EPISODE 05

[INTERVIEW]

[0:00:00.3] JM: Mike Vernal, you are an investor at Sequoia, you're a former Facebook executive. Welcome to Software Engineering Daily.

[0:00:06.4] MV: Thank you for having me.

[0:00:07.9] JM: You were an executive overseeing the transition from desktop to mobile. This was a strategic inflection point for Facebook. What did you learn from that experience?

[0:00:19.7] MV: Interesting question. It's probably worth setting the context and I feel even today, I'm still learning some of the lessons. I think this is probably most acute in 2011. In 2011, we used to do a developer conference every year called F8. I think we had one in 2009, we had one in 2010, we had one in 2011. I think it was slated for roughly September 2011. I think each time we had the conference, the – our internal expectations for what we should launch and the amount of internal alignment we tried to create around the event increased.

In the early days, maybe a couple of teams would launch some things and we think about what we're going to launch three months beforehand. I feel for the 2011 launch, we really ended up with a scenario where we're trying to launch all the things. If I remember back, we were shipping a new version of profile which we call timeline. We were shipping a new version of newsfeed. We were shipping this giant extension of our platform called Open Graph and a handful of other things at the time.

Because there were so many different things that we were trying to launch and they were so interconnected, my sense is most if I remember correctly, most of the company's effort, or a good chunk of the company's effort from early 2011 to really September 2011 was entirely focused on those products and the broader launches at F8. All of those products were entirely desktop-centric. They probably had mobile equivalents and it's probably unfair to say that there were an afterthought, but they were certainly not the primary focus.

We had F8. I remember it being maybe third or fourth week of September, so maybe September 20th or 21st or something. We launched and then there were a few days of people just getting some rest and recovering and getting back into the swing of things. It was really right around then that I think we stuck up our heads and realized that in our nine months of focus for F8, we had been left completely flat-footed as to mobile.

I think during that time, we had treated mobile I think almost more as a tax than an opportunity. It was an incremental platform we had to support. We had a bunch of great people that were working on it and they were doing good work, but I don't think – I'm not sure anyone inside the company would have characterized it as an opportunity. I think we would have characterized it as attacks.

Actually, I think that framing really also informed our technical strategy. At the time we lacked just really great iOS and Android engineers. I think finally, probably supply and demand is coming into balance in 2019, but for probably four or five years, there just weren't enough really great iOS and Android engineers. We certainly felt that in 2011.

Because we have this mentality I think of mobile being a tax, we tried to find ways to minimize that tax. The primary thing we looked at was HTML5 and we were like, can we somehow build these apps in HTML5 and have a thin wrapper around – a thin wrapper, native wrapper around iOS or Android to make that happen?

It was a good theory and I think you could probably – with things like React Native and Electron and the like today, you can actually get closer to that being possible in 2019, but in 2011 it was just way too early. We built a bunch of versions of the Facebook product in I believe it was called Faceweb internally. They just weren't very good. I mean, the engineers were great and we did our best work, but you just couldn't get to 60 frames a second and you just couldn't get to the smooth, buttery feel of native iOS code.

I think in the end of 2011 is when we have this big, mental shift inside the company, where we realized being distracted for nine months, we shipped a bunch of stuff that was entirely desktop-centric, we've been treating mobile as a tax instead of an opportunity, we need to completely

upend our philosophy and start thinking about mobile as the future, as opposed to a tax. Figure out what that means for the product, for the platform, for the business, etc.

[0:04:51.7] JM: Now you're at Sequoia. When you reflect on the fact that it took Facebook a little bit to comprehend the significance of mobile, then again it took a lot of people time to comprehend the significance. It took time for the consumer trends to develop. Do you have any reflections about the adoption cycles of technology and the fact that one of the most savvy tech companies at the time and still today, but took a while to see how big this trend was going to be? Do you have any reflections on the adoption trends of technology?

[0:05:34.1] MV: Well, I think this is fundamentally the innovator's dilemma. I think when you are big and you have – when you're at scale and you see something that looks small, it's very hard to value it based on its future value, as opposed to what it is today. I remember going back before Facebook, I worked at Microsoft for around six years. I remember at the time, I was in some meeting with Steve Ballmer, who is the CEO at the time. He made some comment to the effect that if a business was not going to be a billion-dollar business for Microsoft, there was no reason for them to be playing in that space.

