EPISODE 630

[INTRODUCTION]

[0:00:00.3] JM: If you operate a restaurant, you want to know how many people are inside your restaurant it any given time. You also want to know your occupancy rate if you operate a movie theater, or a coffee shop, or an apparel store. Knowing how many people are in your building can answer several business related questions. Do you need to unlock an additional entrance to your building? Should you open another store because you have so many customers, or do you have so few customers that maybe you can get a smaller building?

So this might sound like a simple question. But how do you solve the problem of counting people inside of a building? A naïve approach to counting people is to use video cameras and count the number of people entering and exiting a building. Machine learning algorithms can be run on this video feed and machine learning algorithms are pretty good at classifying humans, but the downside of this is that you would have to put cameras everywhere that you would want a people counter, and there are many situations where you would actually want to count the number of people where a camera is not socially acceptable.

What if you wanted to count people in a privacy preserving way? What if you want to obscure any identifiable traits of a person that you were counting? Density is a device for counting people. It sits above a doorway and counts the people who are entering or exiting the building. This is a fascinating engineering problem and I was glad to have Andrew Farah, the CEO at Density, come on the show today to discuss it. He explains why the problem of counting people is harder than it sounds and how the Density people counting device works.

Before we get started I want to mention that we are looking for a few roles to hire for Software Engineering Daily. We're looking for an engineering journalist, a researcher, a videographer, a writer, and if you're interested in these, you can find them along with other jobs at softwareengineeringdaily.com/jobs. We would love to have you as an applicant.

[SPONSOR MESSAGE]

[0:02:11.7] JM: Azure Container Service simplifies the deployment, management and operations of Kubernetes. Eliminate the complicated planning and deployment of fully orchestrated containerized applications with Kubernetes. You can quickly provision clusters to be up and running in no time while simplifying your monitoring and cluster management through auto upgrades and a built-in operations console. Avoid being locked into any one vendor or resource. You can continue to work with the tools that you already know, such as Helm and move applications to any Kubernetes deployment.

Integrate with your choice of container registry, including Azure container registry. Also, quickly and efficiently scale to maximize your resource utilization without having to take your applications offline. Isolate your application from infrastructure failures and transparently scale the underlying infrastructure to meet growing demands, all while increasing the security, reliability and availability of critical business workloads with Azure.

To learn more about Azure Container Service and other Azure services as well as receive a free e-book by Brendan Burns, go to aka.ms/sedaily. Brendan Burns is the creator of Kubernetes and his e-book is about some of the distributed systems design lessons that he has learned building Kubernetes. That e-book is available at aka.ms/sedaily.

[INTERVIEW]

[0:03:46.4] JM: Andrew Farah is the CEO at density. Andrew, welcome to Software Engineering Daily.

[0:03:50.5] AF: Thanks for having me, Jeff.

[0:03:52.1] JM: You are solving the problem of measuring occupancy within a building. Why is this an important problem to solve?

[0:03:59.9] AF: Well, we build an anonymous people counter that gets mounted above an entry way, kind of like a showerhead and we essentially get deployed into really large corporate offices so that employees can figure out which rooms are busy and which rooms aren't and so that the buildings can be designed better. It turns out that like most buildings have been built on a guess. So this is kind of the first time that technology has been cheap enough and smart enough to be able to really understand human behavior within space. I think that has some really cool, like positive consequences on the actual design of space.

[0:04:31.3] JM: So if a business does not know their utilization of space, how can that end up costing them money?

[0:04:40.3] AF: Well, to put this in perspective, in the US, there's about 11 billion $ft.^2$ of leased or owned corporate office space, and 42% of it is vacant, but paid for. The initial numbers that we had seen were about $132 billion on space that's essentially vacant, but people are paying for. It turns out it's closer to 300 billion. I mean, it's just a colossal waste of money, and the problem isn't that people don't know that they have a problem. They know they have a problem. The problem is that they don't know which 42%. So we tell them.

[0:05:09.3] JM: This problem might sound trivial to people listening, like the engineers in the audience, which are most of the people in the audience are thinking how hard can it be to just count the number of people that are in a room throughout the day. Why is this a sadly hard problem?

[0:05:29.2] AF: Well, because people are weird. It's probably the simplest way to put it. People in environments they inhabit are super strange. I mean, we have seen people – So we use depth data. We're not like an RGB camera. We just see in depth. So we can't tell, gender, age, ethnicity, or anything else. It's totally anonymous. But people bring stuff with them. Like we've seen people bring like giant trash cans in the middle of the night. They bring plates. It turns out plates look a lot like human heads. They bring strollers and they're on crutches. They take phone calls and they linger. Actually, we saw someone with a manikin once. That one we actually screwed up. We can't count someone carrying a manikin.

