**EPISODE 01**

[INTRODUCTION]

**[0:00:00.3] JM:** Product managers are responsible for guiding the design and overall functionality of software. The relationship between product managers and engineers is complementary. A PM is viewing the product from a perspective that is closer to the customer, so the PM often has the responsibility of navigating high level tradeoffs in the functionality of a product.

Suzie Prince is the head of product at ThoughtWorks and she joins the show to discuss how she navigates tradeoffs as a PM. We also explored communication strategies for how PMs can work productively with engineers and the overall product development process, as well as the product development process in how it may differ in a small company versus a larger company.

Full disclosure, Suzie works on Snap CI, which is a sponsor of Software Engineering Daily.

[SPONSOR MESSAGE]

**[0:00:56.3] JM:** Dice helps you easily manage your tech career by offering a wide range of job opportunities and tools to help you advance your career. Visit Dice at Support Software Engineering Daily at dice.com/sedaily and check out the new Dice Careers mobile app. This user-friendly app gives you access to new opportunities in new ways. Not only can you browse thousands of tech jobs, but you can now discover what your skills are worth with the Dice Careers Market Value Calculator.

If you're wondering what's next, Dice's brand new career pathing tool helps you understand which roles you can transition to based on your job title, location, and skill set. Dice even identifies gaps in your experience and suggests the skills that you'll need to make a switch. Don't just look for a job, manage your tech career with Dice. Visit the app store and download the Dice Careers app on Android or iOS. To learn more and support Software Engineering Daily, go to dice.com/sedaily.

 

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

[INTERVIEW]

**[0:02:09.3] JM:** Suzie Prince is the head of product at ThoughtWorks. Suzie, welcome to Software Engineering Daily.

**[0:02:14.3] SP:** Hey, thanks for having me.

**[0:02:15.6] JM:** Today, we're going to be talking about product management and the interaction between product management and engineering. Let's start off with that. How should an engineer view a product manager?

**[0:02:29.2] SP:** That's a great question. I often think about the other way around, being a product manager and how I view engineers. But I think for engineers viewing product managers, it probably varies somewhat depending on the organization. In general, I definitely think that an engineer should expect the product manager to be a teammate, someone that they work with regularly, somebody equal in the process of software development, product development.

In most cases, I would expect a product manager to be really seen as a leader for the engineer and really be able to answer really vital questions for them, the vision for the product, what is the strategy. On a more tactical basis, what is the number one thing that is should be doing? What can I do to make our product better? Those are the things that I think an engineer should be able to rely on a product manager for and see the product manager as someone who can provide those answers and lead them in the product direction.

**[0:03:35.9] JM:** How should the product manager be interacting with the engineer? What kinds of delegation does a product manager have the responsibility to give to an engineer? Because it's kind of like a different relationship than the relationship between the engineer and the engineering manager.

**[0:03:55.6] SP:** Yes. I would say the way I view it is I view developers or engineers, software engineers, as highly skilled people. These are people with expertise and things that I do not

have expertise in. Particularly, the thing specifically that I'm interested in is obviously their technical skill and their ability to implement things in the right way. I would almost always — I think without exception, even though I work on technical products for technical users. Without exception, I would always rely on software engineers to design the system, to make the correct implementation.

Whether that'd be languages of choice, or software platforms of choice, those kinds of things, I would almost always rely on engineers, or engineer managers to make those decisions. They're definitely outside of what I consider to be a product manager's real.

Also, I think, I also rely on them to talk me through possible edge cases in particular requirements. Although I think that that is a more combined exercise, if I'm thinking through requirements, I obviously should be thinking about that. There's obviously a quality assurance aspect to that as well.

I expect, really, an engineer to give me the gotches, like, "That's not going to work because of this," or "Have you thought about the implications of this versus this?" Those are the things that I rely on for the engineers to do.

Then, the other thing I think is really important, at least for the way that my team works, is to build a system that is up to change later on. I really want to be able to change the product direction depending on the needs of the users. Engineering a system, designing a system that is flexible to changing requirements I think is actually really important and something I specifically look for in the teams that I work with.

**[0:06:11.3] JM:** What are the mediums that you use to communicate with engineers on a regular basis? Are you just using email or do you delegate tasks using Trello? What are the task managers and the communications platforms you use with engineers?