I get the spirit of the comment and I get the spirit of it today. If you're building a product, if you're building a company, you want to go after a large enough market. I think if you're inside Microsoft and you hear someone like Steve Ballmer say that, it's really hard to work on something that is maybe going to do 5 million in revenue the first year and 10 million in revenue the second year. By the way, those numbers would be fantastic if they were a startup, but within a company like Microsoft, that's peanuts.

I think the desktop-mobile dynamic inside of Facebook was fundamentally the same thing. The desktop numbers were so huge and the mobile numbers, well actually, they were bigger than you might think, but it was easy to anchor on desktop really being the focal point of the business. I think what you have to do – I think two things; one is you have to have a culture where you let people explore future things, even if they seem – even if they don't seem that valuable, or even if they seem tiny. You have to find a way to shield them from the inertia and just politics. Politics sounds negative, but just the bureaucracy of a larger org.

When you realize that you've been caught flat-footed, you have to course-correct with all of your might. I think, the first mistake people make in the innovator's dilemma is just not paying attention to things that are small. Then I think the second thing is they try to eat their cake and have it too where they try to keep their current thing and then dip their toes into the new thing. I think you really have to burn the boats and move over to the other thing with all of your might and make that bet.

It's entered the realm of mythology as opposed to fact at this point. I think one of the things that Mark would say at the time was after he made this mental shift around desktop and mobile, if a team came in with desktop mocks for their product, he would end the meeting and say, "Come back when you have mobile mocks." That changed behavior pretty quickly .I do think I came in a few times with desktop mocks and he didn't end the meeting, but the spirit of it was true and it was clear. Sometime around in October 2011 that everyone should just be focused on mobile.

[0:08:49.5] JM: Were there any other moments in your time at Facebook where it felt there was an existential threat to the health of the company?

[0:08:58.7] MV: I think in the early days, maybe up through the IPO and a year after, they were almost quarterly occurrences. I think there was a – I'm trying to think of good examples, but I think there was a constant sense in the early days that not that we were under attack, but that there were constantly ups and downs. I think one example of which has been written about was obviously the fact that Google was working on Google+. At the time before it launched, we knew Google is working on it, we knew Google was devoting a lot of resources to it and obviously, Google had a lot of distribution at the time. They had google.com, they had Chrome, etc., etc.

That was probably the most discreet competitive threat in the mid, the adolescence of the company. I think in the very early days in the first few years, there was obviously MySpace and then around the world, there were things Vkontakte and [inaudible 0:10:03.9] in Germany. I think sometime around 2008-2009, Facebook started to emerge as the clear leader. I think when Google+ came along as one example, or rather before it came along, there was a fear that this was something that might really upset the balance and I think was a rallying point for the company.

[0:10:24.8] JM: We had a show a while ago with a Google engineer who had worked on Google+. I asked him for some perspective on his time working on that. One thing that surprised me was in contrast to the narratives you often hear around Google+, like this thing is a complete mistake, total dead end for the company, total distraction, total hyper-competitive focus. His memory and his position was that this was absolutely a worthwhile endeavor.
Google was worried about these areas of the internet that would be difficult to index. They felt this was a walled garden environment. It was potentially a threat to the internet ideology that Google had dominance over. When you put yourself in Google's shoes at the time thinking strategically, do you think it was a smart decision to put all the wood behind that arrow? I mean, that was Google's own innovator's dilemma moment, right?

[0:11:25.4]j **MV:** It's a very good question. I haven't thought about it before. I would say, I could argue both sides pretty easily. I think the Amazon, Jeff Bezos' view would be – you should be customer-obsessed, not competitor-focused. This is not to diminish any of the work that the Google+ team did, but I think it would be fair to say that the effort was competitively focused as opposed to consumer-obsessed. That I think would be the obvious strike against it.

On the flipside, I think Google had been this incredibly fast-growing company that was really the darling of the tech ecosystem. I think Facebook was maybe six years behind it, but growing at a similar clip and with a similar I think amount of potential. It was a reasonable I think strategic decision to make. It was a reasonable thing to try. I think executed differently, it might have been more aggressive, it might have had more impact.

