over 8,000 open source projects and engineering teams at companies like Twitter, Docker and HashiCorp rely on FOSSA daily to manage their open source licenses and dependencies.

Get a free scan by going to go.fossa.com/sedaily. That's F-O-S-S-A, FOSSA. You can also check at the show we did with Kevin Wang, the founder of FOSSA, and hear how FOSSA can reduce your open source risks. If your company is good at developing software, then it should also be good at managing open source. Check out go.fossa.com/sedaily and learn how FOSSA can improve your codebase.

[INTERVIEW]

[00:04:18] JM: Sunil Dhaliwal, you are the founder of Amplify Partners. Welcome to Software Engineering Daily.

[00:04:22] SD: Great to be here, Jeff.

[00:04:24] JM: Amplify emphasizes investments in technical areas, and this doesn't just include software companies. It includes robotics, and genomics, and healthcare. I wanted to start with those areas. So when I look through Crunchbase or I look through your portfolio companies, I see these robotics and healthcare companies, but these technologies are not coming into my everyday life quite yet. Why not? How long until we start to have robotics impact our everyday life or start to be impacted by genomics technologies?

[00:04:58] SD: I'd say they are in your everyday life, but you just can't see it yet. The interesting thing about both of those areas is how they're going to show up in consumer's lives comes first behind the scenes and secondly on main stage, and I'll tell you what I mean by that. So when you think about how that Amazon order that you just probably clicked on maybe 20 minutes before this show started, because we're all ordering something from Amazon, how that actually got picked and placed at a warehouse and then on to a truck and then from a truck to a plane, the plane to you, or whatever the ultimate food chain was. That was automated. There was a lot of robotics that was part of that, but you didn't see it.

In the same way when you think about genomics, or healthcare, or life sciences, what I would argue is a lot of the work that's actually going into figuring out how new molecules get made and turned into effective therapies, that's happening through the application of ML and AI and computation and it is affecting your life, or at least it's really kind of starting to. But it isn't something that shows up in your day-to-day. A little bit different than when you get a robot driving you around or a robot kind of serving you a drink, but I would tell you, it's coming and it's all coming very quickly.

[00:06:06] JM: Can you give some examples on the health side or on the drug discovery side where it is actually impacting the supply chain today or is that stuff still a little nascent?

[00:06:15] SD: So it depends on how you think of the supply chain. Drugs take a long time to develop. So when you think about the drugs that are probably on shelves and are coming to you right now, I would say no. It's probably a little bit behind that. But when you look at the pipelines of most major pharmaceutical companies or biotech startups, I would argue that those things

are as much about software development as they are about bench science research that's ever been in the past. Those things are working through the pipeline. It's hard to put an exact number on where they are in terms of actually getting approvals, but those are coming.

The first things that you're going to see out of drug discovery and out of life sciences to impact you are actually going to be existing drugs but have already been approved and they are going to get repurposed in a designation a lot of people refer to as off label. So I'll make one up totally. If you have a drug for hypertension and it's approved, the FDA said it's safe and it's efficacious for treating high blood pressure, that's great. Someone has a patent on it and they can use it for so many years with enjoying patent protection.

Role forward, when that patent comes off, all of a sudden the value of that drug goes way down. However, if you find out that that drug can also affect pathways not just for hypertension, but it could also do something to, say, ease some treatment in diabetes or some sort of immunological issue that you're having elsewhere in your body. If the FDA already said it's safe and they already know things about dosing, the timeline to get that to market is really short, and it also gets a broader set of intellectual property coverage for the new application. That's huge found money.

So when you think about how people actually go about repurposing and retargeting those existing approved therapies, ML and AI actually has a lot to do with that in terms of high throughput testing and doing kind of computational looks at all the different variants of places where these drugs can be effective against other treatments. This is happening a lot in oncology where people are looking for things that already work on existing pathways already approved and trying to figure out how they could get into market quickly for people who have cancer. It sounds really – We're talking a lot about healthcare and biotech stuff, but funny enough, at the core, these are really suffering computation problems, which is kind of what makes it really cool.

[00:08:33] JM: Is it simulation or is it just looking over patient records and making intelligent judgments based off that? What's the software side of finding those adjunct use cases for some drug that already has a proven use case?

[00:08:48] SD: There's two. There's more than two, but the two that I would focus on are some things to do with simulation. If you know that a particular drug effects of certain biological pathway, and that's where it actually intervenes, you can look for other diseases that are impacted by similar pathways and you can narrow down the scope of saying, "I've got this hammer. What other nails look like this first nail that I just hit?" to use a really crude analogy, but it's true.

The second way in which you can use computation is this high throughput screening and repetition. A lot of stuff that has normally happened in vitro, test tubes, labs, folks in white lab coats working on benches with pipettes and beakers and whatnot, that is the application of a molecule or a compound to a living biological sample and see how something goes. You just try and to do that as many times over.

A lot of that stuff actually is now some things that you can start to simulate. You can do some simulations computationally to say, "How would this work? There's 20 different pathways, 20 different your cancers, and let's take some guesses as to where we think a better approach might come from." So think of this is a little bit of like making a big haystack, a small haystack, and then makes it easier to find the needle.

Some of it is actually happening in high throughput biological screening where people are developing tools to actually take a living tissue sample or a living cancer sample. May it's even yours if you've had a biopsy and it gets pulled out, to replicate that hundreds or thousands of times over and then say, "Great! I now have a thousand different copies of," God forbid, "Jeff's specific cancer, and we're going to try 1,000 different compounds against it and literally just watch and see what happens." All of a sudden if we start finding a heat map of like, "Things are responding over here." That's really good lead gen for researchers and scientists.

