EPISODE 748

[INTRODUCTION]

[00:00:00] JM: Robots are making their way into every area of our lives. Security robots roll around industrial parks at night monitoring the area for intruders. Amazon robots tirelessly move packages around in warehouses, reducing the time and the cost of logistics. Self-driving cars have become a ubiquitous presence in cities like San Francisco. For a hacker in a dorm room or a researcher in a small lab, how do you get started with robotics? Certainly there are drones and other small options like AWS DeepRacer, but what is the equivalent of the Raspberry Pi for large human-sized robots?

Zach Allen is the founder of Slate Robotics, a company that makes large human-sized robots that are at a low enough cost to be accessible to tinkerers, researchers and prototype builders. Zach joins today's show to talk about the state of robotics and why he started a robot company. What Zach is doing is quite hard. He's a solo founder who is bootstrapped a robotics company from scratch. He has set up in a strip mall in Missouri and he is a row of 3D printers to help create the parts for his robots. He programs and assembles these robots himself.

Whether you're interested in robots or thinking about starting a hardware company, this episode could be useful to you. It's also useful because it shows just how far you can go with some persistence and some money to bootstrap a company.

With that, let's get on with the show.

[SPONSOR MESSAGE]

[00:01:47] JM: We are running an experiment to find out if Software Engineering Daily listeners are above average engineers. At triplebyte.com/sedaily, you can take a quiz to help us gather data. I took the quiz and it covered a wide range of topics; general programming ability, a little security, a little system design. It was a nice short test to measure how my practical engineering skills have changed since I started this podcast. I will admit that though I've gotten better at talking about software engineering, I have definitely gotten worse at actually writing code and doing software engineering myself.

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

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

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

[INTERVIEW CONTINUED]

[00:03:45] JM: Zach Allen, you are the founder of Slate Robotics. Welcome to Software Engineering Daily.

[00:03:49] ZA: Hey, Jeff. Thanks for having me.

[00:03:51] JM: I want to talk to you today about robots and what you're building at Slate Robotics. Let's start with a state of the robots address. What are the industries today where robots are being widely used?

[00:04:05] ZA: That's a good question. Probably the easiest one that everybody would know would be manufacturing. By far the most successful application of robots would be a lot of where the big money is, is probably like the KUKA robots. These are the kinds of things that are in automotive manufacturing. They're able to pick up whole cars and move them across their workspace within like 1/10 of a millimeter of precision. Probably other areas – Probably the next biggest is probably just research and development. I can't think of anything else that's really had anywhere near the success of manufacturing.

[00:04:43] JM: Why is that? What are the shortcomings of robots today?

[00:04:47] ZA: Well, it's a great question. Everybody has their own opinion about what that could be. I happen to think that price and cost are two of the biggest problems with the state of robotics today. So if we were to analogize robotics to personal computing, back in the late 60s, early 70s, computers weren't very popular. In fact, most people thought nobody would ever need a computer for anything until the price came down to a point where everyday people could go out and purchase one and then engineers got excited about it. They started building useful applications and it was kind of off to the races from there.

So I think if that analogy ends up working for robotics, the big problem today is that they're just too expensive and they're at a point where the engineers can't afford the kinds of robots that they would want to play with and to program on. So that's pretty much the whole point of the company and what we're doing.

[00:05:48] JM: So when did you get involved with robots?

[00:05:51] ZA: Almost exactly 2 years ago. I was a software engineer. So a little bit about my story. I graduated from college with a bachelor's in Spanish, a minor in criminology and in global studies, and basically I was going to go work at my dad's place. He said if I went off and got my MBA maybe I can do something in financing, but I ended up working there doing some low-end IT stuff and I learned how to program and how to build software and I ended up over the three years I was there kind of managing all of their technology and building all of their application, supporting all of it. Working on their databases. So that's kind of how I got into technology.

The transition to robotics was really through my just learning machine learning for fun. I picked up the book called *Deep Learning* by Ian Goodfellow, and that was kind of my introduction to the world of neural networks. I was just thinking one day, "Wouldn't it be great if I could take some of this knowledge and try to apply it on a robotic platform?" Like all good startup founding stories, I scoured the internet for solutions and come to find out really the kinds of robots that I had in mind, you needed to spend anywhere from like 75 grand to half a million bucks. So that's really how I got started and that was about two years ago.

[00:07:11] JM: In surveying the robotics landscape, you came to a perspective that there is an orthodoxy in the robotics industry. What do you mean by that? What is the orthodoxy?