I'm sure you read Ben Thompson every so often, but I think he has some comment, I can't remember exactly what it is, but it's around culture and how it's really hard to be – it's really hard to be excellent at two different types of product with the same culture. I think it would have been hard for Google to be excellent at social while also being excellent at search.

[0:12:58.0] JM: I don't want to take this discussion too far, but if you were in charge of that Google+ project, or Google's social efforts at the time and you took that more tempered approach you're like, "Okay, the competitive strategy here is to flank, right?" It's some indirect –I mean, you could even say Google's done that today. Google feels like a social experience now,

but it's much more subtle. You know that Google is taking into account some social graph when it returns your search results. Do you have any sense for how you would have architected that strategy if you were in Google's shoes at the time when you see this threat of Facebook?

Do you do some top-down effort, or do you just massage into your various key notes and conversations you're like, "Yeah, there's this Facebook thing. We should probably think more socially or something like that."

[0:13:50.2] MV: Well, hindsight is 2020 so it's a little bit unfair.

[0:13:52.8] JM: It's factual.

[0:13:54.1] MV: Yeah, fair enough. I mean, the most obvious thing is Google owns the most popular operating system in the world by a factor of four or five. Again, not to diminish Google at all, but it would be hard for me to today articulate what their messaging strategy is. I think there's hangouts, I think there's out low, I think there's duo, I think there's an assistant you can talk to in a separate app, etc., etc. There's a lot of different messaging apps. I mean, Apple has a very coherent messaging strategy around iMessage. It's iOS only, I think to make the iOS ecosystem work better together. I get their strategy. It's not a cross-platform strategy, but I think iMS is just the dominant messaging platform in the US for iOS.

With the benefit of hindsight, I would have just gotten very hard after messaging and gone very hard after mobile. In some senses, I think Google made the same mistake that Facebook did around this time, which is they fought today's war, instead of tomorrow's war. That's the most obvious thing. Even if Google+ in its first incarnation hadn't worked, had Google just built a great, a singular messaging app on Android and had it really been the dominant messaging platform on Android and then also had that be cross-platform, that could have been a very, very powerful social strategy. Obviously, WhatsApp which Facebook ended up acquiring, I don't know how many users they have, a billion plus users. Look at WeChat in China, messaging is clearly was the future of social in many ways. I think it should have – there is a strong argument that Google should have been the best position to do messaging given the dominance of Android.

[0:15:44.4] JM: You were the co-creator, or you worked heavily on the product for Facebook login. What were the most difficult engineering problems around building Facebook login?

[0:15:55.4] MV: I was thinking about this question, because I was thinking about what you'd ask me. I'm the drive here today. This one I think is a little bit of a funny story. My very first job out of college was at Microsoft. At Microsoft, I was working on the XML Web Services Group. I actually sat on the SOAP standards body and I thought about this thing that hopefully no one on the podcast remember is called the WS star protocols. It's like CORBA, but worse.
The interesting thing about that group, the XML Web Services group in Microsoft is it was the – a little bit the descendant of an earlier project at Microsoft called Hailstorm, which again people probably won't remember, but Hailstorm was this very broad Microsoft passport identity layer for the internet does everything strategy in maybe 2000-2001.

It was before I joined Microsoft, but it had been this sprawling project with a lot of people working on it and a lot of noise and energy. Like I said, the team that I joined which was post hailstorm was still a couple hundred people working on all these different web services. One of the lessons I mislearned when I first joined Microsoft was the way you got things done was you allocated 200 people to a project and told them to go and start doing some stuff.

When I showed up at Facebook, one Facebook was very small company, it was probably 70 or 80 engineers at the time. I had joined Facebook to work on platform, because I'd spent my entire career working on developer platforms and the first F8, which was in 2007 was really the thing that got me really excited about the company.

I joined to work on developer platforms and the team at the time was maybe eight or nine people. At the time, there was this interesting narrative and this all ties interestingly back to the desktop-mobile question you asked at the beginning. At the time there was this narrative that Facebook was trying to reappropriate the web and have the web live within facebook.com, and people analogized it as AOL and AOL keywords back in the day.