**[0:06:28.0] SP:** Yeah. The team that I work on is very small. We're all collocated. I would say, in general, it's normally face-to-face communication, so I attend daily standups and I actually sit within touching distance of most of our development team, because we're small. We also use Slack. We're a big, big Slack users. As I'm sure, many of your listeners are, and we love that

forum, so that if people do work remotely or they're offline for some time, they're able to sort of follow the team communication. We use that a lot, and I'm part of that team discussion.

Then, we also use a — We actually make a product management tool that's called Mingle. We happen to use that. It's similar to Trello, that's the same sort of agile card wall board. We use that for tracking specific pieces of work. I would use that in combination with a standup to say, "This is the work we have today. Who's available? Let me chat to you about what those things are."

**[0:07:38.2] JM:** I've worked with product managers at the different companies that I did engineering at, but I didn't ever really have much perspective for how the product manager interacts with the rest of the company, with management. I read this book recently called *Chaos Monkeys,* which is about, basically, culture at Facebook. It's about this guy whose company gets acquired by Facebook and then he kind of goes to the integration process. I definitely recommend it for anybody who's an engineer out there.

One of the things he said that's interesting, and that's maybe just a property of Facebook, but he talks about these scenarios within Facebook where you have a product manager and you have the product manager who's talking to a bunch of engineers, and there are scenarios where the engineers lose faith in the product manager, because the product manager has some responsibilities to go out and negotiate with other teams, and then if the product manager fails to view that successful, the engineers lose faith, and then they start to ignore what the product manager says and they do whatever they want. Then, the burden ends up falling on the product manager when that product cannot ship. It sounded like this was a property of Facebook, at least at a certain time where — The product manager didn't actually have that much leverage, they just kind of were talking to the engineers, but at the same time, the product manager became a scapegoat when a product couldn't ship.

I just mentioned that because I'm curious about the management structure and where the product manager sits in the management structure and also how to keep things harmonious with the engineers. Could you put a larger framing on where the product manager sits relative to other people in the organization?

**[0:09:27.4] SP:** Yes. At least I'll do my best from this perspective that I have. I would say you often hear this phrase, or at least product managers use this phrase often to describe themselves, which is CEO of the product. The way I view that particular — Whether I agree with it completely or not, is that there's responsibility there. You're the CEO of something, and I think that, very often, the perspective is that the product manager is ultimately responsible for that particular product.

In the scenario that you talked about, not shipping on time, I think it's very often in the hierarchy that, yes, that would happen, that it would be the product manager's responsibility for doing that. I think it does depend on the organization where they want to say the buck — Where the buck stops. Many product managers would feel that responsibility themselves, because the hierarchy is setup that way. Whether I think that's right or not, is perhaps a different question. I think, generally, what you would have is you would have a product manager reporting to a head of product who would report into either a senior executive leadership team, or directly to the CEO. In many cases, there's just a single product manager reporting to a founder.

Yeah, I think there is a lot of responsibility for those product managers to manage the situations that you described and really have the respect of the team. I think lack of respect, or them not seeing the product manager as an adequate is very problematic in most of the structures that I have seen.

[SPONSOR MESSAGE]

**[0:11:34.1] JM:** Incapsula is a cloud service that protects applications from attackers and improves performance. Incapsula sits between customer requests and your servers and it filters traffic preventing it from ever reaching your servers. Botnets and denial-of-service attacks are recognized by Incapsula and blocked. This protects your API servers and microservices from responding to unwanted requests. To try Incapsula, go to incapsula.com/sedaily and get a month free for Software Engineering Daily listeners.

Incapsula's API gives you control over the security and performance of your application, whether you have a complex microservices architecture or a WordPress site. Incapsula has a global network of 30 data centers that optimize routing and cacher content. The same network

of data center that are filtering your content for attackers are operating as a CDN and speeding up your application.

To try Incapsula today, go to Incapsula.com/sedaily and check it out. Thanks to Incapsula for being a new sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

**[0:12:50.1] JM:** It's kind of a different situation then, most engineers. An engineer can typically , I mean, you can be an engineer who have good diplomatic ties with anybody else in the company and you'll still be fine as long as you're shipping code. I wouldn't want to work at a place where the kind of thing happen on a regular basis. As an engineer, you don't necessarily have to develop diplomacy, but a product manager seems integral to the role.

**[0:13:20.2] SP:** I would say so. I would say, "Yeah." I think if you had a dysfunctional product manager, and I've definitely seen that, it would be a huge risk to your product and you would definitely want to resolve that dysfunction as quickly as possible.