[00:07:28] ZA: So I came across the orthodoxy by building my own robots and publishing them online. I've published a few on like the robotics of Reddit, which is like I suppose where all the clergymen hang out. If it's a personal project, they'll look at your robot and then they'll praise it, but the second it's a product, they've got nothing but negative things to say about it.

Some of the things that they believe to be true that are false, that have allowed me to have whatever level of success I've had so far are things like DC motors aren't very useful for building robotic arms, and 3D printing is not a technology you can utilize for a production product. It's great for prototyping, but you can't actually make a product and have it be any good if you use 3D printing. So these are kind of the things that tend get – That they get the most upset about about our company, and they're just false. Frankly, if they weren't true, then a company like this wouldn't even be possible, which I think is part of the reason why nobody's been able to quite crack the code yet.

[00:08:37] JM: So these different Orthodox beliefs, are there any strong underpinnings to them? Why do people believe the you can't, for example, 3D print components of a robot?

[00:08:50] ZA: So anybody that designs and builds robots mostly are probably coming from a manufacturing background. So they look at the kinds of robots that we're building through that same manufacturing lens where you have to have an incredibly high-level of precision in the parts for the robot and the robot itself. Everything has to be unbelievably rigid and tight, and 3D printing is not known for its high degree of precision. I mean, these kinds of things tend to be machine down to – They talk in terms of one-one-thousandths of an inch is typically the metric they use for any precision machining.

So 3D printing is just kind of strange for them. To be fair and to give them credit, you couldn't use this technology for the manufacturing market. Our robots today would not be well-suited for intense industrial application where you need a really high-level of precision with each movement where you pick things up, where you place them down. Everything has to be super, super precise. So they're kind of wrong and also correct at the same time. So really, I think the key innovation of this company has really been more in just seeing a market where most people don't.

[00:10:12] JM: That market specifically is robots for tinkerers.

[00:10:17] ZA: Right. Mostly, machine learning engineers. I mean that's been – If I think back to why I started the company, that makes sense. For some reason it took me really a year or so to come back around and realize, "Oh, yeah! The reason I'm building this at all is because I was interested in machine learning and I wanted to apply it myself, apply robotics to machine learning."

So by selling to this market that people aren't even sure exist, they're are also more willing to put up with the fact that it doesn't have the 1/100ths of a millimeter of precision in its movement and they tend to be a little bit more optimistic about what software can do to account for that fact. They're also not doing manufacturing. So it's more for fun in education. So our product seems to make sense in that regard.

[00:11:08] JM: So if I'm a machine learning researcher today and I want to get into robotics, maybe my options are I could pick up a drone from DJI. Maybe I could get a Lego Mindstorms kit. I guess Amazon has these new little cars that drive around or something that you can program. What are the other options for machine learning? People who want to get into – Oh! There's also the Nao robots, the N-A-O. I think those things are useful.

[00:11:40] ZA: Right.

[00:11:40] JM: Tell me about the landscape of the machine learning robots.

[00:11:45] ZA: That's a good question. Do you know off the top of your head how expensive the Nao robots are?

[00:11:50] JM: I have no idea actually.

[00:11:51] ZA: I think they range like at the low-end at like $8,000.

[00:11:55] JM: Really?

[00:11:56] ZA: Yeah.

[00:11:56] JM: These are these little like 2-foot tall little robots, right?

[00:12:00] ZA: Right, and they don't have – I think through their fingers on the hands can move, but they're not designed to really pick anything up. So that's a good example of – It's really expensive. That would be a big investment for just an engineer to go out and purchase for themselves, and you're still really limited in terms of the applications that you can develop on a platform like that.

In terms of the landscape, I mean you can go on the low-end on Amazon. They do have really cheap robotic arms that are going to be built for like a desktop. They're going to be pretty small. I'm sure there's options out there for drones that might be useful for development. I can't think of any off the top of my head that are like really designed for development specifically, and I don't know what it would take to get like a DGI drone to get that to where you're hacking on it yourself.

I mean, really there is this huge gap in terms of like you're either going to spend like $200 for one of these tiny desktop arms or you're going to spend like 10, 20, $30,000 for some of the high-end stuff, and the cheapest like proper human-sized mobile development platform on the market besides what we're building, you're going to spend 50 or $60,000.

I don't know if you've heard of Fetch Robotics. They build a mobile manipulator. That's a pretty good little robot, but they don't show the price in the website, which isn't a very good sign, and I think they run like 60 or 70,000.

[00:13:38] JM: Mobile manipulator. So this is probably – What is it like? A platform where it could roll around and then it's got like a robotic arm on top of it? What's a mobile manipulator?