People would talk about oh, UPS is wondering whether they should kill ups.com and just have a Facebook app? New York Times is asking the same question. Then people from the outside

would look at this and say, “Look at Facebook's evil master strategy, which is to kill the web and reappropriate it inside of facebook.com.”

I could certainly understand if you were Google at the time and your entire business was predicated upon the “open web,” that more things moving within Facebook seems not just strategically bad, but in some senses morally bad, because openness is good. I have walked in and I was chatting with some folks in my first couple of weeks. The sentiment I got was this isn't some evil master plan to get the entire web to come within Facebook. When we designed the first version of Facebook platform, our mental model was Facebook as the operating system, it had some APIs, there are apps that are running within Facebook.
In the very first version of Facebook platform, Facebook had tried to redesign a lot of the functionality like events and groups and even messaging as apps that ran on top of the Facebook OS. Their perspective was our mental model is just Facebook is the OS and these are some apps, and so we didn't really think about how you would use this on other sites, but there's no philosophical opposition to that.

I was like, “Okay, cool.” If anyone object if I go figure out this Facebook login thing and people were like, “Sure, go for it shrug.” There were three of us, I think me, Scott and Waizu who had been on my team at Microsoft as well and had joined Facebook a few months before I had, and James Lozinski. We just started working on it. I don't want to quite say we hacked it together, because that's a little bit dismissive. A lot of the plumbing already existed. There existed this notion of API keys and session keys and APIs and the like.

A lot of what we had to figure out was just how do we – how do we securely get these session keys to JavaScript on third-party sites. At the time, it was super messy. I don't know if it's gotten any better, because it's been a while since I did this. At the time, the way you had to do this was with hidden iframes and you have to do message passing through these three layers of iframes. We had built this message passing protocol on top of – I think the way it worked, it's been a while, is you could change the URI fragment and both sides – it was very ingenious, or hacky depending on your disposition.

[0:20:59.9] JM: Very Facebookie.

[0:21:00.9] MV: Yeah. I think the way it works is both the parent – a parent knows what the URI of its iframe is, but nothing else. Then the iframe itself knows what its URI is, but nothing else. Then the iframe can also open another iframe inside it and it knows the URI, but nothing else. If the parent opens up an iframe to your website and then that website opens up an iframe back to the parent, you actually get this communication protocol.

The only thing you can change is the fragment, because if you change any other part of the URI, it causes a reload. You'd end up with this – the parent would pass a message in the fragment down. This would pass it down. It was incredibly hacky, but it worked. That was actually the JavaScript message passing stuff was the vast majority of the complexity. Then once we had that working, we – the second hardest thing was actually just designing the UI, so that it was pretty clear to users what was going on.

We basically had it working within maybe four or so months. To me, it was always – it was just a fascinating story, because again, Microsoft had 200 people working on this thing for two years and I'd never really exited the “spec phase.” We got the entire system working in a few months and launched it. I think I announced it in May and launched it in July and I joined in January. It was pretty fast turnaround.

Again, because we had been entirely focused on just getting it to work as opposed to anything else, people were concerned that we weren't using OAuth or OAuth 2, I can't even which one at the time. There's this other engineer named Luke Shepard, who came along, who cared a lot about OAuth. He retrofitted the system to be OAuth compliant and we went from there.

[0:22:56.0] JM: With the retrospect of time, what is it about the cultural differences between Facebook and Microsoft, or perhaps Facebook and the broader cultures of companies that allowed you to ship that thing in four months and it was stagnating in Microsoft?

[0:23:19.6] MV: I mean, I think there are a lot of things. First and this is a lesson I've painfully learned, because I've made this mistake a number of times, especially later in my career at Facebook. Strategy is important, but you can't be – you have to find a way to balance strategy with both simplicity and customer focus. I think a lot of the mistakes Microsoft made at the time were strategizing around what was in Microsoft self-interest and what are the things that could

be built, but without really thinking up a technology and thinking up its use cases, as opposed to going to people today and understanding their pain points and making them better.

Then two, just getting – innovator's dilemma applies to more than just – the philosophy behind it I think applies to more than just products and markets. It also applies to teams. I think it's easy at a place at a large company. I think as any company gets large, it probably falls victim to this where it's if you could why throw three people at a problem when you can throw 50 or a 100. It's like the line from contact, which is why buy one when you can buy two you for twice the cost.