[00:10:44] JM: We saw AWS enable people to stand up startups for SaaS companies pretty easily or mobile backend. What has been the impact of cloud computing on these kinds of applications on the biotech side?

[00:11:01] SD: It's huge. If reinvents coming up a month from when we're recording this podcast, the very first day, they actually have a whole track that they run at AWS now, which is

just all computational biology. It's amazing, the group of people that get there and what they're showing off that they're actually doing.

I would argue the most innovative stuff that's happening within pharmas and biotech's right now is actually coming out of these computation teams, and their data teams, and people who are almost data scientists and software engineers first, and biologists second. It's really, really powerful.

[00:11:37] JM: How do you vet these kinds of deals? If you're looking at a genomics company or a drug discovery company, how does the vetting process vary from a software company?

[00:11:47] SD: Yeah, it's a good question. So first thing I would say is, at its core, Amplify isn't investing in drug discovery companies or genomics companies at their core. What we have really focused on is these companies that are unabashedly saying like, "We're software companies and data companies, but what we're really good at is understanding these particular applications in these markets." So they're kind of in between. They're kind of these hybrids.

It's funny you asked me that question. We go out at the same way that we go at a lot of different markets. Very little upfront and we learn through a handful of very small deals and we get exposure and we make mistakes and we risk a little bit less money, and then we learn a lot. We create networks of researchers, and data scientists, and co-investors and other people that can teach us things, and we start to create our own mental models and frameworks of what's interesting and what's not. Then we start making bets. We learn as we go.

I've never had a new market where we know everything going in. Frankly, I've never had an old market where we feel like we know everything anyway. But it's definitely a process, but that is – I think you can make the same case for how people have gotten into robotics and how people get into autonomy. For us, there're a lot of similarities. It's the same type of technical founder. It's the same type of subject matter expert. It's the same problems that you have to work through in building and scaling an early stage company. That's really what Amplify has done time and time again regardless of the market or the end market, is, "Hey, what is it take to get a very smart domain expert technical founder and make them a fantastic company builder and ultimately

executive?" That journey, funny off, is actually really similar regardless of the market that you're looking at.

[00:13:37] JM: We've been doing a lot of shows about Kubernetes both on the business side of things and on the open source side of things and on how does it impact your business, whether you are a Kubernetes business or whether you are just a startup that has infrastructure to run. How has Kubernetes change your perspective on software investments?

[00:13:57] SD: The more things change, the more things stay the same. So we're doing this podcast on the – Is it the 5th or the 6th of November?

[00:14:04] JM: Yeah.

[00:14:06] SD: Just this morning, it was announced that VMware is buying Heptio. The more things change, the more things stay the same. Every single person out there went from physical servers to virtual machines and then they realized that they got locked in on virtual machines to VMware and then there is containers that came about and now people are looking at container orchestration and how you're actually going to build this next generation DCOS. They're all really good evolutions.

Your question was how do we look at Kubernetes and what do we think about Kubernetes? I think Kubernetes is the powerful tool and it's the flavor of the moment as tools go, which is the more things change, the more things stay the same. It will not be the end-all be-all. Just there's does nothing that's Docker for making – Organizing containers and containerizing workloads, nor VMware was the end-all be-all, because workloads are going to change.

Right now, what Kubernetes is, is a powerful abstraction that lets people make their applications very, very portable to the infrastructure that they want and gives them a heck of a lot of flexibility in building and playing workloads. The fact that it's taken off so quickly is a pretty good testament to the fact that that's a real problem and it hasn't been solved and it exists more and more as people start looking at cloud and ultimately a multi-cloud world.

[00:15:27] JM: Now, they were infrastructure management tools before Kubernetes, there were OpenStack and Cloud Foundry. What lessons do you take away from those previous ecosystems that map on to the Kubernetes ecosystem, or maybe that the Kubernetes ecosystem has corrected for?

[00:15:47] SD: I'd flip it around. I think what a lot of people get excited about with every one of these new trends is what corrections they're making to the sins of the past. Vendor lock in is a problem. Portability is a problem. New architecture is a problem. You get the new tools that are going to pop up and solve the, which can be really interesting to see in today's acquisition of Heptio by VMware is going to be a really – I think it will be interesting on this dimension, is for any of these compelling new technologies to get mainstream, to run mainstream workloads at scale and be the tools and the platforms that enterprise and large-scale customers depend on.

There's a whole other set of crafty ugly things that you've got to do to make that work. That's where, funny enough, I actually think Kubernetes is going to learn from open stack and learn from OpenShift and others, where what they realize that those products actually do is they give customers frameworks and frameworks that they can rely on and support organizations and integration teams and all the hard work to take the new thing and make it work with their old stuff. It's kind of a flip on the question you asked. It's going to be interesting as these projects emerge. I think we're all going to see them take on – As these projects mature, we're going to see them take on a lot more of the nature of these previous things than people would hope.

[SPONSOR MESSAGE]

[00:17:21] JM: Simple Contacts lets you renew your contact lens prescription online easily. For many years I have ordered contact lenses on websites with low quality user experience. I have ordered contact manually from my optometrist, which requires me physically driving to the optometrist's office to pick them up. It doesn't make any sense.

Contact lenses are expensive. It's a high-margin product that is cheap to produce. Why is it so hard to order them over the internet? Well, now it's actually easy to order contact lenses on the internet. Go to simplecontacts.com/sedaily and use promo code SEDAILY to get $20 off of your contacts. I used Simple Contacts and I was amazed that it actually works. The website is pretty

good. I uploaded my prescription. I ordered my lenses and then they arrived quickly in a box. If you have contact lenses, there's no reason not to try it out and save $20 on your next batch of lenses. Why not? Go to simplecontacts.com/sedaily and use the promo code SEDAILY.