[00:13:50] ZA: I mean, that's a term that can be used to describe anything that can manipulate the world around it and is also mobile. So sometimes you'll see like a Roomba with like a small little robotic arm on top of it, and that's technically a mobile manipulator. Generally, I see the term used to apply to things that are human-sized that do have an arm on top of it. So like with the Fetch Robotics mobile manipulator, it's around base that maybe is like 500 millimeters in diameter. It probably is 4-foot tall. It has a head with a camera and then it has one arm hanging off the front of it. Then there's a flat workspace on the top of it. That kind of structure is really what I have in mind for the company and the kind of products that I want to build.

If I'm an engineer that's wanting to really play on something and have a really useful platform that I can develop applications on, something that's tall that has a workspace on it, that has a proper like human-sized arm, has a head with a camera, and then a base that can move around somewhat quickly, then your realm of applications, the total number of things you can build on there is pretty endless. That's what I think is awe-inspiring.

So really, the idea is to try to build those kinds of platforms that right now at the very low-end $60,000, at the high-end, a couple hundred thousand dollars trying to build those kinds of platforms, but for a price that I think is will at least open the eyes of engineers a little bit within the four or $6,000 range.

[SPONSOR MESSAGE]

[00:15:41] JM: How do you know what it's like to use your product? You're the creator of your product. So it's very hard to put yourself in the shoes of the average user. You can talk to your users. You can also mine and analyze data, but really understanding that experience is hard. Trying to put yourself in the shoes of your user is hard.

FullStory allows you to record and reproduce real user experiences on your site. You can finally know your user's experience by seeing what they see on their side of the screen. FullStory is instant replay for your website. It's the power to support customers without the back and forth, to troubleshoot bugs in your software without guessing. It allows you drive product engagement by seeing literally what works and what doesn't for actual users on your site.

FullStory is offering a free one month trial at fullstory.com/sedaily for Software Engineering Daily listeners. This free trial doubles the regular 14-day trial available from fullstory.com. Go to fullstory.com/sedaily to get this one month trial and it allows you to test the search and session replay rom FullStory. You can also try out FullStory's mini integrations with Gira, Bugsnag, Trello, Intercom. It's a fully integrated system. FullStory's value will become clear the second that you find a user who failed to convert because of some obscure bug. You'll be able to see precisely what errors occurred as well as the stack traces, the browser configurations, the geo, the IP, other useful details that are necessary not only to fix the bug, but to scope how many other people were impacted by that bug.

Get to know your users with FullStory. Go to fullstory.com/sedaily to activate your free one month trial. Thank you to FullStory.

[INTERVIEW CONTINUED]

[00:18:02] JM: I was thinking about some applications that you could do with your robot. It's not a custom robot. The idea is it's going to be kind of a more mass application or for researchers at least. It's a human-sized robot. It's about – I don't know. What is it 5-feet tall, 6-feet tall?

[00:18:19] ZA: Yeah, it's about 5, 5-1/2-feet tall.

[00:18:21] JM: 5-1/2-foot tall, beautifully designed robot with a couple robotic arms and a four-wheeled platform on the – That rolls around, and you could do things like maybe make a robot butler out of it. You could have a robot that drives around your house and says, "Danger!" or something, or like shows you security camera footage or something. What are some potential applications that you think people could tinker around with this machine learning researchers?

[00:18:51] ZA: Yeah. There's a spectrum of applications where on the low-end of that spectrum you have the kinds of applications where you don't need any new technology to be developed. You don't have to do too much heavy lifting. That would be things like just building a web interface for it so you can login to robot and drive around and check in your house or your office or whatever. To like the high-end, where how do we train the robot to be able to cook or clean, set the table? Those kinds of things are going to be really, really sophisticated applications, and I don't have any experience building that kind of stuff. But I look forward to trying to build those kinds of applications.

I mean, I don't know what the killer app for personal robotics is going to be, but it might be something like cooking you dinner, doing your laundry, setting the table, making your bed. Maybe someday somebody' is able to crack the code with that and that ends up being kind of the VisiCalc of personal robotics. I think that's how the future unfolds here.

So you've been hacking on these. You've gotten a robot fully built. I want to know about the process for doing that, because I think you got the cost down to a pretty good place. I was looking at the site today, it was something like 6,400 bucks or 6,000 bucks. How many of the components that you built this robot out of were you able to take off-the-shelf?