But the point is that in order to make sense of all of this strange human behavior, you actually have to design a system that can collect enough data without invading privacy and then run essentially on board local computer vision and machine learning to determine what is not human, just so that you can isolate the subject. I mean, it is an enormously complex engineering task. We've been working on it for 3 plus years.

[0:06:29.1] JM: So the problem of counting people, your solution involves machine learning. It involves computer vision and it's – The problem statement includes wanting to make this anonymize. So you don't want to just have a video camera and be doing naïve, computer vision, because there are certain circumstances where, in fact, widespread circumstances where you want to deploy a people counter that is not recording the entirety of video. Why is privacy so important for the implementation?

[0:07:06.8] AF: Well, you're never going to be able to convince an engineer it's okay for management to put a camera above an engineer's desk. It's probably the single biggest reason why our company exists. You can't put a camera in a conference room. Not one that is monitoring how that conference room gets used, and that sort of cultural concern for privacy is the reason that you can't put a camera in a financial institution, or a tech company, or a secure engineering lab, or a cafeteria. As a result, you essentially like are running blind. You're building a building blind and you're operating building services, for the most part, blind based on time schedules as opposed to demand.

A really good example of this is like we have a customer, a very large fortune 500 tech company who only 25% of their employees show up for lunch on Fridays and they actually buy food to accommodate 60% to 75% of their staff. The only reason they don't know that is because they don't have historical point of sales data because they give away food for free. So they can use us as proxy for demand.

[0:08:05.2] JM: You started Density 3-1/2, 4 years ago, somewhere around then, and the technology in this space of things that you ended up using in the implementation that you've ended up building, where you have this anonymized video data – We'll get into the specifics of what it is, but this technology has accelerated rapidly. So 3-1/2 years ago, what were the options for sensors and cameras in software packages to try to build this solution? What were some of the designs and the options that you considered that didn't work so well?

[0:08:45.7] AF: Well, I mean, we designed the system essentially to figure out how busy our favorite coffee shop was. So the start of the whole thing was we just wanted to be able have like a little bit of telepathy to be able to see how busy is a space before we show up. It turns out doing that is like incredibly hard, especially if you want to preserve privacy.

So where we started was actually, like the proof of concept was Mac address tracking. So we would use the probe requests from smartphones. This is before Mac address tracking was easy or like regularly talked about or sort of kind of in every router. So this is like 2013, 2014. But we started with Mac address tracking and we would listen for probe requests from smart devices and we would use that as proxy for count.

We realized very quickly that that does not provide an accurate count per room. So we moved on to a passive infrared and then active infrared distant sensing often used in motion detectors or in robotics, like the iRobot vacuum cleaner. So like sense distance.

We did some like signal processing to figure out like whether or not someone had actually walked through an entryway. Again, it turned out that we just didn't have enough data to make sense of the complexities of human behavior or the environments it was in. So we finally landed on depth. So we use for Class 1 infrared eye safe lasers. We illuminate the space essentially at a point of entry, and then as people walk beneath the device, we measure what is in effect. This is imprecise, but it's sort of a good way to understand it. We measure essentially the amount of time it takes for light to return back to the device and we collect – I don’ t know, roughly 10 million individual depth values per second. Then at the certain frame rate, we measure direction, speed, collisions lingering and so forth. To process all of that, our product is effectively a laptop above an entryway. Incredible processing capabilities for the size product and it has roughly 800 individual subcomponents.

That sort of like a little bit long-winded, but the point is it’s an incredibly complex product and it has two or three individual forms of calibration before it even leaves the line and arrives at a customer's location.

[0:10:50.0] JM: To get to the product that you got today, did you have to develop a system for doing rapid prototyping? Did you get to the point where you were saying, “We figured out what we want to build. We want to build this people counter system. We want to count the people in a spacer or count the human density in a space. Because we have the problem statement, we need to develop a system for prototyping as we go along, because we realized we're going to have a lot of failed experiments." Or did you not have to develop an actual systematic way of doing the prototyping? Were you just kind of making things up and pulling components out of one piece and putting them into another as you were going along? What was the system for rapid prototyping?

[0:11:34.5] AF: So we have a saying at Density, and that is to just put it above the door, meaning all the planning and sort of process aside. I think of things are very important, especially sort of a methodology that allows for rapid prototyping. Nothing beats a demo. Nothing beats like, "Does it work? What do we see? Show me what the device is. Put it above a door."

We actually had this circumstance where we had been spending a ton of cycles arguing over light, and would 850 nanometer work? We're going to be competing with ambient sunlight. Well, I don't know the arctans on the – I mean, it was just like sort of unnecessary detail and someone had the bright idea to just put the damn thing above a door. It was a prototype. It was obviously bigger than we currently ship with. Put it above a door and, of course, right out of the gate, person walks through an entryway, pings our servers and we pull down into a simple web application plus one. I swear, it was one of the most remarkable moments we had ever had and it was a really good lesson. Sometimes like putting it out into the real world is probably more efficient than sort of discussing the details.