The occasional dysfunctional or — Yeah, lack of certain skills for an engineer are not key to their success and their role. It's probably not as much the problem.

**[0:13:50.2] JM:** When a company is small and it's building its first product, the plan for product management is often quite clear cut. There's a minimum viable product that has to get shipped and then the company wants to get some traction quickly. ThoughtWorks, where you work, is a bigger company. As I understand, the company started as a consultancy. Overtime, you started developed your own products.

In any case, the structure of the company makes it such that there's less pressure to get something out the door immediately, because it's not going to impinge upon the company's existential ability to survive. I point that out just as an example of there are companies and there are different sizes and different structures, how does that affect how the company thinks about product management and the pace of shipping product?

**[0:14:45.2] SP:** Yeah, I think that is a really good question, because depending on what you're trying to do, you would have a different take on quickness, versus completeness, or something like that. I think — I always start with asking the question, "Why are we doing this thing?" and be that, "Why are we building the product in the first place?" or "Why are we building this particular feature?" or "Why are we fixing this particular bug?" The answer to that question helps the product manager or the whole team decide those different urgency versus polished levers.

A good example would be — A really simple example would be if you have a critical bug and that is stopping people from using your product, the full value of your product, you would have urgency around doing that. If you have a onetime shot at releasing. Often, that's what startups feel that they have. They have this one opportunity to go live.

You're going to make a different choice than if you already have a set of existing customers, or driving to a different sort of revenue goal. Perhaps you're not doing it for a revenue tool. I think the question of weight is what is the motivation for the organization to do this at all? That will help you answer the questions about how you should go about doing it, I guess.

**[0:16:23.7] JM:** How do you divide your time between talking to customers of the product and talking to your own team?

**[0:16:31.9] SP:** Yeah, that is a really great question as well. Product managers are always talking about prioritizing, prioritizing for their product, but also talking about their own prioritization skills and their own task management.

I think that the first thing that we always need to do is recognize where our tendency lies. Some organizations just naturally be sort of more insular and spend more time with each other. I would say that, in general, ThoughtWorks tends to be like that, which means that we have a really good inter-team communication and team communication. It's really easy for me as a product manager to spend my time doing that, because it comes naturally to the organization and this demand from my time to do that.

I think that it's important for me to recognize that and then be very specific and very focused on doing the other parts of the role that either may not come naturally to the organization, or the

team, or to myself. I think the key part is recognizing what is my default here and then making space for the other one that is not.

Some other organizations have seen a lot of product managers spend a lot of time with their sales teams, because there'll be a draw from the sales team for the product manager's time. That product manager needs to be recognized, "Okay. I'm spending a lot of time with this particular group, perhaps not with my engineers, maybe not with my customers," only to be very specific about doing that.

The other thing is that when you're doing it, to not let urgent issues influence you. As being this sort of CEO of the product, you're often in demand for what I would call urgent, but perhaps not important issues. You really need to manage, "Am I spending my time on things that are urgent, or am I spending my things on the most important things?" Those two things; recognizing your normal tendencies and asking yourself, "Does it just feel urgent rather than a strategically important work?

**[0:18:55.4] JM:** An engineer is going to have certain key performance indicators that will be things that they are checked against by their manager, maybe at the end of a quarter, or at the end of a sprint. These are the benchmarks for how they measure their success. What about for a PM? What are the KPIs that a PM gets measured against?

**[0:19:18.8] SP:** Yeah, I have to think about this a little. We don't have very formal KPIs for our product managers at ThoughtWorks, but having said that, I do sort of have them in my mind of the kind of things I'm look for in people and whether I would call a product manager successful or not. I think — We talked about earlier this situation you're talking about at Facebook and whether the product ships or not. Whether I would consider that a KPI for a product manager?

I think product success is probably what people would first say. If the product is successful, then the product manager must be successful. If the product is a failure, the product manager is a failure. I don't think it's as simple as that, because I think products are successful for many, many reasons, and they obviously fail for many reasons as well.

I think the thing that I look forward, it maybe isn't exactly what you were looking for in an answer, but it's the one that I have, is that I'm really looking for the product manager to really, really understand the status of the business of their product. Who is using it? Why are they using it? What can we expect from this product and what do we need to do to reach that moment of success for the product?