[00:20:24] ZA: So pretty much everything plastic on the robot is custom designed and manufactured, including a lot of the metal components, like the hubs and motor mounts and stuff like that. In terms of off-the-shelf components, I mean things like motors, the sensors for angle detecting, the microcontrollers, the camera in the head, the computer that we use on the robot. It's an NVIDIA Jetson. Those are all off-the-shelf parts. Yeah, with the TR1, there is a ton of custom components, and that's part of the reason why we've had to keep raising the prices for it, because it just takes a very, very long time to manufacture one of these.

[00:21:08] JM: You don't have a background in manufacturing. How did you learn to build a robot? Did you read textbooks or stack overflow? What was your process?

[00:21:19] ZA: Mostly playing around with 3D modeling software, like the CAD software, like Autodesk Inventor and Fusion 360 and just kind of trying to come up with ideas. I mean, the easy part is like building the base, designing that, because that's just a couple of wheels around the frame, even up to the head and the neck. Like you can pretty easily wrap your head around how things like that would work.

The really hard part of any good robot that's along this kind of category is the arm design, and that's ultimately what takes so long to try to build. So I started – With the TR1's ones arms, I started with trying to work with stepper motors, and I spent a long time trying to get that to work, but stepper motors don't have enough power for the size of the arm. So it was just a ton of trial and error.

I ended up trying to take an open source robot arm design, just like, "Okay, if that works, I'll just throw it on the robot." So I've taken that design and I just modified the ever living crap out of it to where it's not even resemblable to the original design anymore, because you have to fit in all the motors. I mean, that was a design that was built for stepper motors, but I had to use DC motors that have a lot of torque and I also have to fit in the angle encoders everywhere. So it's just trial and error. It took me the better part of a year to get something that even worked at all. I guess just relentless commitment to try to figure it out.

[00:22:51] JM: And this is just you working on it, right?

[00:22:54] ZA: Right. So I first started working on it beginning of 2017. I was employed through August of that year when I incorporated the company and went full time with it. Then probably spent about eight months in my garage working with that. Got some preorders and was able to get an office space where I'm at currently.

[00:23:15] JM: And this is mostly bootstrapped, correct? You haven't raised any money for it?

[00:23:20] ZA: Right. Yeah. It's mostly been funded by orders, and my dad and I have also put 10, $15,000 into the business.

[00:23:30] JM: And that capital has gone towards buying 3D printers, but you've got like a wall of 3D printers and this – I guess, what is it like? A storefront that you've revamped as a machine shop?

[00:23:42] ZA: Yeah. So two doors down, we've got McAlister's Deli. The one right next to us is a vape shop, which I fairly certain they sell some sort of illicit substances out of them. Then to the right of us, we've got some sort of a family doctor and a salon. I mean, it's just a strip mall and it works.

[00:24:03] JM: Right. So paint the picture of this machine shop. So you've got – And there's some great videos on YouTube that you've taken of your machine shop, which is very clean by the way. You've a very clean tinkerer. You've got this wall of 3D printers. You've got some computers. You've got some system monitors on the wall, and that you've just got these like 6-foot tall robots that are in there and like random robotic arms around there. What else do you need to tinker on robots? Do you need some trays full of processors and motors and stuff, and where do you get this stuff?

[00:24:39] ZA: Well, if you have a 3D printer, that's a big part of what you need. Then on top of that, just finding somewhere to buy the motors and the sensors and you need some sort of onboard computer for the robot. You need microcontrollers. I mean, all of those you can buy online. What I did was just design the parts out on the CAD software and just think about, "Okay. Well, I probably need this motor." So I'd go out and buy that motor. I didn't know anything about computing torque requirements or there's some coefficient of friction, but I don't know what number to put there. So it's just a lot of trial and trying to find a motor that works. So I've bought a bunch of motors that don't work to ultimately find the one that does. Yeah, just a lot of time to figure it out. I mean, really just a computer and a 3D printer, you're very much there.

[00:25:35] JM: Describe some of the mistakes that you've made in your process of learning to build robots.

[00:25:42] ZA: So I'm working on right now, I'm hoping – In fact, by the time this podcast launches, I will have launched the next version of our product, the TR2. That is the culmination of all the mistakes that I've made in both the design and the manufacturing of the TR1. So I've got a long list of things that I could talk about there.

Number one is just how complicated the design is. I mean, I don't know why I didn't see this, but there's like hundreds of custom parts on the TR1. Every new part that you put on to a design is something that has to be kept up with, has to be manufactured. That's one extra thing that has to be assembled. It's really complicated. So I've cut just the number of parts down with the next robot by like 10%. So that's one thing, just complicated design.