It takes I think a really high-level of maturity in the leadership of a project to say, "I don't want any more people. Stop trying to send me people. I'm just going to use – I'm just going to take these 10 people and go build something." Incentives and companies are usually around on some level, competing for resources, because resources are usually scarce, and so there's usually some headcount planning process. It would be weird in a headcount planning process for you to say like, "You know what? I have 10 people right now, I only really need eight. You can take two more. I just want to focus with these eight," because you feel you're signaling to everyone else that your project isn't important or something.

You just end up with these dynamics as companies grow, where you think if you have an important problem, you have to throw a lot of people at it. That obviously doesn't work, you end up with just complexity, you end up with 200 people not quite knowing what to do, but trying to do busy work, etc. I think having three people that just go make it work. By most measures, the thing that we launched, I think you could criticize on every single dimension. There were probably technically a bunch of problems with it. Maybe the UI was not as great as it could have been. It certainly wasn't standards compliant.

[0:26:03.7] JM: You were a vice president at Facebook, vice president of I think product and engineering around this time. You were in charge of, I think the department was called utility. This was obviously an esteemed role in the organization, but newsfeed was not in your domain. Newsfeed was driving a lot of the product, the direction of the product, at least as I understand from my other conversations with people. I think a quote was, so the feed goes, Facebook goes. Were there any frictions in getting products out the door, because feed was not directly in your dominion?

[0:26:48.4] MV: It's a good question. I would say there was healthy friction in terms of people wanting to do what was right for their individual products. I think the broader question is very interesting. I don't know if you read over the weekend this very, very long blog post by [inaudible 0:27:05.4] called [inaudible 0:27:06.8]. He skimmed it.

[0:27:08.6] JM: More than skimmed. Concerted, scoured like read for five page –

[0:27:12.9] MV: It was a commitment.

[0:27:13.3] JM: All right, skimmed down, read for five minutes. Wow.

[0:27:16.7] MV: Yeah, it was a commitment. I think someone I work with referred to it as a Russian novel. It was interesting, because he talked about these dimensions of I think engagement and status and utility and something like that. It's interesting to see someone else use that term to think about it. I think the high level – the going in thesis was that we could make Facebook both more engaging and more useful.

Actually, there were two orgs that primarily worked on the core blue app. One was called engagement and one was called utility. I think the going-in thesis, it was an interesting hypothesis. I think what we learned is engagement. It's hard for one have to do both and I think engagement just ends up trumping utility, especially since the core economic engine of Facebook was so deeply tied to newsfeed, which was the driver of engagement within the app.

I think we ended up in this dynamic where engagement was always as the focus and the center, and the utilitarian aspects of Facebook ended up being more on the periphery. To that point, I don't think there was friction per se. I think, the dynamic that happened most was people would build a new product and they would – they would want to grow the product, because that's what people want to do. They would see newsfeed as this just incredible distribution channel.

They would construct feed stories that would show up in feed. An example might be an app like events, or functionality like events. You might want to have a story in huge feet when someone joins an event, when someone says they're going to an event, blah, blah, blah. It would be very

natural for the events team to say, “Hey, here's my events product. I want to grow it. It's valuable. It's getting people together in the real world. People like it. I want to grow it.”

The best growth channel is actually in newsfeed, and so we should figure out a way to generate more stories in feed. Then the feed team would say, “That's fine, but we want to one add balanced and efficient and auction as possible for the content that shows up in newsfeed.” We want to show content that people like and that want to engage with, and so we're not going to artificially boost your content. We'll put your of content into the auction like we put other content, but we're not going to help it out in the early days to help you grow to achieve some critical mass.

That was probably the core tension around people wanting maybe artificial boosts in the early days to help kick-start some product. The feed team to its credit, I think holding a pretty hard line and saying like, “No, we want to have a neutral balanced field. We want to show people the content that people like and we're not going to artificially change distribution.”

[0:30:30.0] JM: What was the biggest mistake you made as a manager at Facebook?