Thank you to Simple Contacts for being a sponsor of Software Engineering Daily and for making a product that I'm just getting use going forward. It's pretty much the way that I would want to order contacts, and it took a really long time to finally get here, but I'm glad it is here. So go to simplecontacts.com/sedaily if you use contact lenses.

[INTERVIEW CONTINUED]

[00:19:05] JM: I've gone to a couple KubeCons and I'm going –

[00:19:08] SD: Going again this year?

[00:19:09] JM: I'm going to the next two actually, yeah. I always like walking around the Expo Hall, because you kind of get a feel for what people are buying, what they're not buying.

[00:19:22] SD: How many people are telling the same message as everyone else.

[00:19:25] JM: Yes. How many people have different perspectives on the world, and they'll present like, "You should buy our view, our vision of how the world is can unfold in the next five years versus Red Hat's vision, or versus Microsoft's vision."

When I talk to some of the vendors there or when I talk to some of the enterprises at these conferences, it seems like a lot of the buying patterns are – They are driven by this aversion to lock-in. It's a fear of lock-in. Rather than a focus on opportunity, which is surprising to me, because if you're like an insurance company, or agriculture technology company, or an oil refinery, or a bank, even if you're locked in to some infrastructure provider and even if the infrastructure provider were to ramp up the cost, they're going to have to pay year-over-year like by two X and you have to pay twice as much.

It's still so de minimis compared to your overall business. I just don't understand why they're so concerned about lock-in when – Why wouldn't they just be more concerned about the other competitors in the world, like all the startups? If you're a bank, you should be worried about challenger banks. You shouldn't be worried about, "Should I be on Kubernetes or should I go all in on AWS and be entirely serverless?" Does this just seem like complete madness to you ever?

[00:20:45] SD: To a degree, and I think there're people who over rotate on this stuff all the time. The smartest arguments against lock-in that I've ever seen actually have less to do with cost and more to do with exactly what you're talking about. The smart IT buyers know that what they need is flexibility, because they've got big existing battleships that they need to transition and turn to go and take on new business challenges. I think the lock-in that they worry about the most is once we get so wed to a particular vendor, we are wed to their view of innovation and a roadmap that we lose the ability to decide and define our IT infrastructure and architecture to do what we needed to do.

I think that smart people worry about lock-in for that reason. A lot of the people that have been around in the 80s and the 90s and the early 2000s, lock-in to them means like the Oracle compliance team coming and auditing you for how many licenses you've got, and it's just like a dude in a suit with a vacuum cleaner coming for your budget. What they saw around windows, where there was just very kind of clear calcification of the platform and then everyone got stuck into on top of that, like they had to build to this, they had to believe in client/server. This got stuck.

So they take those lessons and they want to expand on them or they want to take them forward to the current era. Some people still get caught in their underwear on costs, like exactly like you're talking about. They just miss the forest from the trees. But I think the smart ones understand that if they get really locked in, the big downside to them is about stifling their own innovation and how their IT team can actually meet the business need.

[00:22:25] JM: But you can get locked in to the wrong open source technology. People get locked in to OpenStack and the community maybe doesn't become as thriving as they – I think OpenStack still is a really thriving community, but I think there's probably a lot of people who – I don't know, how many, maybe you could tell me. But there are people who went into OpenStack

and then now they're saying, “Oh, we’ve got all these legacy OpenStack and we wish we were managing it through Kubernetes.”

I mean, you can make a bet on the wrong technology, whether it's open source or whether it's cloud. You should be judging AWS in that overall environment rather than saying, “Okay, AWS could be like Microsoft, or it could be like Oracle.” You can even get locked in to open source projects just as much as you can get locked in to some vendor.”

[00:23:12] SD: Yeah, totally. I think where this starts to set people back is if you worry about lock-in to the detriment of actually moving forward. If you are so worried about lock-in that it prevented you from – You’re so worried about Amazon lock-in that you didn't move to the cloud, then you're doing yourself a disservice.

Rather than saying, “Great. We’re going to move ahead and it's incumbent upon us to figure out how we keep our options open or how we develop Google or Azure or whomever as our second source provider, or we focus them on cloud interoperability.” I think that's the prudent stuff to think about the long term, so you don't wake up one day and go, “We’ve traded one set of problems for another set of problems.” I think that's the thing that people are always trying to get out ahead of.

At this point, 2018, I think most of the people have caught on to not letting perfect be the enemy of good. It was a great example. J.P. Morgan, they came to the Valley and did a little technology roadshow for a lot of investors and startups and they did a great contrast of –

[00:24:13] JM: Like the stuff that they’re actually implementing, the stuff that they’re building?

[00:24:16] SD: Yeah. They talk about their problems and what they’re implanting and what they’re building and they just kind of do a dog and pony show to build connections out to the Valley. But one of the interesting things they did, they said, “Hey, we’ve been doing this for six years, and six years ago, let’s put up the slides that we talked about then and now what we’re talking about today. One of the things six years ago was public cloud ain’t for us. We need a private solution. Regulation isn’t going to allow it. Security isn’t going to allow it. There're just too many reasons why like this Amazon thing was cool for you guys.”

Now you fast-forward to 2018, to 2018 slides. It's, "We're all in on public. Yeah, we want some things to work so that we don't get stuck with these problems over time." Think about that is an example of if they had – I'm not saying that they just came to the conclusion that they're all in in public in 2018. They did that a few years ago. But think about if they made that decision in 2012 and they have the foresight to go, "This is where we're going to go." Think about how far ahead their business is from an IT point of view if they didn't drag their heels. That's the type of stuff like you are pointing out that I always freak out about when I hear people get wrapped around the axle on whether it's vendor lock-in, or it's not secure enough, or it's this. There are ways to adopt things over a period of time rather than just putting up walls and saying, "That'll never happen here." Those are the people I think are shooting themselves in the foot.