The arm design right now is it kind of ties in with the complexity, but I started very much with like, “Here's how I want the arm to look.” I want it to have this number of joints and I want it to be this long, and I kind of want it to almost model a human arm. Then, okay, now I’ll just try to figure out where to put all the sensors to put all the motors and all that kind of stuff. There's no room for the wires. The wires have to be run outside of the arm.

So with TR2, I started with the actuator. How do I design the perfect actuator that has the motor, the sensor? Has all the electronics. Has some other features, like torque sensing built into it. It has a metal tube going through the axis of the actuator so you can run all the wires through it. So I’ve started with a crate actuator design and work backwards from there. Okay, now how do build an arm given this actuator? That just makes for a much, much simpler design. It makes it much more effective and you’re able to build in a lot of features without adding much complexity. It’s more distributed in terms of the design, because all the complexity is just built into the actuator, but the actuator is reused.

The early versions of the robot I sold for like – Most people that have ordered one got this thing for like $3,000, and that's just like way too low. We've just been shelling out cash trying to fulfill those orders and like we haven't made any money on those early orders. I mean, that was just a stupid mistake, mostly because I didn't know how long it would take to fulfill these. It really required like a team of like four or five people working around the clock to maybe make like one every two weeks, which is not a recipe for success. But it gave me the opportunity to learn those lessons and apply it to future products. Yeah, I don't know. Besides that, those are probably the big ones.

[00:28:29] JM: So once somebody has a robot, a TR1. That's the robot you've built. How do they program it?

[00:28:37] ZA: You have the world of options at your disposal. I guess there's an easy way and a hard way. The easy way would be to plug into the ROS ecosystem. The TR1 is fully supported on ROS and has most of the low-level stuff and low-level integrations built out, including how do you communicate with the microcontrollers that actually move the motors and actually read the sensor data, including there's a the robot description format that describes where all the joints are, describes the geometry of the robot. That's all been built out. So you can fully visualize the data, and there's also a package called Move It that we've integrated with that allows you to do inverse kinematics really easily. Basically, you just describe your goal position that you want the end effector to be at given a certain orientation and then Move It will calculate how to move the arm given the current state to that particular position.

So by building your application on top of ROS, you get all of these features on top of it where you can visualize it and you can do inverse kinematics and that kind of thing. So there you would just need to build a ROS node. That's going to be your particular program. The way ROS works is you're subscribing to certain ROS topics and then you're also publishing to other ROS topics. So let's say if you take a – Let's take the web interface program. You're going to build a server that's going to accept commands from the browser client, and the net server is just going to publish data, like if I press the arrow key, it's just going to publish, move the base forward, move the base backwards by just publishing on those ROS topics. So that makes building higher-level applications without having to get too worried about how the low-level mechanics work. That makes it really easy, and ROS is just this kind of message passing bus built on top of a HTTP. So really, even within the ROS ecosystem, you should be able to work with just about any library that you want to work with, or any programming language you want to work with. But C++, Python, Node.js. I mean, these are all pretty well-supported within ROS.

[00:30:59] JM: Ross has been around for more than 10 years. It has all these robotics middleware in it. Can you talk more about the ROS ecosystem and is there a broad palette of open-source ROS things to take off-the-shelf?

[00:31:16] ZA: Yeah, there's a lot a lot of things. I had mentioned Move It. That's a package that we didn't build, that we are just able to borrow, because it's open source. Just given the robot description, you're able to integrate your robot in all of your specific hardware into Move It. That's something we didn't have to figure out how to do inverse kinematics or anything like that.

All of the visualization software is a program called rvis that will visualize your robot and show the joint state and just the overall state of the robot. So one thing is you're programming something, and let's say the arm isn't quite moving to the position that you want. You can pull up rvis and you can see what the robot thinks its current state is and you can compare that to what you actually see in the physical world is what the actual state is. That helps with debugging. Maybe you need to go back and recalibrate certain joints or something like that.

There's also the simulation side, which is really cool. Gazebo is a visualization package that plays nicely with ROS, and that's something I've used before. That way you're able to build programs that work on the simulation and you're able to work within that environment first. You don't have to risk breaking something, whether that's on the robot or in the physical world. It's really easy to restart your program and restart the environment. So it often is much easier to work first within simulation, and then port that same program over, because you're just using the ROS topic interfaces and you're using ROS. You're able to take that same program and just run it directly on the hardware without really doing anything different other than starting up your hardware interfaces instead of the simulation.

So there's just a lot of stuff like that, and that's probably just kind of scratching the surface. There's a whole bunch of other ROS packages that I don't even know about I'm sure would be really useful for whatever you're building.