I sort of think about that as I want them to be very knowledgeable about their product, but also speak very truthfully about how we need to get to the place that they think we need to get to fort he product and what that requires. I guess what I'm saying with that is if I look at the market, and I think in order to be successful, we need to do these sets of things. To do those sets of things, I need these kind of resources, and I need this kind of marketing, and I need these kind of engineers.

I need a product manager to be able to tell me that, and the organization should expect the product manager to be able to lay that out, "This is what it looks like. This is where we're going. This is what I need," so that the organization can make decisions about where their investment lies.

That's a long winded KPI. I don't think it was very succinct, but the thing is I want them to be knowledgeable and I want them to tell the truth and very clearly articulate so that the organization can make correct choices about that product.

**[0:22:00.7] JM:** Are you ever doing design work as a product manager? If you're about to launch a product, or you're about to launch a feature, is it the responsibility of the product manager to do a mockup, or a prototype, or the visual representation of the new product?

**[0:22:19.6] SP:** This can vary, again, a lot. I, sometimes, do them, and I am not a designer and I do not have formal skills in that, but it needs be sometimes. I think that's very much the sort of product manager ethos. It's going to get done. If I don't have a designer or my engineer is looking at me right now and we need to make a decision about something and that's going to take weeks for me to get a mockup, then we're probably going to sketch on a whiteboard together.

I would say that it really does depend. In our organization, we definitely prefer to have experts, expert designers. Sometimes, that doesn't happen. Yes, it has definitely been known that as a product manager I would that. Generally, I'll stick to some guidelines that somebody else, expert in our organization, would have laid out in advance. That makes it a lot easier for me to make those choices. Sometimes it is just gut and creativity, and some product managers of a highly skilled asset. It really does depend on the organization and the specific skills.

I would say if I saw a product manager being very reluctant to do it, that would concern me, because you would not want the lack of that to block progress, and I would want to sort of do it quickly, put it out there and see if it works.

**[0:23:59.4] JM:** Speaking of gut, there is a tension between — If you're deciding what features to focus on for the next sprint or the next product iteration, there's a tension between the data that you're going to get from doing something extremely granular and numerical like A/B testing a feature and the idea of serving users or getting anecdotes or getting these user stories that may be more one off. How do you decide what types of features to focus on? Are you looking at things like A/B tests, or are you looking more at surveying users and getting anecdotes?

**[0:24:43.4] SP:** Yeah, I'm probably doing both at different times for different reasons. It definitely depends on the lifecycle, where in the lifecycle of a product that you are. Probably, at the very beginning, when I'm doing something like an MVP, I would have done a lot more market research that might be surveys in my cases. More likely to be interviews for people, so face-to-face conversation, watching them use other tools, those kinds of things, to sort of build a direction.

Then, I'm probably going to do something small to start off with. Then, I might use A/B testing. It really depends on how quickly I need to get feedback. A/B testing, I think it's really great for some organizations. For other organizations, it can be actually really quite challenging to do it. If you don't get enough data quickly, you sort of have to leave those tests out there for a long time to get any sort of result with any statistical meaning. I think it really depends on the stage of the lifecycle of the product and the kind of changes that you are making and also sort of the amount of bandwidth that you have.

 

My default tendency is to have conversations with people. I feel that gives me a much bigger surface area to understand our users and learn a lot more doing that than doing a survey perhaps, because I'm actually looking at them and I can see their emotions.

Then, we like to put features out there and get feedback about them and only do your A/B testing for very specific cases where we're trying to encourage people to work in a certain way perhaps or to make a choice about a layout for a page in a usability test way. We wouldn't A/B test everything, because it would take too long for us to get feedback. We'd rather just put our first thought out there or our first design out there and use feedback conversations to understand how that went.

**[0:27:08.4] JM:** There's a famous quote by Henry Ford where he says, "If I would have asked customers what they wanted, they would have said a faster horse." Instead, he gave them a car, which is not something they were asking for. I could see this happening in the realm of a product manager where if you just keep asking the customers what they want and looking at the local maxima that A/B tests suggest, you end up with a product that is not a step change. Is it a responsibility of a product manager to think about that step change product development, or is that something that's the responsibility of somebody higher up in the chain? How do you develop products that are not closely tied with whatever is the local maxima of what you're developing?

**[0:28:01.9] SP:** Yeah, I think this is a really great question, because I think it's really quite challenging. I think that one of the things that a lot of us like to do — We're engineers, we have that sort of background. We'd like to think that these are like these process, you put something in and something else come out. I just don't believe that that's true, and you can't just, like you said, ask the customer and they'll give you the answer.