[0:30:34.1] MV: What I would say is this; there were a handful of times where I would end up with a hodgepodge of different teams that were working on things. I think at the time, my approach was how do I fit these things together in a coherent way? It was a little bit like, I've got a bunch of ingredients on the table, let's figure out what soup I can make from them. As opposed to forget the ingredients on the table, what is the – who is the customer? What is their problem and what problem are we trying to solve? There's one these 10 ingredients that matter and then the other nine ingredients don't make any sense.

I spend a lot of time trying to figure out okay, there's these 10 puzzle pieces. What is the right way for them to go together? I should have just said, actually let's – one, let's ditch these seven or eight of these things are good in the abstract and I'm glad that they exist. They don't have coherence with everything else. Let's find another home for them, or just figure out something else because you end up playing the game with the pieces you have, instead of intentionally designing, like thinking from first principles what games you should be playing.

[0:31:51.9] JM: That basically boils down to an issue of focus, which is something that I hear a lot of people struggle with, both at small companies and large companies. To what you said earlier, you want to be able to have the departments of tinkering, right? You want to be able to have departments tinkering, I don't know, with various degrees of dedication on cryptocurrency, or augmented reality, these far-flung things. Are you talking about those kinds of departments, or what kinds of departments are you talking about that you should throw out?

[0:32:28.1] MV: No. I'm not saying that the – I'm not saying –

[0:32:32.4] JM: What's the characteristic of the puzzle pieces that were burdening you?

[0:32:36.2] MV: It's a very good question. How would I describe it? I would say, so the initial vision went unfulfilled, but I still think is interesting. There is an alternate universe where we had – where we would have pulled it off and we would have seen and maybe Facebook would look and feel different. I think from a technical perspective, there was always a – there was always an obsession with the graph at Facebook, and a sense that the world could really interestingly be modelled as a graph. Or really as two concepts; a graph and a feed.

I think the way these two things relate is the graph is a set of nodes and then relationships between those nodes. Then the feed is really like a database replication log, is a reverse chronological view of all of the changes that are happening in the graph. That I actually think is a pretty interesting and beautiful concept. I would say in the early days, there was this vision that were a few things. One is could we model our internal applications in this graph structure in a better way? Two, could we expose those applications to third parties, so that they could integrate them with their apps, which is why the Facebook API is called the graph API?

Then third was this notion of open graph, which is the thing that we launched in 2011, which was can we also help third parties model their data in this graph-oriented way and then help them integrate their data set with the Facebook data set? With the obvious benefit that as there were these state changes and maybe third party apps, that those state changes would also be eligible for showing up in newsfeed and potentially search and other things.

You can imagine one of our launch parties way back in the day was Spotify. I think when Spotify launched in the US was with Facebook login and was back in 2011. Our mental model was look, there's artists and there are albums and there are songs and they're listening to – those are the nodes and then there are edges like liking a song, or listening to a song, etc. Let's figure out how we can model all these things together and then integrate them into Facebook.

We had visions of being able to do things like aggregate, just understand which songs are your friends listening to the most this week, this month, etc. A sense that you could do this at the abstract platform level; you could understand the most number of edges from your friends to some particular node and some time window. Really if that is your abstraction, what is the difference between the most listened-to song on Spotify, or the most-watched thing on Netflix, or the most committed to repo on github? They're all the exact same thing.

That was the starting premise of a lot of this. Then once you have these graphs, again the very first version of the majorly rebuilt search engine that we launched was called graph search. The idea again was graph search actually let you traverse this graph in its graph structure. You could use natural language, which didn't work as well as it could have, but I think could have gotten better in terms of show me all of my friends who listened to some song. You could construct these graph queries in natural language and they would actually map down to actual graph queries. That was the starting premise.

I think there were a couple of mistakes; one is back to the Microsoft mistake, I think it was just too large a coordination problem. It would have been better to get the system to work really well for one use case and then move to adjacent use cases and we just tried to do too much at a platform level and then educate people too quickly and it just didn't work. Then two, we ended up with some strategic drift, where if you take the search team for instance; about a year into graph search, the search team decided that actually graph search was the wrong metaphor and that we should just do keyword search like Google and you should just be able to type in a couple of words and find a post.