[00:25:35] JM: Yeah. They could've been like Netflix, or I guess they – Or like Monzo or something, like one of these newer banks that was built entirely on the cloud and they have a super small workforce and they're super-efficient. Yeah. The J.P. Morgan thing – So what are they struggling with? Are they in microservices and continues deployment and do they have a data platform? Are they doing machine learning? What kinds of things are they struggling with?

[00:26:03] SD: That's a great question. I wish I had them here to give you the direct answers. I'm not going to try and give it. I'd give you a laundry list, but that might be a great group of people to get on to your next podcast. I know who to connect you with, because they're very open about telling people where they want help and what they want solutions with.

One of things I'd bring you back to though that is very J.P. Morgan, for example, specific, but it's also very generalizable, is let's go back to Kubernetes for our example. Kubernetes is super easy to think about when you're starting with a white sheet of paper, the proverbial greenfield deployment. I have no existing opinions about what my application needs, what my infrastructure is, what my workforce can do. It's beautiful to look at the brand-new thing.

Now think about J.P. Morgan that measures their COBOL programmers in the thousands, thousands of COBOL programmers, billions of dollars of annual IT spend, applications and infrastructure that literally spans decades, mainframes. I believe, and I could be have this

wrong, but I believe that they increased – Mainframe business at IBM was up last year, a year ago. It actually grew. A lot of these things don't go away.

Now you think about them and what problems they have, and this is what a lot of enterprise problems — a lot of enterprise customers have as problems, is they don't get to think about the world with a white sheet of paper and a greenfield deployment. Every technology or solution or idea for innovation has to be viewed in the context of where are we coming from. I think that a lot of those really crafty problems that large organizations like that have to deal with are entirely about the evolution of a backward compatibility, about the readiness of a workforce, about legacy applications. You're not going to convince any of those people that it makes sense to completely rewrite and re-platform a certain application right up until the point where it's absolutely clear that they got to take that on. This is not going to be in their nature to do that. It's not worth it. Risk to the business is not worth it. It's really cool for me to always hear what sort of legacy encumbrances these guys have to deal with.

[00:28:14] JM: When I think about some of the trends that I've seen in the vendor enterprise interaction, at least in light of the chronology of the show. So I started the show kind of like right after, I guess, Cloudera had kind of gotten going and the other Hadoop vendors had gotten going. I think I started my podcast around the same time the devops idea was starting to really gain a lot of momentum.

My sense is that there had been a lot of – I guess there'd been probably a lot of enterprise spend on Cloud Foundry and other modernization stuff. But it didn't really get articulated as well as when the devops movement really started to convince enterprises, "Okay. Here is a roadmap for modernizing your deployment process and for building out new services and doing multi-cloud potentially, doing testing more efficiently."

I think of that as kind of like a defense of posturing, because it's like, "Let's take our existing infrastructure and modernize it." One thing I'm trying to understand is when that turns into a more offense posturing and you start to look at, "Okay, I'm an insurance company. We fixed our deployment process. Our website doesn't go down anymore. We've got like decades of data about insurance. Let's figure out how to turn that insurance data into money, into an API for

people, or into value," and this is the whole like data platform thing or being able to build machine learning applications on top of that.

How is that happening? Is that already happening, or are we kind of waiting for these enterprises to just get a handle on their development process before they start saying, "Okay, we've actually got this giant data lake. We can start to build machine learning models around it." I'm just trying to understand how these legacy enterprises are approaching their gigantic data advantage.

[00:30:13] SD: The good ones are actually not focusing on, "Let me get all the data perfect, and then I'm going to think about innovating." The really good ones start with, "Where are the problems in the market? How do we approach younger consumers? How do we give people experiences that are more in tune with the ways that they want to interact with their vendors, or the people who sell them services, or their technology and rethink from that level?"

You see this from time to time in applications that come out whether it's – I'm just going to pick a few, because we're talking about banking. Bank of America has voice interaction and voice interactive assistance now available that you can work with to do simple banking things when you're online with them. There's a lot tighter mobile integration between what your smartphone can do when you walk up to an ATM. It's not unique to them, but there are a number of people who are doing similar things like that.

I promise you what Bank of America didn't do is, say, "Let's get our house in order and modernize this pipeline and start exposing this kickass new platform we built for these new services." That started because a business unit and a product team said, "Here's a problem. Let's go solve that." Then they probably ran into a whole bunch of infrastructure [inaudible 00:31:31] and you had people make really hard decisions, like where they're going to go? We're either going to build on the services and the platforms that are available to us from IT and legacy, or we're just going to shoot that and start again.

Fidelity went through a big transition like this where they tried to be very –This is Fidelity, the big mutual fund company in Boston. They had a click to computer initiative, which is probably about eight-years-old. Very kind of coincident with when devops was really starting to rip through the

enterprise. Where what they wanted to do was create this central IT organization, create this really powerful substrate that everybody else in the organization would use this modernized platform of services that they could expose to all their business units. That was really helpful to them to just kind of turn things on and get ahead of stuff.

I guarantee you it didn't solve all of their legacy problems, but it just gave them a point where they could go, “All right, as of here, when you're trying to do Agile, when you're trying to do more continuous delivery, when you're trying to build new applications and get them rolled out to customers quickly, here's the set of services that we, as IT, can expose to you.” You saw something like that there. So that was a kind of a middle of the ground approach.