I'm a big believer in it being the product manager's responsibility to look beyond, yeah, the local maxima, particularly because the product managers in our organization and the roles that I have are singularly responsible for a product and we don't have multiple product managers for a single product. When we have that one person, they're responsible for that whole product. I would definitely expect them to be looking for step, change things themselves and not just

following, “Oh, my customer said this, so I’m doing this,” or “My sales team said this. I’m doing it,” or “My found this this. I’m doing it.” I definitely would not expect that.

I think that’s a particular skill for a product manager to hear the market, hear exactly what they’re saying but not just do what they say, but really understand the underlying needs of that market and what the opportunities are and to do that. That’s in general, I would expect.

There definitely are large organizations where you have product managers for specific features and you may have a product manager for a group of products or a head of product. In those cases, that responsibility might lie with those people and is perhaps more likely to come from those people, because that’s just by the nature of their role, have much more visibility into the opportunities that they can take.

I think the key thing is somebody within the product organization in everybody’s business needs to be thinking about that. I think if you don’t think about that, then you’re almost ultimately building something that will eventually be superseded by somebody else. I’m probably quite like the eventually being fairly quickly, because I think so many people build great software. You need to be doing something more than just building what was completely obvious.

**[0:30:41.7] JM:** Do you work in sprints?

**[0:30:43.4] SP:** We don’t, on our team. Some things in ThoughtWorks will either use scrums or they use sprints, or in XP, we call it iterations. Our team do not. We work on more a pool-base model, so I just keep a backlog of items. I prioritize it, sometimes, on a daily basis, but regularly planning to prioritize on a weekly basis. Then, every morning, we just talk about what are the most important things. If people are still working on things from the day before, they’re mostly likely to carry on. Sometimes we do stop work if there are urgent issues coming in. We do a more pool-base model.

We’re very small, so I think it works very well for us, because, again, the few of us just talk together every morning and make those decisions. With a larger team, I would definitely expect a bit more planning required just for coordination purposes.

**[0:31:51.3] JM:** When I was working as an engineer, most of the companies I worked at did sprints, but I feel, overtime, the — Sprints, they are more batch oriented. We're moving towards a world where you can do things more continuously. The idea of a sprint feels antithetical to that. It feels almost — My last job at Amazon when we're doing these sprints, it was like, "This feels forced. I don't know why we're orchestrating around this sprint idea when everybody on the team is kind of working at different pace, they're working on different features, and we can all ship independently of each other."

Do you think the idea of sprints are kind of outdated or getting outdated?

**[0:32:39.2] SP:** Yeah, I think everything you describe is how I feel about sprints. To be perfectly honest with you, I feel like they are training wheels for organizations who were used to working in these ways where people work, basically, waterfall, where things were handed off and the timelines were a lot longer. I feel like, to me, the sprint is a training wheel to change that organization to say, "Hey, first of all, you should all work together. Secondly, you should try to delivery something in a small period of time."

I think, very quickly, if the organization does make that change, and I think it's actually quite challenging for many organizations to make that change. I still see a lot of organizations who say they're doing sprints, but, really, they're still doing some terrible waterfall way of working. I think for those people who never needed the training wheels or quickly understood, "Okay. This is how I want to work." They do feel really restrictive and they have all these sorts of ceremony around them that I personally find really distracting and kind of a waste.

**[0:34:02.6] JM:** Let's talk about the product development process from a specific product point of view. You are focused on the Snap CI product, which is a tool for engineers to do continuous integration. It's also a sponsor of Software Engineering Daily, so I want listeners to understand that. I think it's a good topic of discussion both because there're a lot of people who are moving towards continuous integration, or they are working on developer tools. The market of tools that are sold to developers is really expanding, and I think there are actually a lot of people who are listening to this who are trying to build tools for developers.

 

I want to start with this conversation about how that works. How does building a tool for engineers compared to building things for other customer bases?

**[0:34:55.3] SP:** Yeah, it is different, and I think the biggest thing that I was struck by when building a very developer-focused tool like Snap CI, is I made a lot of assumptions that developers, as developers, understood themselves as users of products. Now, I'm a developer and I use these CO products. I thought that they would understand when talking to me that I was having a discussion with them as a user of a product.