[00:33:20] JM: Let's say I've bought a TR1. I know some machine learning code. I'm familiar with writing machine learning software, and let's say I want to write code to make my TR1 a personal robot butler. What will be my steps to programming the TR1 to make a butler?

[00:33:39] ZA: That's a really good question. I guess that's ultimately kind of an architecture question. You've probably got a million options there. If you're building on top of ROS, you probably would be well-suited to build a program or build a node that's going to listen to voice commands and that one node is only worried about just listening to has somebody commanded the robot to do something. That's going to involve natural language processing, a whole bunch stuff that I don't know all that much about. Then you could have your other node that's going to be doing your computer vision stuff and maybe given – If that node notices something, then it's got to pass a passive message on to another program that's just you worried about – Maybe it passes a message that says, "Hey, there's a cup on this table," and then this program over here is going to take that information and actually move the arm to that position. Building a really general-purpose platform like that. I mean that’s like I think probably the next really hard problem to be solved. It's just going to take a lot of work.

I’ve seen people work with – There’s one customer that I think he's been trying to do reinforcement learning on the robot. I don't know what degree of success he's had so far. But with that, I mean, that's a much simpler problem where you just spin up one node and you're going to train the data either using an Xbox controller or something where you're going to try to show the model, “Here’s successful cases,” and then given that data, you can run the program through and maybe you have like just a button you press that says, “You succeeded,” or “You failed,” and that’s how you can do it. Gosh, there's a million different ways to skin that cat.

[00:35:22] JM: Have you written much software for the TR1 yourself or have you more just try to present the minimum amount of software on there? What kind of software have you had to write?

[00:35:34] ZA: Most of what I spend my time writing is going to be more on the low-level end building out the – One example is the interface between the main computer and the microcontrollers, the NVIDIA Jetson and the Arduino's, and how do I interface properly between those two and make sure that if the data gets dropped, how do we handle it? Implementing checksums and that kind of thing. I’ve been spending a lot of time the past week or so working on that, also providing the support for Gazebo. I mentioned Move It,, all the visualization stuff. I mean, that all is going to be – I have to write a description file of the robot that has to be specific to the hardware and describe all the joints and making sure all of that works well and that the grippers work and that kind of thing.

So mostly what I've spent my time on is the low-level stuff. I've worked on building – I mentioned the Xbox controller where you can take the Xbox controller, run the demo and then he can control all of the joints with that Xbox controller. I’ve mentioned the browser interface for controlling the robot. Those kinds of things are mostly what I’ve worked on. But I am looking forward to overtime once that low-level stuff gets towards working really, really well, focusing more on the high-level stuff. I don't know if – I would prefer to not have to rely on our customers to build the killer app. If we can build that ourselves, that would be great. Because I don't want people buying this robot thinking that they are essentially just working for us. So whoever comes up with it, great, but that's going to be a big part of probably what, as I transition over the next year or two, spending a lot more time on is the software.

[SPONSOR MESSAGE]

[00:37:30] JM: Clubhouse is a project management platform built for software development. Clubhouse has a beautiful intuitive interface that's made for simple workflows or complex projects. Whether you are a software engineer or a project manager or the CEO, Clubhouse lets you collaborate with the rest of the product team.

If you're an engineering-heavy organization, you will love the integrations with GitHub and Slack and Sentry and the other tools that you're feeding into your issue tracking and project management. To try out Clubhouse free for two months, go to clubhouse.io/sedaily, and it's easy for people on any team to focus in in on their work on a specific task or a project in Clubhouse while also being able to zoom out and see how that work is contributing towards the bigger picture. This encourages cross-functional collaboration. That's why companies like Elastic and Splice and Nubank all use Clubhouse. Those companies have all been on Software Engineering Daily and we know they are competent engineering organizations.

Stay focused on your project and stay in touch with your team. Try out Clubhouse by going to clubhouse.io/sedaily and get two months free.

Thank you Clubhouse for being a new sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:39:00] JM: There's a large graveyard of robotics companies that have raised a lot of money and failed. I've seen you write about some of these. What are some of the lessons that you take away from these companies that have raised 100+ million dollars trying to build robotics companies and failed?

[00:39:20] ZA: Rethink Robotics just shut down. A big problem that robotics companies have is that they play in markets that they know nothing about. So if I am robotics PhD and I'm going to go off and try to raise some money to start a robotics company, typically people will go, “Okay. Well, maybe I'll try to build a robot for the elderly. I’ll try to build a robot for healthcare. I'll try to build robot for manufacturing.” You're somebody who spent your entire life studying how to build robots, which is awesome, but you don't know anything about these markets.