I'm sure there are some people who are doing the thing that you were talking about, which is like, “Let's get the data lake ready. Let's get a CICD pipeline built. Let's get our entire devops/ agile methodology in place, train everybody up and then let them lose on the world.” My guess is a lot of people had done that. They’re still hoping and praying that sooner or later they're going to get services out the door. It's just very hard to run it that way.

[00:33:07] JM: I’ve talked to a lot of companies that are trying to build this kind of data platform tshat they can sell to an enterprise. Companies like Dremio, and Prisma, Periscope Data. What do you think of this sector? Is this something that enterprises want? Do they want an all-in-one data platform provider to help – I guess that's kind of what Cloudera and MapR and Hortonworks did for a while. What do you think is going to happen in the data platform world?

[00:33:39] SD: I think the Cloudera, Hartonworks example is a pretty compelling one. It’s pretty instructive, because there was a lot of stuff that they did that was very opinionated on technology. Do it our way and then we’re going to solve all your problems. Pretty soon they realized, “Well, everyone doesn’t want to do it exactly our way, because it isn't about Hadoop. Maybe it's about Spark, or maybe it's not just about Hadoop or Spark, it’s about some other tool. Maybe not everyone is going to put everything in the data lake the way that we would specify it. Maybe it’s going to look different in these ways.”

Soon, these companies got out of the technology business and got into the solutions business. So the lesson that I would take away from that is it's really hard to be completely opinionated

on, “Here is the one ring that's going to rule them all. Organize around us, and this by the way could be very instructive for our Kuberntes conversation.” IT’s very hard to say, “We found it. This is the thing. At this point in time, we’ve got it. Everyone's going to focus on this. If you just change all your workflows and your tooling and your people around this technology, it’s going to solve all your problems.”

Frankly, vendors are really good at selling the, like that's how lots of startups sell themselves. But in reality, I think in reality the bigger the problem, the bigger the scope of the problem. Whether it's like datacenter operating system, or cloud orchestration, or massive unlocking of enterprise data. The bigger you go, the more custom the problems are to your organization, the less likely it is that there's really one ring that will rule them all. Which you instead end up with is those vendors find themselves either becoming a little bit more heterogeneous than they thought on the way in in terms of how they have to solve someone's problems and how they have to deliver solutions to those customers who now look to them as key vendors, or they do with some smaller vendors do, which is they do start cleaving off the world. We are not your end-all be-all for everything, but we are your vendor for this particular piece of your infrastructure.

You want analytics pulled out of all your web-facing applications? Great. We can suck in data from these NoSQL data stores, from these web clicks, from these APIs and we’ll give you this searchable, rehashable real-time event store that you can then spit out and give analytics and insights into all sorts of stuff. But we’re going to limit it to just these integrations and just these technologies. So they find themselves narrowing what they can touch as opposed to going really broad.

I think you end up going one of those two directions, but rarely, rarely do you show up with, “Here’s our really opinionated view of the future. Everybody do it our way. When you do it our way, magic is going to happen afterwards.”

[SPONSOR MESSAGE]

[00:36:29] JM: Your audience is most likely global. Your customers are everywhere. They’re in different countries speaking different languages. For your product or service to reach these new

markets, you'll need a reliable solution to localize your digital content quickly. Transifex is a SaaS based localization and translation platform that easily integrates with your Agile development process.

Your software, your websites, your games, apps, video subtitles and more can all be translated with Transifex. You can use Transifex with in-house translation teams, language service providers. You can even crowd source your translations. If you're a developer who is ready to reach a global audience, check out Transifex. You can visit transifex.com/sedaily and sign up for a free 15-day trial.

With Transifex, source content and translations are automatically synced to a global content repository that's accessible at any time. Translators work on live content within the development cycle, eliminating the need for freezes or batched translations. Whether you are translating a website, a game, a mobile app or even video subtitles, Transifex gives developers the powerful tools needed to manage the software localization process.

Sign up for a free 15 day trial and support Software Engineering Daily by going to transifex.com/sedaily. That's transifex.com/sedaily.

[INTERVIEW CONTINUED]

[00:38:18] JM: You guys are in Gremlin. We've done a number of shows with Gremlin, and that company has a pretty big head start on – I don't know if anybody else is doing 00:41:14 as a service. I could see them having a services business. They could do a services business. They could probably do it today. They probably have some services kind of thing where they interact with customers on a regular basis. But if you double down on that, you could build this revenue stream that's kind of a distraction from the actual SaaS business. When you're talking to the Gremlin team, how are they thinking about or how do you think about the tension there between the services kind of business and the really appealing SaaS API business?

[00:39:04] SD: Yeah. We're really excited about Gremlin. Those guys are, as you pointed out, way out in front on the whole Chaos Engineering concept both from the expertise in the

background of the team and also in terms of just what they're allowing their customers to go do and really bringing this discipline to the forefront.

Not to kind of over rotate on exactly what Gremlin will do or won't do, but what I generally find happening here is you do have this choice, “Am I going to be in the services business or am I going to double down on building killer technology?”

In the early days for markets that are really, really big, we tend to find the companies double down on building really killer technology for one reason, and it may not be obvious, and that is that services businesses tend to be a crutch for crappy product building. You can avoid a lot of really hard product engineering stuff by putting a consultant against the problem and sending them out in the field with some custom cod and some crafting and crafting. Flip that around and say, “How do I actually build a product that's good enough where I don't have to put a bunch of humans around it for my customer to see value?” and all of a sudden the complexity goes way up of what you’ve actually got to build.

But having said that, when you get that right, man, those are really powerful products. The ones, the beautiful ones that seems simple on the outside with lots of complexity hidden underneath that can adapt to tons of different environments and tons of different user workflows. It's that combination of simple on top and powerful underneath that happens, I think, right at that moment where you don't jump down the services rathole and instead double down on building a kickass product.