[SPONSOR MESSAGE]

[0:12:49.5] JM: Stop wasting engineering time and cycles on fixing security holes way too late in the software development lifecycle. Start with a secure foundation before coding starts. ActiveState gives your engineers a way to bake security into your language's runtime. Ensure security and compliance at runtime.

A snapshot of information about your application is sent to the active state platform. Package names, aversions and licenses and the snapshot is sent each time the application is run or a new package is loaded so that you identify security vulnerabilities and out of date packages and restrictive licenses such as the GPL or the LPGL license and you identify those things before it becomes a problem.

You can get more information at activestate.com/sedaily. You want to make sure that your application is secure and compliant, and ActiveState goes a long way at helping prevent those kinds of troublesome issues from emerging too late in the software development process. So check it out at activestate.com/sedaily if you think you might be having issues with security or compliance.

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

[INTERVIEW CONTINUED]

[0:14:19.5] JM: So describe the technology solution that you ended up building for Density. You have this thing that fits above a door and is going to be able to count the people that are walking in at any given time. Can tell you how many people are in a room or what the occupancy percentage is. Explain what that technology looks like on the device.

[0:14:43.4] AF: So do you mean like how does it specifically work?

[0:14:46.5] JM: Yeah. I mean, what are the components? I guess top-down explanation. So first of all, what is it doing? You have people walking in. How is the device perceiving these people? Then what is it doing on the device and then how is it communicating that information back to the server where me, as the shop owner, can look at this information and see how many people are in a room at any given time.

[0:15:13.0] AF: Yeah. So just a point of clarification at the end, we don't sell to large or small retailers. We really only sell to large corporate offices or universities, people who have really large campuses, at least for now. It's sort of the appropriate first market for us.

As to like how the system works. Really, what leaves the device is telemetry data or the health of the device and account data. We have some other sort of metadata that goes up about what the algorithm is perceiving or seen, but we don't stream what the device see. We don't we don't stream those depth values. That all gets processed locally on the device for a variety of reasons, but the primary one just simply ensuring that we don't hog bandwidth.

So when a person walks beneath a DPU, which is a depth processing unit, a density DPU, light sort of bounces off of that person and whatever it is that they're carrying. We have labeled tens of thousands of individual static frames. So this is what an arm looks like. We call them coloring parties. That is what the top of a head looks like. This is what head and shoulder looks like. This is what a door looks like. This is what walls look like, and there are quite a few label types and categories.

But we've labeled tens of thousands of these static frames and the machine learning model that exists locally in that device essentially determines whether or not what it's seen as human or what it seen as a dog. Interestingly, if you want to fool or device, you can get on all fours and crawled through the door and Density will see you, but it will think you're a dog or something nonhuman.

Anyway, we process first off – So we actually have what's called F1 score. So we have to get two things right. One, we have to identify that a human entered at all into the scene and an event occurred. Then the second thing that we have to do is determine which direction that human went.

Now, that sounds relatively straightforward. The problem is that people will often enter the scene and then linger for many minutes, which means we have to call it measure or monitor the collection of depth values that is the top of that person's head and shoulders for as long as they are inferring, and can be really hard, especially because our devices get deployed all around the country in lots of different spaces where there are thousands and thousands of employees using space.

So give you a sense of scale. We need to be on every point of entry. So we'll be on – Let's call it – We a customer who has a cafeteria with five entrances. We have a DPU above every entrance and then we automatically reconcile count, accurate count for that space, that particular cafeteria. One of their doors over 120 day period had 91,000 entrances and exits. They have 5,000 employees. The next closest door had 131,000 entrances and exits. The most popular door was to the elevator. It was 515,000 entrances and exits, and there were two of them with the same number. The point is, when we showed the small tangent, when we showed this data to the customer, they thought we were lying. We actually measure accuracy on a door-by-door basis, on a DPU-by-DPU basis. So we can actually prove that we were 96% accurate. It was actually a variable. So we actually don't give an accuracy score generally for the product. Instead, we provide accuracy for each door down to two decimal points.

So it was like 96.02, 96.7, 95.6 and then so and so forth down the line. We showed them these numbers and then they said, “How did you do that?” It turns out that when you don't have ground truth, it's actually very hard to design any kind of system like ours. Because we can't compare what our algorithm – How our algorithm performed against a baseline. We have no control.

So what we do – Sorry. Is this may be on target?

[0:18:52.7] JM: No. No. This is fascinating. I think what you're alluding to is something that’s actually just a widespread issue in science that kind of doesn't get talked about as much, because it's always a question, “What is your null hypothesis?” If you don't have a good way of gathering a null hypothesis, then you're just guessing.