What often happens is they talk to me as if they were a developer on building our product, and so we'll very quickly get into implementations or they will want to understand how Snap is built behind the scenes. You just never have that kind of conversation with anybody, anybody else if you're building even business-to-business software. I build software for project manager. Before, they rarely ask you about those kind of things.

That was very different, is the curiosity or the kind of conversation that developers naturally want to have with me as a product manager for this kind of product is very different. It means that I have to be quite formal or strict when I want to have a very specific type of conversation with someone. I want to conduct a usability test, or I want to ask very specific questions and get answers to them.

I'm finding that I can't do it quite as casually or conversationally with people that I used to before, because I don't get the information that I want, because it goes a sort of certain way, which is normally to the guts of the product which makes sense. I've just learned that I need to be, "Okay. I need to schedule time with you and I'm going have a script and make it very clear that this is the kind of thing that I want to do."

Interestingly, when I do that, it actually makes — I found that it makes those people much more reluctant to want to do it, because, suddenly then they are very — It's very clear to them that they are users of software then rather than an expert in software development. That's one thing that has sort of really, really been different to me. I think as builders of software, we should be aware that are users are maybe not aware that they are users of software. That's one thing.

The second thing is expectations are higher. I feel like the expectations of us as fellow software developers, we set on ourselves are probably higher. Also, that our users set on us than I've experienced before. One aspect of that is really clear if, perhaps, we have a bug, or sometimes you have incidences. Our users ask us a lot more questions about, "Oh! What happened there," or "What was the cause of that?" The really want useful explanations.

Again, what I've learned from that is that they want to learn from us. It's not necessarily that they're, in any way, judging that. It's just like, "Oh! I want to know if that happened on my product, how would I react to that." That's a very different interaction that we have with our users. We definitely want to be a lot more open about how we build software, about how we handle particular cases, because we found that people are hugely interested in that. Then, therefore, more interested in who we are.

Then, the other thing is I feel like they're also more understanding when we do have things like that, explaining to any other type of user why something happened. Let's say, a good example would be Snap is hugely integrated with GitHub, and if GitHub has a problem, which in the last couple of months, there was a denial of service attack. That impacted GitHub. All of our users, if they don't ask us any questions, because they just know what's going on. They're like, "Oh, yeah. This thing is going on. Oh my God!"

If our users were — Other types of users, they'd be like, "What's going on?" and our explanation that that's a denial of service on some DNS server somewhere else. It's not going to make sense to anyone. For our users, they're like, "Oh my God got you. That must be shit?" We're like, "Yeah. Glad we're not them." We have a different conversation then if I was dealing with banking customers, or media customers like I did before. I would probably be having to explain why we rely on this other system, whereas our users are like, "Okay. Yeah, got it."

**[0:39:59.1] JM:** You said about the conversations that you have with developers, is interesting to me. I request a lot of feedback from users for the podcast, and the conversations that I have in the general Slack channel are useful, but the really good feedback I get is often when somebody sends me an email and they'll be like, "Oh, I listened to the show. I'm a fan of the show. I particularly liked this episode. I have a recommendation for you. You should check out this topic."

I'll respond to them, and it'd be like, "Hey, thanks for the email. By the way, I would love any other criticism or any feedback. Is there anything really subtle that you've been thinking about?" Then, they'll often respond with a really detailed email where they will talk about something that I had no idea some problem with the podcast that I had never identified. I think this gets at what you were saying where there are these different kinds of feedback. There are the things that they will say to you off handedly, or in the public, or if you just ask for a broad survey. Then, there are the other things that if you really dive deep with them or you set up some sort of like user session, then they'll really tell you what is at the bottom of their hearts about the product.

**[0:41:14.1] SP:** Yes. Yes, it's definitely multilevel and you sort of have to build that relationship with people, which I think I had never really noticed it before. I think, perhaps, I thought would be easier, because we're all doing the same thing, but it definitely requires this growing and this process to get to it. Then, like you said, it's hugely, hugely valuable when you get to it, and that's what we need to get to really.

[SPONSOR MESSAGE]

**[0:41:49.1] JM:** To understand how your application is performing, you need visibility in your database. VividCortex provides database monitoring for MySQL, Postgres, Redis, MongoDB, and Amazon Aurora. Database uptime, efficiency, and performance can all be measured using VividCortex. Don't let your database be a black box. Drill down into the metrics of your database with one second granularity.

Database monitoring allows engineering teams to solve problems faster and ship better code. VividCortex uses patented algorithms to analyze and surface relevant insights so users can be proactive and fix performance problems before customers are impacted.