I think a lot of that shows up in what Gremlin does, but Gremlin’s also in the business of making their customer successful. I know that they will do what they have to do to make sure their customers are successful. But I would also tell you that I think the beauty of that product and other great products really comes down to how well people get out ahead of those things and not let themselves you services as a crutch.

[00:41:08] JM: Yeah, it’s a weird business. Interesting. Seems really good. I like talking to them. I like Kolton.

[00:41:14] SD: Yeah. It's cool to see that Chaos Engineering wasn't a thing until – You could look at the Netflix Simian Army stuff and all the cool stuff they did internally. Until Gremlin came along, that was viewed as like black magic that a whole bunch of like ops engineers did internally. It was never a thing that you could say, "We'll show you how to do it. We'll give you a product to manage it. We'll make this accessible to you." That I think has been there genius. They took a really hard thing and made it very accessible.

[00:41:41] JM: Well, he rebuilt at Amazon, right? Didn't he go from Netflix to Amazon, and then at Amazon he was like, "I'm going to build chaos thing again." Then he is like, "Maybe we'll just do this as a company."

[00:41:51] SD: I still remember that. I mean, it was really funny, a little bit of war stories. I think this is 2010, maybe it's 2011. It was one of the very first velocity conferences that O'Reilly puts on on web performance and web infrastructure. They used to hold it at the – Shoot! Is it the Hyatt? The one right next to the Santa Clara Convention Center that hosted a bunch of these O'Reilly Conferences. Anyway, Jesse Robbins was the cochair of the conference, and Jesse was at the time the CEO of Chef, which was a company that I invested in, was on the board of. I got really lucky investing in Chef, because that was my window into seeing what this devops transition looked like from 2010 onward to literally see the smartest minds in infrastructure coming out of places like Amazon, and Netflix, and Yahoo, and Google, and eBay and wherever else and start taking their craft to the rest of the world and like putting this movement together. It was amazing to be front and center on the night.

But that night, Jesse said, "Hey, you're the venture guy. Buy a bunch of beers and I'm going to host the speakers and a few close friends for like informal cocktails late-night up in this week that they give me." Tnd there was probably about 20 or 30 people in the room that night Max, and there was operators like John Allspaw who was at the time running infrastructure at Etsy, and Steve Souders who, web performance guy in his own right, and Steve has his own company now that he's working on. They were the other cochairs.

Jesse was there with a bunch the Chef people. Artur Bergman who runs Fastly was there who we had already invested in. That night I had invited out Olivier Pomel and Alexis Lê-Quôc to come out from New York, who are the founders of Datdog, who were just getting this thing

started and we're still trying to figure out if we're going to invest in this brand new thing called Datadog. I introduced him around the room to bunch of those guys.

Mark Imbriaco who's run infrastructure at LivingSocial and currently is doing it at Epic Games. A lot of really smart people were in the room that night, and funny enough decided to invest in Datadog coming out of the evening. But also in that room was a young engineer name Kolton Andrus. He wasn't that young. Kolton's a little bit of an older guy. But he was a staff engineer and we got connected and kind of had loosely stayed in touch since then. It was over that arc of like five or six years that Kolton actually refined this idea and developed what it could be and really kept working on it to the point where it became Gremlin, and then we reconnected with him and him and [inaudible 00:44:33] when they were going to start the company. He knew that he was going to put something like this together. Didn't know exactly what the product look like. Didn't know exactly what the direction was. But because we had known him literally all the way back until that one of those very first Velocity Conferences, it was like, "Yeah, whatever you guys are going to do, we're in. Let's get started."

We literally started helped him start Gremlin off of a hand wave about what the idea could be. There was no deck. There's no business plan. There was no product vision. There probably wasn't any official code written as Gremlin Inc. but it all came out of that relationship that we traced back to that one night at the party.

[00:45:10] JM: Netflix was doing Chaos Engineering in 2011. Weren't they? Why did it take six years for that technology to be productized? I mean, so many technologies –

[00:45:20] SD: That's a good question. You could say Amazon was, right? Because when Jesse Robbins –Even before Jesse started Chef as a cofounder, his title was Master of Disaster at Amazon, like his actual business card. He was entirely focused on site reliability, Amazon-wide, and was the person who was like, "We're going to test and we're going to drill and we're going to tests and we're going to drill so that failures don't impact up time, or they minimize it."

From there, Netflix and the Netflix on to the rest of the world, why does it take a bunch of time? It takes a bunch of time, because these problems aren't the most critical problems that people

face. At that point, Amazon and Netflix were at the scale where they were actually able to do the math to say very simply, “What is a minute of downtime cost us?”

Second thing that started to happen is those companies were also at the scale where they could go, “We have so much infrastructure. So many layers of abstraction, so many dependencies, that we can't spend all of our time engineering out the likelihood that something will fail.” We have to think systemically. We’re sitting on top of a complex system and now we have to assume that the system will fail, and it's not up to us to just sit there and beat ourselves up, because we didn't prevent the failure from happening. We now have the ability to work with infrastructure systemically at a scale where we can go, “Failure is a statistical event. It is not a thing to be avoided. It's the thing to recognize will happen and to engineer around.” That was a really big shift. But you had to be focused on running at a certain scale and running at a level of abstraction for that to really be obvious where you could say, “It is out of my hands to really know that every one of these services will deliver me the result that they need in the timeframe that my application is expecting and that these hard drives that I never touched sitting somewhere in an Amazon S3 bucket will never go down, and that there won't be a fiber cut between Ashburn and Palo Alto at the exact same time that I want won't run out of disk space on this instance of this node.”