So this company that you are dealing with just had some intuition about how many people were coming through each door, and that intuition was not based on some dataset or some rational perception. It was just that it doesn't make any – That it doesn't make sense to us that 515,000 entries through this door are occurring. I mean, we could extrapolate this scientific calamity to medicine or plenty of other more controversial areas, but even just on the surface of detecting people coming through doorways, it's fascinating that you don't have a ground truth and yet this company felt confident that they could disagree with you.

[0:19:53.7] AF: Yeah. What was really cool was we showed them the numbers, they no longer felt blind and they immediately deployed us all across the US. The other cool thing is what you just described, this sort of like gut response to sort of data list or sort of instinctual sort of response on what is right or anecdotal response of what's right or what not is how all buildings have been built. It’s how all cities have been built.

It's essentially like most human infrastructure is essentially like an architect's best guess. What's really cool is like once you have the data, it can become a science. All of it can become a science. And so we believe that if New York, New Hound New York was used, then New York would design itself differently, more optimally, and that's a very, very exciting future.

[0:20:37.9] JM: That’s what’s exciting about the whole Google building a city from scratch in Toronto thing. I’m sure you’ve heard that, right?

[0:20:46.5] AF: Sure, of course. Yeah, Doctorow and Sidewalk Labs.

[0:20:49.1] JM: Right. I mean, I think that’s cool. When you think about stoplights, for example, why do stoplights work? You pull up to a stoplight in the middle of the night and it's a red light and there's nobody coming in the perpendicular path and you’re like, “Why am I sitting here waiting for this light to turn green?”

[0:21:07.8] AF: So interestingly, an engineer and a friend of mine recounted the story about stoplights. So he and his buddies were visiting the US and they had a rented car they came up to this stoplight, and I think they’re from New Zealand. I think his friends were from New Zealand. They came up to this stoplight. They were in a convertible and they were all sitting there, it’s the middle of the night. It's a red light. They’re in a country that is not their own. So they're like debating whether or not to run this red light and they’re really concerned that if they like run the red light they might get pulled over and then they’re go to a United States prison or jail or whatever and they're just like having this back and forth.

As my friend recounts it, his close friend is just like exceedingly bright, like clock speed, brain clock speed, it’s just really fast. Comes out of his reverie – Was not paying attention to tee discussion. Comes out of his reverie, looks at the situation, listens to his friends having this this asinine debate. Jumps out of the convertible, walks across the street and presses the walk button for the pedestrian and then comes back and jumps back in the convertible and, of course, the light turns green. It’s because the system knew that a human needed to pass and so it should change the light.

I think I'm going to try to segue this back to people count. But I think the point is all of our – Our systems can be more efficient. Our systems can be more efficient, especially if we have data. So from our perspective, we've really tried to build a system that allows – It operates in the background. It does not invade privacy. It's respectful of the people that it's sort of trying to assist. Then we build an API that is essentially used to integrate with all different sorts of systems.

So to give you a couple of examples, the culinary example is the one that I just gave you. They're actually using it to determine how much food to buy based on the demand for certain types of food. They spend millions of dollars in purchasing food, a lot of food waste.

There's another company that's actually measuring the number of times a bathroom gets used and then pushing a notification. Our system is just pushing a notification after 50 uses, because they know that after 50 uses or a certain number of uses, it needs to get cleaned. Now that's a really simple sort of pedantic example, but it's the kind of thing that can have a profound impact long-term over sort of an experience of the space.

We have another group who is a university. They built a system that essentially tells students, and essentially when the library, their cafeterias and the gym is busy in real time from their phones so they don't have to waste time wondering. There's another group is actually using Density to reduce the amount of tailgating that happens into secure facilities.

So if you have essentially a place where people badge in, if you can compare that to ground truth, the number of people who actually walked through the door, then you can reduce the number of times that people tailgate or essentially two walk in, but only one person badges. There're all these ancillary systems that get better, smarter and more helpful if you just had the data.

[0:23:47.8] JM: On the device, you're doing this depth processing. The people are captured on some kind of recording.

[0:23:57.4] AF: So we actually do know post-processing or anonymizing. It's actually anonymous by design. So we don't blur anything. Instead, we simply get depth values. So you can actually stand beneath the device and look directly up at it and we cannot distinguish any facial features. What we see is essentially height. So we have individual height measurements at a very sort of granular detail, and that gives us sort of a contiguous body of depth data that looks like a silhouette. So you kind of look like a ghost. Frankly, the visualizations are mostly for our annotation process and for quality assurance. The machine actually doesn't actually sort of see it as a video-esque or GIF looking image.

[0:24:47.8] JM: What are the hardware components that you need to do that? Are any of those made from scratch or are they all off the shelf?

[0:24:54.6] AF: So the whole system is any system with this number of components is not sort of built entirely from scratch. It's essentially a combination from a global supply chain. So we have about 137 unique components. Meaning sort of very specific suppliers and a supply chain that we manage specifically, and then we have about 800 subcomponents in total. So that's everything.