It's fairly naïve to just come in and say, “Well, I don't know anything about this market, but I’m going to try to build a product that the market cares a lot about.” I think that's typically the fundamental error of robotics engineers, is just trying to build a product within a market they don't know anything about. I mean, I think that's really what happened at Rethink Robotics, is that they built I think some of the best robots, even for the price to date, but they didn't have the level of precision that manufacturing required.

They had these series elastic actuators that were great, because they can measure torque and if they bump into something, they can reverse. So it’s very hardware-focused solution to how do I deal with bumping into something made for great robot, but the customers that use these, they want something with much higher precision. The series elastic actuators just can't provide that. So that’s just as an example of they built a product and then they went off and tried to find the solution to it.

Rethink had universal robotics come along and build a robot for the similar price, but just specs that blew it out of the water. I think that's ultimately what killed Rethink Robotics. But there's nothing wrong with trying to build robots for healthcare or for the elderly or for paraplegics. I mean, these are the things that like sound really nice, but there is a sense of arrogance to think that you know anything about what it's like to be elderly or to be paraplegic. So to build a product like that that they truly care about, I don't know. You would probably need a lot of consultation and – I don’t know, it would take a lot to know truly what that market wants and desires so you can really fill a true need and it's not just another kind of me to product.

[00:41:55] JM: So the idea is rather than trying to build a domain-specific robot, you're trying to offer a platform that may be domain experts, like some elderly care expert could build robots on top of.

[00:42:14] ZA: Yeah, ultimately. That's the kind of person that you want building those kinds of companies. Somebody that knows a lot about what it's like to be elderly. Somebody that's worked a lot with people who are elderly, and ultimately that's probably who you would want to be making all the decisions about the kinds of features and specs of the product, not just the engineer who doesn't know anything about the market, because there's a big difference between a great robot and one that actually serves a need.

[00:42:41] JM: The element of price point that you talked about earlier, I've heard you say that there are these companies – These companies are building mostly expensive robots, and the idea is that they're going to reduce the cost over time and your position is that you should instead start with a low-cost robotics platform and maybe lever up into higher value products over time. What makes you so sure that that is a good way to go about it? Because it seems like it's not – I mean, I get the demographic you're going after is machine learning researchers may be at a university where they couldn't afford a $50,000 robot, but there also are big industrial arenas where maybe they would they would be willing to pay $50,000 for a robot. So why start with that low-cost platform?

[00:43:36] ZA: I think just practically it's much easier to go with – If I have a cheap robot, it's much easier to go in and upgrade the motors, upgrade the sensors, upgrade maybe plastic parts to metal parts. I think that's a lot easier to move up like that than it is to take a really expensive robot. I mean, you're not going to take a Boston dynamics Atlas and just say, "I will just throw some smaller motors in there." It's not going to work. The robot just won't work after that. So I think there's a practical element to that.

You can also ask why wasn't it IBM that was at the – I think they did pretty well, but why weren't they at the forefront of the personal computer revolution? I think it was Steve Wozniak went to HP 10 times saying, "Here's the Apple2 design or the the Apple1 design. Please take it," and they just ignored him, and then he went off and started the company and the rest is history. Why didn't they take that? They had that opportunity right there in front of them.

The big guys with a lot of money – I mean, first of all, I can't build a high-end expensive robot, because I don't even have the money to like even build a prototype for a half a million dollar robot. So I can't go up. They could go down the market, but they're going to see the same thing that HP saw with Steve Wozniak, which is, “This is a tiny market. We’re going to invest a couple million bucks into this market that maybe is only a couple hundred grand a year.” That doesn't make any sense.

But I talked about really the true innovation of the company is just purely marketing, where I see this market as something that's going to be growing at a tremendous rate over the next 10 years and I believe that be true, but the big companies either don't see that or they don't believe it to be true. So that's probably why the big players don't come down the market and play in the space that I do, because it just doesn't make sense.

[00:45:35] JM: I want to know about your psychology as you've been building this company, because I’ve seen these videos of you just hacking in your rented storefront space and it’s kind of epic. I mean, you're just building a human sized robot by yourself. I mean, I've been recording a podcast by myself for a while, and the early days in particular were a little bit psychologically stressful, because I was just learning how to do this thing that I didn't really know how to do, and that was just a podcast, and you're doing a robot. Can you take me through some of the psychological challenges and how you've overcome them, and what your state of mind is today?