People recognize that the complexity to the applications and services just outstripped their ability to pin down all the underlying variables. What I think was it was just a timing thing. Amazon was there. Then Netflix got to be there. As an organization who said we’re not in the infrastructure business. We’re literally going to depend on Amazon to make sure that Netflix streaming runs. That's a scary place to be, where you don't even own the infrastructure and you've turned it all over to somebody else. Think about how groundbreaking that was at the time.

Necessity, mother of invention, those people needed to get out ahead of that problem. Now I think what you see happening why Gremlin is working and taking off the way that it is, is the world starts to look a heck of a lot more of like Netflix and Amazon in 2011. There’s just a lot more people who go, “Great! I'm all in on the cloud.” “Oh, wait a second. I can't physically touch that server. I can't configure a network to make sure that these things happen exactly the way

that I want." Those are big dependencies. I think people are just going to continue to recognize that they can own the machine and left the engineer around it.

[00:48:33] JM: Okay. Gremlin is a very differentiated company today. There's a number of investments in your portfolio that are I think, to a naïve observer, less differentiated. You said Chef today, when you hear Chef, you won't hear it in a series of commas, Chef, salt stack, puppet, whatever your other infrastructure scripting tool is.

Datadog's kind of like that. Datadog is a fantastic infrastructure monitoring tool, but to a naïve observer they'd be like, "Okay, there's 11 million infrastructure monitoring tools. Why do I care about Datadog?" What's good about those sectors is they're not necessarily a winner-take-all. There's not super strong network effects in them. So you if you want to start a new infrastructure logging company, it's not like you're zoned out by Facebook's network effect. You can dart a new logging company. You can find customers.

So in some sense, as an investor, you know, you have a downside a little bit capped, because you know they're going to get some customers. But how do you identify in the non-winner-take-all markets, how do you identify the ones that are going to be successful enough to drive those returns that you need from an investment?

[00:49:45] SD: That's a great question. Basically, how's does a crystal ball work? Everybody gets it wrong. Everybody gets a little bit lucky depending on the company, depending on the deal. I think that's the first starting point. Earlier, you go in particular, is you're never going to know exactly how it pans out. But there are some dynamics that we do look for.

So there are things that kind of improve your margin of safety, give you a wider margin of error. Those things look like, "How well does this team know this problem?" If you told me – This comes and speaks a lot to why Amplify really focuses on technical founders. A lot of times we interspersed the word technical founder with practitioners and domain experts, but the reasons we like technical founders, practitioners, domain experts, is because they know problems.

You think about Datadog for a great example. Olivier and Alexi ran infrastructure and engineering at Wireless Generation. Not a lot of people know Wireless Generation, very

successful company. It was built. It was sold I think to News Corp. I'm not positive. But they had a really – They built a really successful business and they ran a 60% or 70% engineering and ops team. They were responsible for keeping a site up and getting code shipped and everything else. They were the ones who lived there and sat, like I said, day-to-day, "What we can understand about our infrastructure in Amazon is pitiful. To solve our own problems, here's what we ought to do."

They came up with the vision of what the solution looked like from an intimate understanding of the problem. I look at that all the time. Is it a surprise that we backed Kolton to go after the Chaos Engineering problem as someone whose first job was to focus on site reliability and then turned in the guy who developed the tools and then realized how it would extensible somewhere else.

Is it a shocker that we backed Artur Bergman when Artus was the CTO of Wikia and saw every single ways that traditional CDN and delivery was failing him. He said, "No, there's a different way. There's a better way. This is not built for the modern internet. If I think about what I need for performance and security and reliability and taking advantage of what an edge architecture can do, I need to do something different.

A, number one, how we look at it is who are the people and how tightly aligned are they with the problem? I'll use the canonical example of the people I like to pick on, which is you come out of Harvard Business School or Stanford GSB and you kind of create a matrix and you go, "The market that's most interesting for me to go after is this one, and I've arrived at that conclusion by lots of analysis and by talking to a bunch of people. Wonderful. That's great."

Even if you're right, are you going to have any credibility to go in and talk to that first customer who's going to have to look you in the eye and trust you to build their software? No. Technical founders who go, "I've been in your shoes. I live this problem. Let me tell you how I solved it and let me tell you why I started Datadog, or Gremlin, or Fastly, or Chef." They all share that origin story. That's a huge thing we look at and we think that that dramatically increases the likelihood of success or creating something that actually stands above the crowd.

We also look at big markets. There are interesting companies doing interesting stuff that just aren't around big trends. I think a lot of times people get bent out of shape when we don't look at those and go, “That's a really big differentiated thing running in a big differentiated trend,” but that happens. It’s true. We think about that a lot.

Datadog’s growth not surprisingly mirrors the growth of Amazon. The first thing people do once they get on Amazon and they get going is they go, “Man, I need a monitoring solution to understand what it is.” It’s like literally the first-order problem they run into after they get going. Datadog is there and their growth has been stupendous as a result.

Certain markets are just better and bigger than other markets where you have to have a little vision to see where they might go. We all have our misses where we didn't have enough vision to see what could become. Yesterday, just in our partners meeting, or two days ago, we were talking about how many people really thought Twilio would be as big as Twilio is. We sat around going, “Man! Voice API and text API, how many applications really need that?”

Now, mobile took off and everybody – Notifications and communications are huge as a developer service. Why Stripe for payments? All of these things, like they are all driven by the really big markets. That's the stuff that matters.

[00:54:09] JM: Only like half the population is online.