In retrospect, I think that was a mistake because it was a – it's complicated to answer it. It's like your Google+ question earlier. Because on the one hand, I do think that's what customers would say that they want. It was really an effort to make search work more like Twitter search, which is

you can type in a couple of things and you can find the tweet you're looking for, like how can we make Facebook work well for that use case?

I think the differentiated thing would have been make graph search easy enough to use, that you could actually write these queries over the graph and ask interesting questions of the system. I think it's a little unfortunate in retrospect that we didn't fully explore that. That is overall a way of saying as we started to have this drift, as graph search ended up not graph search, but search. As other things became less focused on modeling out this graph and modeling out the open graph of how all these things connected together, they became less this coherent story of modeling the graph and then searching over it, and it more became a set of disparate parts.

It's hard to have perspective on something when you're in the middle of it, but I think at some point I should have realized these things weren't deeply connected together anymore. They had drifted apart. Then it became well, if it's not all connected by the graph, what are they connected by? I think the core mistake I should have realized is the premise of this has changed and we should we calibrate and reorient based on that.

[0:39:00.1] JM: Wow. One of the motivations for doing this interview series was the fact that the engineering practices of Google are well-documented. People talked about them a lot. In contrast Facebook's engineering practices, what makes Facebook engineering unique in a very good way is less well-documented. There are a lot of lessons to learn, especially given that when Facebook was coming up, the obvious model organism to look at would be Google. Like, let's copy everything that Google does engineering-wise, hiring-wise, onboarding-wise and Facebook did not do that.

From what I've heard from some people, some of that is because Facebook is this super rich, highly interactive PHP application, it's the first social application. Other people have said, it comes down more to an ideological thing, maybe not necessarily driven by the PHP ramifications. How do Facebook and Google engineering organizations compare to one another?

[0:40:11.2] MV: Well, I never worked at Google so it's hard to me to comment in depth. I will say a thing about this two, an interesting anecdote about Facebook culture is that I think many

of the very best engineers in Facebook's history will alter the cause of catastrophic engineering failures. I won't call people out by name, but a bunch of the worst bugs in Facebook history that would take the site down for two or three hours, or cause pandemonium were caused by some of the prior guests on the show, and just really, really great engineers.

I think the reason for that was this – not fearlessness, but I think it's really – again, going back to the Microsoft experience, I remember there was this element of lore within Microsoft. I don't know if it's still true, but I was told that so much of Excel was written in hand-tuned assembly. There was something like 91 global variables in Excel that were protected by one spin lock. That no one could unravel the hairball. Excel is basically, you can change the navigation and you can add a bunch of bells and whistles, but – at some effort, there was an effort to change the number of rows from – no, the number of columns from 128 to 256. I think they just made a signed and unsigned. That was I think you're still limited to 256 columns in Excel, because of the morass at the very center of it.

I think it's easy for systems, especially systems that are scaling quickly and evolving quickly to just become inscrutable and have people not be willing to roll up their sleeves, get their hands dirty and actually just go in and try to fix it. I think one of the core – The whole move, fasten big things gets maligned all the time now, because it's subtle and people don't like subtlety. I think a lot of the basis of fundamentally came from the engineering team. It was not a product thing, or a company-wide thing. To me, it was fundamentally a technical mantra.

The intent behind it was look, it is better to dive in and try to fix something and make it better, and maybe you make some mistakes along the way than it is to be just incredibly cautious and never get anything done, right? One of the things I thought was impressive, especially at the early days of Facebook, in the early days of Facebook was just the constant evolution of the underlying primary product infrastructure.

I think some of the very best engineers in the company and the group that GraphQL and React and a lot of those things came out of was this group called product infrastructure. They were constantly rethinking the core frameworks inside of the company and then just touching all the code to update the code to abide by these new frameworks. That's in most companies, I think there is either just fear. People are too scared to go do that, so you end up with 91 global

variables protected by a spin lock. Or there's so much – it's so glacial the pace, because you're so careful about the changes that people just get dissuaded from ever doing this, because why work on some new abstraction if it's going to take five years for it to roll out throughout the codebase.

There are a bunch of pros and cons to “move fast and break things.” I do think the tempo within the engineering org and the willingness to accept a few mistakes in the interest of making the codebase better, faster, stronger, faster was really definitional.