Processor, elimination, lens to capture the infrared light, we operate at 850 nanometer. So it's four Class 1 infrared lasers. The relevance of 850 nanometers that we actually compete with visible sunlight. So we are an indoor product. So it's not a problem. But if you were to theoretically design an outdoor product, you would move further up the light sort of spectrum to 940 nanometer or thereabouts, and that way you'd have to increase the amount of power you consume, but you wouldn't have to compete with sort of the visible light.

Let's see. There's a IR plastic, or IR – It's called Ira Glass. Although it's made of plastic that allows higher light through, but does not allow visible light through. We have our depth. Essentially, what does depth calculations, it's a two chip solution. We support POE. So there's an RJ-45 jack and then we also have USB.

We actually decided to put the antenna for Bluetooth and Wi-Fi. Even though we support both hardline power over ethernet and like straight to a wall outlet power with a converter for the cat six cable, plus Wi-Fi for connectivity. We actually put the Wi-Fi and Bluetooth external to the device in a dongle so that we didn't have as large of essentially an attack surface.

When you're going into corporate networks of any kind, like having a wireless device, even just the antenna can be just like a huge info set red flag. So this one is actually physically removable, which is actually really nice, because it reduces the complexity of FCC and some of the other processes.

The device is actually actively cooled. So there is a small fan embedded in the lid. It's kind of cool. So it pulls in cool air from the bottom. There's a dual venting solution. It pulls cool air in from the bottom and then it exhausts warm air out the top. We have amount, if you're sort of familiar with the device. You can look at it on our site. There's amount that sort of mount flushed to a wall. The way we do circulation, it's really well-designed by – We have this incredible mechanical engineer and an industrial designer. They essentially use the mount as a thermal baffle so that the air doesn't mix, which is really cool. Then the entire thing is anodized aluminum.

So we don't operate at 26 Watts, but we have to support a peak of 26 Watts. Now, it's very small period of time where it spikes like that. typical operation is, I guess, call it 14 or below, but the entire device is designed to be – It's essentially a heat dissipation mechanism. So we have obviously a heat sink inside as well as an active cooling, but then just the surface area alone acts as a way to dissipate heat.

Then, honestly, that's kind of all the major components. Once it's up and running, it's got a pretty lightweight data footprint, because it's mostly text. We're not sending video. We're not sending sort of this really heavy, heavy data stream, and compression these days is remarkable. So, anyway, that's kind of how the system works. Then we only sort of process. We essentially process that data or attribute that data to the different accounts in the cloud and then that gets pulled down by our API, or by – We're obviously a self-consumer APIs. So you can also use some of Density's existing dashboards or web clients are mobile systems.

[SPONSOR MESSAGE]

[SPONSOR MESSAGE]

[0:28:43.5] JM: At Software Engineering Daily, we have a web, we have an iOS app, an android app and a backend that serves all of these frontends. Our code has a lot of surface area and we need visibility into problems that occur across all of these different surfaces. When a user's mobile app crashes while playing a podcast or reading an article, Airbrake alerts us in real time and gives us the diagnostics that let us identify and fix the problem in minutes instead of hours.

Check out airbrake.io/sedaily to start monitoring your apps free for 30 days. Set up takes only a few minutes. There's no complicated configuration needed, and Airbrake integrates with all of your communication tools from Slack, to GitHub, to JIRA, and it enhances your current workflow rather than disrupting it.

You can try out Airbrake today airbrake.io/sedaily. If you want to monitor and get visibility into the problems that may be occurring across your application, check out Airbrake at airbrake.io/sedaily. Thank you to Airbrake.

[INTERVIEW CONTINUED

[0:30:04.3] JM: What I think is cool about hardware in 2018 is that you have this buffet of components from the global supply chain, and because there was this massive ramp up in small component production due to the increase in smartphone production, you've got these small components where the economies of scale have already been achieved. Therefore, even if you're just a small company, you can purchase these one-off little pieces and wire them together into building something that otherwise probably would not be economical to prototype, like a depth perceiving anonymizing device that fits over a door. Then once you prove out the prototypes – So the prototyping processes is made cost-effective by the economies of scale that have already been achieved from the smartphone stuff. Once you actually have the device prototyped and figured out, then you can leverage this global manufacturing supply chain, which competition is driven down the cost of and you can get your device built even if you don't have a massive manufacturing demand relative to the smartphones. You can still kind of draft off of that same supply chain world and get your device built at an economical cost. Would you say that's an accurate depiction of supply chains in 2018?