If you have a database that you would like to monitor more closely, check out vividcortex.com/sedaily to learn more. GitHub, DigitalOcean, and Yelp all use VividCortex to understand database performance. At vividcortex.com/sedaily, you can learn more about how VividCortex works.

Thanks to VividCortex for being a new sponsor of Software Engineering Daily, and check it out at vividcortex.com/sedaily.

[INTERVIEW CONTINUED]

**[0:43:08.7] JM:** There are a ton of continuous integration tools in the market.

**[0:43:12.9] SP:** Yes.

**[0:43:15.7] JM:** When you started building Snap CI, did you look at the other tools for inspiration, or did you go from first principles, from the personal experience of people at ThoughtWorks. How did you — What was the ideation process like for this highly competitive market?

**[0:43:34.5] SP:** Yeah, that is really interesting. I was absolutely not the product manager when they first started, but I do know what happened. I would say that without exception, the products that we build are built from principles or visions in mind. We definitely don't look at the market and say there's an opportunity there for something for the sake of doing something. It's almost always driven by some more fundamental desire to do something in the software industry.

With Snap, it was definitely deployment pipelines in the cloud is the thing that was driving us to build Snap CI. We already had a CI or a continuous delivery product, it was called Go. That's an on-premise piece of software. We wanted to do something similar in the cloud.

We were definitely driven by a goal and a vision in mind. That being said, we definitely looked at the competitive products, specifically, the cloud products at the time to see what they were doing, to see the kind of paradigms that they were using, the kind of metaphors and the UI, the kind of designs that they have.

I think it's hugely important that even if you are sort of a mission-driven or have a vision for your product, that you are aware of the market, because that's ultimately you're going to be compared to by your users all the time. It's sort of more than just due diligence. You really do

need to know what they're doing even if it's only so that you can answer questions for your customers and those kinds of things.

**[0:45:25.6] JM:** The emphasis on cloud — I think cloud is more of these things, it's like mobile where everybody knows it's a big deal, but even people who know it's a big deal probably underestimate how big of a deal it is. What are the ways that the cloud has influenced how we do CI to the degree where you feel like you had to build a completely different CI product for cloud deployment?

**[0:45:56.3] SP:** Yeah, I think there're two big parts to it that influence — Maybe the first is influence software development, and the second is influence our product. I think the advent of distributed version control versus the sort of on-premise your own source code management really did change the way that people think about software development and definitely get changed the way that people build software for sure.

I can see that very, very clearly when I compare the organizations that I know who are still using something like TFS, or Subversion, and the organizations that use Git and understand how Git flow works and those kinds of things. I think that's the first one. The idea that this could be distributed really changed the way people thought about how any of their tools could work. That just naturally leads to this idea, "Okay. My source is — First of all, I've got distributed source control. Then, I've got GitHub, or GitLab. My source is in the cloud. Somebody else has got it." Then, I think it's sort of naturally leads to, "Okay. Maybe my build service can be in the cloud as well. Maybe I can be doing continuous integration that way."

The second part is I think the sort of growth in this idea that we can all build software and I can start a startup and do these kind of things. I think the idea that more and more people want to work on things themselves or with small groups of people mean that they can't have these big data centers where all these kind of infrastructure is, that they need to be able to get started quickly.

This sort of idea of things as a service really influences the types of products that we have with Snap. You just get sort of used to being sort of cloud native, I guess. I don't know if that's the right word, but your default is just somebody else looks after this and I pay them to look after it

so that I don't have to. Then, it's very easy to see how something like Snap would fit in. Then, things like serverless architectures sort of move on from that and this idea that you can build things without actually having to look after or own the infrastructure.

**[0:48:33.0] JM:** Or write tests, thankfully.

**[0:48:35.4] SP:** I would hope not.

**[0:48:39.2] JM:** Do you have any advice for engineers who are looking to switch to being product managers or vice versa? I think most of people listening to this show are engineers, but some of them probably want to switch to product management, and there probably are some product managers listening who want to switch to engineering. Do you have any advice around those?