[0:44:08.1] JM: You're now an investor at Sequoia. How do the strategic decisions that you make in your day-to-day life differ from those you made at Facebook?

[0:44:20.1] MV: Many people ask me how investing is different than operating. The surprising thing is it's actually far more similar than it is dissimilar. I think if you're an exec in a company, especially an exec that is managing multiple different things, from a day-to-day perspective and from a year-to-year perspective, what are you doing? I think you're making sure you have the right leaders for a bunch of different areas. You're working with them to make sure you have the right strategy. You're working with them to help them hire and build out their team and you're working with them on resource allocation. You're working with them to help figure out how their team should scale and where those heads should come from, etc.

In many ways the work a board member and an investor is very similar. You're working with teams to help them think about their strategy, to think about their hiring plan, to help them actually hire and close people, to help them get finance and then help them get financed again in the future, etc., etc. My day-to-day relationship with many of the companies I work with feels very similar to the dynamic I had with people I worked with that Facebook. In fact, a number of the teams that I work with are ex Facebook teams that I knew at Facebook.

I think, probably the core difference is generally in an operating role, the things you work on emerge organically from your org. They're little seeds that you place and then they grow and then they hopefully become big things, or they're acquisitions that come into your org. Whereas in the investor world, I meet probably around a thousand companies a year. Of that, we’ll partner with one or two.

One, you unfortunately have to say no to a lot of people, which is never a fun thing because there's lots of amazing entrepreneurs that we meet and then lots of amazing ideas, but we only have capacity to work with one or two companies a year. Then once you say yes and hopefully both sides say yes, then it's actually surprisingly similar.

[0:46:28.0] JM: What are the potential synergies between the social media ecosystem and the cryptocurrency ecosystem?

[0:46:34.0] MV: Well, I think it's a good question. I haven't thought about it that deeply. The thing that seems to be happening across a number of messaging networks is just the introducing – the introduction of some coin to enable either peer-to-peer transactions, or probably transactions between consumers and businesses. That's probably not an acute pain point in the US, because the dollar is pretty stable and the US is pretty big. Transacting between peers is not that hard. The square cache app is great, it's really big. There's Venmo. Even Facebook Messenger has that integration.

I think it's probably much more acute in company in countries rather that have less stable currencies, Venezuela being an obvious example, countries where the payment is much more cash-based for a variety of reasons, so just digital payments are less mature. Or countries where you have a much higher rate of cross-border transactions.

There's this product in Africa called M-Pesa. I think for 10 years, people have been looking at that and saying, "Wow, that's a really fascinating case study. How do we offer something like M-Pesa for the rest of the world?" In some senses, I think the crypto messaging stuff is probably yet another attempt at cracking that nut.

[0:48:00.9] JM: Are there particular engineering problems that you saw people struggling with and either trying to solve, or successfully solving within Facebook that you are surprised have not gotten turned into companies yet?

[0:48:17.2] MV: So many. I mean, I think one of my thesis is that companies like Facebook and Google and probably Uber at this point live 5 to 10 years in the future. That they've just built

out the software stack that everyone needs, the rest of the world would benefit from, but will just take 5 or 10 years to be commercially available.

An obvious example of this is Confluent, which is an exceptional company that's growing very quickly. Kafka was first developed at LinkedIn and then the team left to start Confluent. Facebook had a similar message bus inside Facebook, I'm sure Google has one as well. Point number one, I think is that Facebook and Google just live in the future and that technology is applicable to a really wide range of people. Point number two is especially as companies mature, they get very – like Facebook and Google, they get large and people just leave and they start companies, or they join other companies. When they land in their new companies, they just miss the infrastructure that they had in their old company.

I think that dynamic also creates this pre-existing market pull for the system. People are looking for this thing that they had at Facebook and don't have anymore. I think there's a lot of opportunity there and certainly, if anyone is bouncing out of Google, or Uber, or Facebook to build a company around one of the shiny toys that only they have internally, I would love to talk to them.

[0:49:47.8] JM: Mike Vernal, thanks for coming on the show. It's been really fun talking.

[0:49:50.3] MV: Yeah. Thank you so much.

[END]