[0:31:34.8] AF: I think at the prototyping stage, definitely, and you're going into production. So a typical cycle will follow essentially an 18 month process. So 12 to 18 months is sort of the fastest you can ever go from prototype to production. I'm not saying this just because it applies to our particular product, that the person who leads, our senior director of operations, came from Apple. He was responsible for quite a large portion of the supply chain, global supply chain for some of the Apple watch. His experience, I think, has taught us a lot about how to think about building product.

But one of the things is really critical is that in designing – In sort of prototyping a product and making it work, you are still a year away from having a product that could be production ready. So you'll go through like prototype and various form factors. You'll sort of arrive at a form factor and then you move into what is essentially called engineering validation testing, or EVT. There's EVT, DVT, PVT and ramp. So EVT is engineering validation testing. The D in DVT is design validation testing. The P is production. Then last is ramp. Ramp is essentially like you're in continuous build.

So when you think about building a product, the line that you design is as much a product as the product it's rendering. So it totally depends on what your objectives are. But if you're trying to build a product that can scale, call it hundreds, thousands, tens of thousands a week or more, then going through that lifecycle is like really critical. Frankly, there just aren't any shortcuts. It's one of those things that I think I, at the beginning, really felt like we could shortcut, because you can do that when you're a startup. But the reality is you can't.

Especially with a product that is as complex as ours with as many critical systems that have to work properly. I mean, we have wheat we have full stack engineering. By full stack, I thought I knew what full stack meant when we were a software agency. Full stack is like from bare metal, laying the traces from bare metal embedded systems, mechanical engineering and sort of everything that goes into making sure that that system can even turn on without browning out to the OS. All the algorithm work that has to work properly with that OS and not spike your power consumption or your power budget. All of the systems that sort of determine whether or not the device is on properly, working properly, is online, has the right kind of bandwidth necessary. All of the web systems that sort of allow us to remotely monitor in real time all devices and remotely push OTA's or any kind of over the air update and all the way down to sort of your standard web stack for turning this into a usable functional system, all for plus one, minus one.

I mean, it is just a – It was a colossal lift. So I totally believe that prototyping is way more approachable today than it's ever been. We are definitely the beneficiaries of that. But the reality of building a production ready product with any kind of complexity where you don't outsource it to a manufacturer and say like, "Hey! This is my spec. Build it," like an ODM or whatever, and you're like building it in-house. I think if we had known how hard it was before we jumped in, I don't know that we would've done it to be honest.

[0:34:45.5] JM: I imagine it's both humbling and empowering. So it's probably humbling in the sense that there were times – I'm sure there were times when you were building this thing when you were like really questioning your ability to get it done, or maybe you just have ironclad confidence and I'm mistaken. But in any case, you got there or you're very close to getting to where you could probably say, "Look. We got this thing. It's full stack. We've got the software figured out. We've got the hardware figured out. We've got the interconnection between the two figured out," and you didn't have to like go back to school to figure out how to do full stack engineering.

What are the big takeaways from managing a full stack engineering problem? If there are people in the audience who they aspire to someday working on an integrated hardware and software solution, what are the biggest learnings? By the way, I think you're not like formally trained as an engineer, right?

[0:35:44.3] AF: No. No, I'm not. I'm not. No. I was a writing major. But the technology was kind of what I did because it was awesome and fun. I didn't really actually know I could get a job in it. So I would say almost all of the founding team were self-taught engineers of some kind, designers and engineers of some kind.

[0:36:02.1] JM: I got to my question. So I was just kind of rambling around it. But, look, what are the big learnings for building a full stack engineering solution?

[0:36:10.6] AF: So low grade paranoia is a great thing. Non-debilitating self-doubt is a great thing. I think the thing that were most concerned with is the stuff that we don't know, like the stuff that we don't know that we should. So makes it really hard to identify what we need to know in order to be able to solve some of these problems. That's where experience like can fundamentally change your ability to build.

I would say that the thing that has been the most helpful as sort of a rallying point for our organization is like, “Please identify what the fundamental problem is that you’re solving.” What is the thing that is most important? I think a lot of people call this first principles. From our perspective, we would argue about whether or not the thing would work instead of just putting it above a door to see if it worked. Sort of an example of like “Is there a faster way, a more efficient way to solving the fundamental problem?”

But you have to identify it first. You have to be honest enough about the things that you’re not good at. We brought in some people who’ve had some very deep experience in embedded systems and security. Some very deep experience in PCP layout and design. Some people with remarkable industrial design and mechanical engineering backgrounds, but at the same time we paired them with what I would say were almost less to no experience, little to no experience in hardware engineering but quite a lot of experience in software design, and that sort of commingling was really good because the hardware crew didn't know what the software crew knew and made assumptions about what was easy and what was hard. Sometimes that actually led to more efficient solutions and vice versa.