[00:46:20] ZA: Yeah. With any startup, and I'm sure you've experienced this a bunch, you have times where you’re just to the moon as far as just things are going incredible and you're just on cloud nine. Then on the other end, you have days where it's just tremendously discouraging and you’re asking yourself, “Is this going to work?” So I've definitely had a whole bunch of days like that.

I mean, at this point I've had those experiences and it continues to work out one way or another. So I do still to this day have days where I come in and just looking at the robots and being like, “This is kind of the craziest startup idea and I don't know how this is going to work.” But every time I’ve said that, it's worked out one way or another. So I’m just assuming the train is going to keep going. Another huge motivating factor is that there doesn't seem to be anybody else that's working on this problem. There doesn't seem to be anybody else that believes the kind of things that I believe about how the future of robotics is going to play out and the kinds of robots that we can build today and the kinds of robots that we need to build today. There’s nobody working on these sorts of – I mean, there's nobody building the kind of products that I'm envisioning and that I'm building today. If I gave up, really the easier route would be just to give up and go get a normal job. Life gets a lot easier if I did that, but the most discouraging thing that would come from that is just knowing that that would just push off the development of personal robots just that many years out into the future.

How much longer is it going to be until someone else comes along and learns the things that I've learned and is willing to stick with it? So maybe if another company comes along one day and really truly understands the things that I know and is able to do it better, maybe the world doesn't need me for this. That's the number one thing that keeps me going.

[00:48:20] JM: How are you finding customers?

[00:48:22] ZA: Reddit. The first customers that I got, I posted on the machine learning sub-Reddit and I got a lot of feedback from that. So that's where I first handful of order reservations came from, and that’s going to be the strategy going forward, is just make compelling products and just post them online. If it really grabs people's attention, if it’s the kind of product, I just trust that it will get the recognition it deserves.

I mean, the easy thing is finding people that are willing to listen to me. One of the harder things is finding the right customers. Like I've been featured in our local newspaper, our local news channels and stuff as well, but those aren’t the right customers. The machine learning engineers are harder to get in front of, but I think if you make a compelling product and you show them what's really possible, I trust that we’ll be able to generate enough sales off of that.

[00:49:20] JM: Well, Zach, it’s been really interesting talking to you, and I am inspired by what you’re doing. Is there anything else you'd like to add about consumer robotics or just your experience or what you’re doing at Slate?

[00:49:34] ZA: Yeah, I mean I hope a lot of people are willing to ride this train with me, and all I really hope to do is just evangelize how I think personal robotics is going to come about and that it's not that far away, and that if we can build the right hardware, it can be affordable enough and we can get a bunch of engineers who are excited about that both as potential employees at

Slate Robotics and also as customers, then we can really begin to build the kind of future that we all hope and dream of one day. One where you don't have to go home and cook dinner. You don't have to make your bed. You don't have to pick up clothes. You can just focus on what you know best. You can just focus on your craft, whether that’s software engineering or machine learning or whatever, and you don't have to deal with as many distractions. That's just the future that I'm hoping to build and it's going to take a ton of people, at a lot of energy to get there. So I just hope people will join us on that and check out our website and come be a part of it.

[00:50:39] JM: Very inspiring. Okay. Well, thanks, Zach. I appreciate you coming on the show.

[00:50:42] ZA: Yeah, thanks a lot, Jeff.

[END OF INTERVIEW]

[00:50:46] JM: OpenShift is a Kubernetes platform from Red Hat. OpenShift takes the Kubernetes container orchestration system and adds features that let you build software more quickly. OpenShift includes service discovery, CI/CD built-in monitoring and health management, and scalability. With OpenShift, you can avoid being locked into any of the particular large cloud providers. You can move your workloads easily between public and private cloud infrastructure as well as your own on-prim hardware.

OpenShift from Red Hat gives you Kubernetes without the complication. Security, log management, container networking, configuration management, you can focus on your application instead of complex Kubernetes issues.

OpenShift is open source technology built to enable everyone to launch their big ideas. Whether you're an engineer at a large enterprise, or a developer getting your startup off the ground, you can check out OpenShift from Red Hat by going to softwareengineeringdaily.com/redhat. That's softwareengineeringdaily.com/redhat.

I remember the earliest shows I did about Kubernetes and trying to understand its potential and what it was for, and I remember people saying that this is a platform for building platforms. So Kubernetes was not meant to be used from raw Kubernetes to have a platform as a service. It was meant as a lower level infrastructure piece to build platforms as a service on top of, which is why OpenShift came into manifestation.

So you could check it out by going to softwareengineeringdaily.com/redhat and find out about OpenShift.

[END]