**[0:48:59.8] SP:** Yeah, I think that the biggest thing that I look for with product managers when we are looking for them, or I look for myself, is the role is so diverse and it really does depend on different types of organizations. The product manager needs to sort of be well-rounded in a whole bunch of skills. If I'm an engineer moving towards product management, I probably am pretty technically savvy and fairly analytical and perhaps I have a decent amount of experience handling support requests and some urge in issues like that. Perhaps, I don't have experience in customer management, or working with stakeholders, or sales teams. Really, what I would be looking for is someone to assess, "Which of the skills of a product manager am I lacking and where should I get — How can I sort to get those?"

A good way to do that as an engineer, I've often seen, is start to be seen as a leader, so either raise your hand if there's a particular feature and sort of go deep on that feature and really understand it. Try to understand the users of that particular feature and maybe be an advocate for users. I think if you can start to demonstrate that you understand the broader aspects of product development, particularly either in understanding your users, or perhaps you could go in the direction of understanding process. We talked about sprints earlier. Maybe you can start to be an engineer who has opinions about how your team works. That might be another way to grow your skills and be seen as someone who's more well-rounded, because that's what I would be looking for in product management.

With product managers who want to be engineers, I actually think that maybe this is more challenging. I don't know. I think you need to start to build something, and that's what I see happening. I actually have a friend who just — She did a whole bunch of online tutorials, and then she actually went and did a code boot camp. I think that that would be one way to go.

Another way, I think, to go would be trying to spend more time with your engineers, but I think that's kind of hard in your day-to-ay life because you've got to get your day job done and it's not to learn to code. That would definitely be one way to do it, is to perhaps show more of an interest in your day-to-day role.

I think, because the skills are so very specific. I would say, with a product manager, you're definitely going to have to do it either outside, probably be self-taught either outside of work or go do some sort of educational training. Whereas, I think, an engineer to a product manager, probably you can learn enough of those skills on the job. Perhaps, to look for an opportunity in your own organization or as long as you've demonstrated that you have that broadness to your skills.

**[0:52:21.2] JM:** Is there anything that is changing about product management today that maybe is subtle, something we have not covered? Some sort of trend or counterintuitive direction that product management is going in, or is it the same as it's always been?

**[0:52:42.0] SP:** No. It's definitely not the same as it's always been, and I think it really is changing. I think that you touch on a couple of those things that I think are really important to product managers and some of the questions you asked me.

The pace at which we build software is actually very impactful on how product managers have worked and will continue to work. The idea that you can have an idea for something, visualize something in the morning and be shipping it perhaps in the afternoon or the next day is a really big step change for some product managers who were previously working with 12 or 24-month roadmaps. That's hugely influential on the kinds of decision making that a product manager needs to make the kind of attention that they need to pay every day to the state of their product.

 

It's actually quite likely that a product manager who used their own product one week to the next would have a different experience. That's very different even when I first — Not that old. Even when I first started making products, you wouldn't use your product one week and then use it the next week and expect anything other than maybe even a bug to have been fixed. Now, the experience could be very different, because we're just building and releasing software a very different pace. I think that that definitely influences product managers.

The other one is the amount of data that we have available. There's a lot of talk about big data and how data is important, and you and I touched on A/B testing. I think that, sometimes, data — The idea that the data has the answer and the data is going to solve this problem for me, can be quite distracting for product managers. I see them always running tests or turning these tiny miniscule, like we talked about, the local maxima all the time without thinking about the bigger thing that they're trying to achieve.

I think that, yeah, the pace at which we build and sort of the idea that we really still need to stay very high level and very aware of the full landscape, rather than on these miniscule pieces of information that we have, is very important, because it's very easy to get distracted with all of these information that we have and all of the paces going on.

I think that's a very big important part of the product manager's role in the organization and for the team is to keep them very grounded on, "Okay. What is this big thing we're trying to do? How can this piece of information help us now but not let it be distracting for us?"

**[0:55:46.3] JM:** All right. Suzie, thanks for coming on the show. It's been great to hear your perspective on product management.

**[0:55:52.1] SP:** Oh! Thank you so much for having me. It's really lovely.

[END OF INTERVIEW]

**[0:55:56.6] JM:** I am currently looking to hire an iOS engineer to lead the development of Adforprize. This is a full-time paid opportunity. Adforprize is a platform for user-generated

advertising. Every other type of content on the internet has become increasingly user-generated. Why not advertising?

In Adforprize, users create ads to win prizes. Companies post these prizes as an incentive for people to make ads. If you'd like to work with me on recreating how advertising works, send me an email, jeff@adforprize.com. You can also check out a short video explaining how Adforprize works on adforprize.com. Thanks for listening.

[END]

 