The software team didn't know what was easy and what was hard with hardware, and that naïveté sort of being naïve essentially gave us the ability to try stuff that I think other hardware companies never would if it were filled with a bunch of veterans. So I would say really look. If you're looking for teams – If you’re looking to build teams, that’s sort of another discussion. If you're looking to join a team or you’re looking for teams that fall into that category, look for groups who are going to be successful in my opinion, and I've only got one point of reference. So I’m kind of bias. But look for groups who are diverse, who are not coming from a homogenous place. I mean that in the literal way, like gender, age, ethnicity. Look for diverse teams. They build better products. They see blind spots faster.

But I also mean it in sort of the broader sense that like sometimes not knowing the right answer or having experience in the right answer is the most efficient way to find it, because they're going to ask questions that other people might like gloss over. I don't know if that answers your question.

[0:38:39.5] JM: It does. We met when I was working on this other business. I was trying to do this other business alongside Software Engineering Daily, and the biggest problem that I had around that time was that I was trying to do two things at once. I have found that focusing is a huge issue for me and I think is an issue for a lot of engineers or a lot of creative people who get excited about a lot of different things at once and then it just end up being pulled in different directions.

It looks like you have either never have that as a problem or you've managed to overcome that, because you're very specific about what Density is doing. You are building a hardware device to count people, and it seems like if you can find the focus of the business and you can find the mechanisms of growth along that thin continuum of things that you're working on, you're goal as a business owner. But has focus been an issue at all for you? How have you found that place where you're just focused on something narrowly?

[0:39:46.4] AF: First off, I really appreciate the credit your trying to gift to us, but we struggle with focus all the time. It's sort of this constant battle of when to explore and be curious and when to stay focused and diligent. I think it's, again, sort of somewhere in that balance that you find – That the great product design happens. You can think really creatively, but sometimes it can be just sort of – Without the right focus, you can end up wasting time.

I think, generally, we like to go where the enthusiasm is. Meaning, like enthusiasm is this really lovely renewable resource. If you can figure out what it is that someone is truly excited about, they will put up their own blinders to focus on that thing. When people are not excited about something, it doesn't mean that it's like not worth doing, but it does mean like it's probably worth asking, like, "Is it worth doing?"

Maybe I have sort of a utopian perspective or opinion on like what product design should be and that like all things should be fun, and I don't mean to describe it that way. But we have 100%

retention. We were founded four years ago. We have a 100% retention. We were founded four years ago. We have a 100% retention at our voluntary retention. We've had some churn where we felt it wasn't working for the company and we've had to let some people go.

But no one has voluntarily left us yet, and I suspect that will change and certainly as we grow. But I believe I attribute that to staying ahead of people's needs and actively trying to invest in the things that they're enthusiastic about, and it's paid off. I mean, people love working with one another and, frankly, when you have an organization that as a hardware startup needs to be, you can build a lot of things and there's so much area – Sort of a company can build anything is a very exciting incidental outcome.

[0:41:25.7] JM: Definitely. To wrap up, I want to talk a little bit about the go to market. So when you're trying to find people to sell the hardware device to and you're trying to figure out the pricing model for counting people. I mean, I imagine counting people. That's something that the business value will vary from customer to customer. What's the go to market strategy? How do you figure out sales? How do you figure out finding the right leads and how is that process evolved over time?

[0:41:58.0] AF: Well, I should say a couple things. One, it's always evolving, especially early. I mean, we just started shipping our production volume eight weeks ago. So it's still very early. That said, we've deployed a lot of product before kind of our volume became available. But we were very fortunate. We had a lot of inbound interest from this particular market. We've had a lot of inbound interest from pretty much every market that deals in human space. So advertising, real estate, retail, insurance, travel, government transportation, sort of federal spaces and so forth. It was just all over the map.

But the one that really made a lot of sense to us from a distribution standpoint was large corporate offices. They have control over a very large set of square feet. In some cases, 15 million, 50 to 70 million ft.^{2}. When we're talking about one unit for every 1,000 to 1,500 ft.^{2}, it's a lot of potential devices. We also wanted to build a model that made people count really accessible. We think the people count is most interesting in the context of other data and less interesting by itself. So our API was actually our first product. So it was the device an then the

API. It wasn't a dashboard. It wasn't something flashy. We wanted people count to exist in the systems you already relied on.

Within sort of that space, we also wanted to make it dead simple to pay for the thing that you cared about. People don't buy the device. They buy the value that the devices data creates or can create. So what we decided to do is give the hardware away for free and charge for access to the data. So charge for essentially a recurring cost for generating and providing that data on an annual basis.

Without getting into sort of specific prices, I'd rather sort of talk about like why we think that that's relevant. It's sort of a new product category. There's not something that we're replacing. So price was actually very difficult for us to come up with. There were no comparables. That was okay. So we essentially tried to assign a price that we felt good about in terms of cost recovery and the ability to sort of truly service that customer the way that they needed to be on a per device level, and then some fun things came out of.