[00:54:11] SD: And only half the population is online. It is crazy to think about the fact that we’re in the U.S. You and I are both sitting here in San Francisco, California. So we’re first like San Francisco myopic and then we’re probably North American myopic. The reality is, as we’re talking about at most like 450 million people that we actually like really think about day-to-day, the number of internet users globally is – It will soon be in order of magnitude larger than that. When you get a solution that works in some of these markets, not all of these things need to be localized and translated and have their Chinese language equivalent, or Hindi language equivalent, or whatever language and customs equivalent solution. In the way that you could argue Uber, or maybe Netflix, maybe not Netflix. Netflix is global, but there’re a lot of things that do need to be localized. But a lot of what we do in infrastructure land doesn't. The solution that

works here is going to be the right solution no matter where you are. Those make those markets like even bigger.

[00:55:16] JM: Yeah. All right. We're running out of time, but I want to ask you one more question. So when people listen to these kinds of podcasts, they probably – You get excited about software. You get excited about, "Oh, investing. That sounds fun. It sounds interesting." I love talking about this stuff. I think it's really interesting and it's really fun. I just get wrapped up in it. But what I've learned, I guess reading enough about it and just like being here, being in San Francisco a bit, is that there's a Machiavellian side to the investing world.

[00:55:48] SD: I'm not sure I like where this expression is going.

[00:55:50] JM: No. I'm not calling you Machiavellian, but you've been a venture capital for a while. What kinds of adaptations to your views on human nature and the ways that you interact with people have you had to make since getting into this highly Machiavellian world?

[00:56:04] SD: Yeah, that's a really good question.

[00:56:07] JM: That you feel comfortable saying on air.

[00:56:08] SD: Yeah. I think your question is how is my experience shifted my views on what people do and how to interact with people and kind of what it takes to do what I do or what it takes to build companies. Ego is a real thing. It matters. People put a lot of themselves in starting companies. There's a lot of history and emotion and personal rationale that gets tied up in someone's decision to start a business. That's a really important driver of why people are compelled to make things work. That's when ego can be used for the purposes of good.

It also shows up as a force for bad when that gets in the way of what people are trying to accomplish, when an individual's ego supersedes the mission of a business or doing right by customers, doing right by employees. I've seen that a lot of times and it's something that I'm much more attuned to lookout for.

People's capacity for growth I think has been a thing that I vastly underestimated, and I think VCs get a bad rep some time, because they go, “OH, venture capitalist guys and investing companies, things don't go well. They got one play and that's fire the CEO and get a new CEO.” Maybe at Kindle, like your sports team isn’t doing well, so they fire the manager and they get a new manager and a new coach, whatever.

There're some truth to that. There’re some reasons that why that is a very common solution. A lot of times, problems need to be solved from the top. But what hand-in-hand with that is when somebody as a founder, and this could be because of their background is not being an exact. Their background as being an engineer. Their background is maybe being a woman. Whatever it is, those preconceptions oftentimes got wrapped up in the like, “Well, we just need to get a new CEO.” It’s two years in. Companies gotten this big. They can't really take it any further.

I would definitely tell you that early in my career, that whitewashing with that logic was something I fell victim to. I've been blown away at the capacity for growth that I've seen amongst the founders that I get to work with. It’s not in everyone. Not everyone will scale to the challenge of what they have to do as a CEO, or as a leader. But I did not know that when Olivier Pomel started Datadog, that he would build an 800+ person company spanning multiple continents and hundreds of millions in revenue and build literally one of the best cloud software products in the world today.

I didn't know he had the capacity to do that, and I'm not sure he did either. I think there's plenty of things that he learned along the way, and I’d say the same thing about Artur at Fastly, and I think we’re seeing it with guys like Kolton at Gremlin and others. It's amazing to watch people who come from one discipline really take on another. I’d say that's another thing.

One thing that really hasn't changed, and maybe this is just part of my view on people, is I think fundamentally people are good and they want to do right by other people. Sometimes you meet people who are not, and they are exceptions and not the rule. But a lot of times I wring my hands over whether someone will do something that’s solely in their self-interest or do something to someone else's detriment. I continue to be surprised by the number of people who actually are always, always thinking about how to put others ahead of them and subordinate their interests to that of their team, or their cofounders, or in cases, their investors and their

shareholders. It happens every single day, and that's another thing that I think I started with and I kind of felt and I hoped and believed, but it's amazing to see it get playing out time and time and time again. I've been doing this for 20 years. So I've seen a lot of these data points.

[00:59:39] JM: Okay. Let's end on a high note. Sunil, thanks for coming on the show. It's been great talking.

[00:59:43] SD: Yay! I really appreciate you having me. Thanks a lot, Jeff.

[END OF INTERVIEW]

[00:59:48] JM: We are running an experiment to find out if Software Engineering Dailydaily listeners are above average engineers. At triplebyte.com/sedaily, you can take a quiz to help us gather data. I took the quiz and it covered a wide range of topics; general programming ability, a little security, a little system design. It was a nice short test to measure how my practical engineering skills have changed since I started this podcast. I will admit that though I've gotten better at talking about software engineering, I have definitely gotten worse at actually writing code and doing software engineering myself.

But if you want to check out that quiz yourself, you can help us gather data and take that quiz at triplebyte.com/sedaily. We have been running this experiment for a few weeks and I'm happy to report that Software Engineering Daily listeners are absolutely crushing it so far. Triplebyte has told me that everyone who has taken the test on average is three times more likely to be in their top bracket of quiz course.

If you're looking for a job, Triplebyte is a great place to start your search. It fast tracks you at hundreds of top tech companies. Triplebyte takes engineers seriously and does not waste their time, which is what I try to do with Software Engineering Daily myself, and I recommend checking out triplebyte.com/sedaily. That's T-R-I-P-L-E-B-Y-T-E.com/sedaily. Triplebyte, byte as in 8 bits.

Thanks to Triplebyte for being a sponsor of Software Engineering Daily. We appreciate it.

[END]