We give customers a lifetime guarantee for the hardware and we essentially retain ownership or responsibility, I mean, of the device itself. So if it ever breaks, if the technology gets better for that particular door that could better service that door or whatever else, you literally slide it of its bracket, you send it back to us and we give you a new one. That sort of continuity of data access is more important.

Frankly, I think, it's where a lot of the systems are going. The hardware should be invisible if you're not interested, but it should be remarkable if you're up close and really curious about it. So we try to design a model that allowed us to have ultimate flexibility and to really, really service the actual customer need.

[0:44:43.7] JM: You are a friend of the show. You've helped me out in the past, and so I can vouch for you being a solid guy. I know you're hiring, and since you're a friend of the show, I just want to give you a chance to talk little bit about the company from that perspective. What are you hiring for and what is it like to work at Density?

[0:45:03.9] AF: So we're hiring in software engineering, and I would say like software engineering is probably the place where we're really excited about. Two particular positions, one is a backend engineer. It can be remote. Oh! I also forgot to say. The other is a dev ops engineer, which I'll talk about in a second.

The other thing I wanted to say is we have 40 people in 14 cities. We have our office or our headquarters in San Francisco, but we have another office in Syracuse, New York. So it's third in San Francisco, a third in upstate New York where we were founded, and then a third distributed all around the country. For the most part, we don't discriminate on location and there are no center sort of like clusters of a particular team in one place or another. So it allows us incredible amounts of flexibility with who and where we hire. But the two positions are backend engineer and dev ops.

Just to give sort of a sense of what they would be dealing with and then I can talk about the characteristics we're looking for. Our micro service architecture has seven APIs; health, core, accounts, algorithm, telemetry, logistics and integrations. We process roughly 30,000 telemetry requests per minute from sort of DPUs in the field, which is about 500 requests a second. Then our core API processes thousands of requests per minute during open hours. Then we have tens of millions of events created.

The point is, it's a colossal amount of data and it's coming from everywhere. We have deployments all across the US and a couple of places internationally, although it's more sort of for prototyping in preparation for full international support. But it's a really cool system and we've done our best to sort of architect it for scale, but finding people who understand how to design for scale, support for scale and do so responsibly from an infosec perspective is very hard, that that's a very particular skillset.

The last thing I'll say is that the thing that matters most to us is – I would say at the top of our list of values is humility. We think that humility is this really functional value. Someone who is humble is more likely to be convinced of an opposing opinion, and we really love being able to sort of enter those debates where sort of even from the quietest voice in the room, the idea may prevail.

So I think I'm happy to talk a bit more about what we're looking for, but we don't care where you went to school. We don't care if you went to college. We don't care – We have some people on our team who are some of our best engineers. We have our 18 and 56. We have people from all backgrounds and walks of life. To us, all that we're really interested in is their capacity to be able to teach us something and learn really rapidly and to be able to build interesting systems while being humble.

[0:47:46.7] JM: Cool, and how can the listeners find out about jobs at Density?

[0:47:50.9] AF: Just jobs.density.io.

[0:47:53.7] JM: Andrew, thanks for coming on the show, and I just want to thank you one more time. When I was building this other company, Adforprize, I was working on that about a year ago. When I was really in the thick of it trying to build Adforprize, at the same time I was working on Software Engineering Daily. You were really generous in making some time to help discuss some things that I was having issues with within the scope of that company. This was in the midst of you working on Density, scaling Density, building the product, getting your supply chain set up and whatnot.

So I just really value that you took the time to spend some time with me back then and hopefully am I paying it forward somewhat by having you back on the show. It was of course awesome to have a conversation. I'm looking forward to chatting more in the future.

[0:48:45.3] AF: I have no double you'll come up with a whole bunch of other ones and I hope you will give me a ring when you do. I really appreciate you having me on the show, and thanks again for the time.

[END OF INTERVIEW]

[0:48:56.1] JM: You listen to this podcast to raise your skills. You're getting exposure to new technologies and becoming a better engineer because of it. Your job should reward you for being a constant learner, and Hired helps you find your dream job. Hired makes finding a new job easy. On Hired, companies request interviews from software engineers with upfront offers of salary and equity so that you don't waste your time with a company that is not going to value your time. Hired makes finding a job efficient and they work with more than 6,000 companies, from startups to large public companies.

Go to hired.com/sedaily and get $600 free if you find a job through Hired. Normally, you get $300 for finding a job through Hired, but if you use our link, hired.com/sedaily, you get $600 plus you’re supporting SE Daily. To get that $600 signing bonus upon finding a job, go to hired.com/sedaily.

Hired saves you time and it helps you find the job of your dreams. It's completely free, and also if you're not looking for a job but you know someone who is, you can refer them to Hired and get a $1,337 bonus. You can go to hired.com/sedaily and click “Refer a Friend”.

Thanks to hired for sponsoring Software Engineering Daily.

[END]