EPISODE 459

[INTRODUCTION]

[0:00:00.3] JM: Imagine that you are a lawyer. Your work involved managing files with dense technical texts. Your coworkers collaborate with you to accomplish a complex goal that can be broken down into smaller pieces. Your work has formal specifications, but there are degrees of freedom in how you express an idea. In all of these ways, the job of a lawyer is similar to the job of a software engineer, so why don't lawyers use tools to improve their workflow like software engineers do?

As a software engineer you have project management tools like Asana that improve collaboration. You have APIs like Stripe that reduce the time spent on a complicated implementation. If tools like Linters and Source Control that prevent you from making fatal errors, all of these tools save you time.

At many law firms, lawyers do not have that incentive to save time. They're paid based on billable hours and not individual milestones. Historically, this hourly billing made sense. Lawyers have been around since long before computers, and the amount of work that might go into a legal task was hard to predict before you had computers to log data and sort documents and standardize communications.

In contrast, a software engineer has always had the ability to automate work. That's why in most cases we're not rewarded based on our time spent solving a task or paid based on hitting our KPIs and our milestones. The legacy of hourly billing, lawyers can look at repetitive and administrative tasks as opportunities to make more money.

Justin Kan on has been building startups for a decade, and in that time he's interacted with lots of lawyers, from incorporation, to fundraising, to selling his company, Twitch, the interactions with lawyers consistently seemed less transparent and less efficient than would be optimal. For an engineer like Justin, the natural inclination here was to build software and sell it to lawyers, but there would be so much resistance. You would have to convince the lawyers to change their pricing model to fixed pricing, which would give them the incentive to buy software and to work more efficiently. instead, Justin teamed up with a few entrepreneurial lawyers who were willing to start a new law firm from scratch, a new software on day one.

The software company is called Atrium Legal Technology Services, or Atrium LTS for short, and the law firm that uses the software is Atrium LLP. Both of these companies are very new and they were publicly announced a few months ago.

The two companies work side by side in undecorated office space in downtown San Francisco. When I took the elevator up to see the company, the elevator doors opened and revealed two paper signs pointing to opposite ends of the office. On the Atrium LTS side of the office, engineers were writing software to extract the meaning from documents.

Today, lawyers at old law firms are paid hundreds of dollars an hour to fill in document templates by editing a text document. As the Atrium LTS software gets better, document preparation will be done through web applications with the variable names disambiguated from the parts of the document that never change from client to client.

On the other side of the office sat Atrium LLP. The legal team was dressed a little more formally than their engineering counterparts on the other side of the office, but there was nothing close to the formality of a traditional Silicon Valley law firm who's far from the décor of the Menlo Park law firms, and the office space was actually more Spartan than most well-funded startups signaling to the employees that this is an unproven business strategy and there was a ton of work to be done to validate it.

This sentiment was echoed in my conversation with Justin. It's possible and even plausible that Atrium LLP could become the biggest law firm in the world powered by the software of Atrium LTS, but the road to getting there will take patience and stead execution. I enjoyed hearing Justin explain the motivation for starting Atrium LTS and Atrium LLP and I look forward to covering the company more in the future.

We've done several shows about the intersection of software engineering and law including our show; Dissecting Software Anti-Trust with law professor, Harry First. To find all of our old episodes, you can download the free Software Engineering Daily app for iOS and for android.

In the other podcast players, you can only access the most recent 100 episodes, but with the Software Engineering Daily app, you can find all of our back catalog and get recommendations based on your listening history. With these apps, we're building a new way to consume content about software engineering. You can also find the desktop application at softwaredaily.com, and these apps are open sourced at github.com/softwareengineeringdaily. If you're looking an open source project to get involved with, we would love to get your help.

Shout out to today's featured open source contributor, Craig Holiday. Craig has worked on the Software Engineering Daily iOS app to iron out performance issues and implement features like two times playback, so you can listen to this episode in half the time. Big thanks to Craig and his brother Keith for all of their work and their contributions to the open source community.

With that, let's get on with this episode.

[SPONSOR MESSAGE]

[0:05:49.4] JM: The octopus, a sea creature known for its intelligence and flexibility. Octopus Deploy, a friendly deployment automation tool for deploying applications like .NET apps, Java apps and more. Ask any developer and they'll tell you that it's never fun pushing code at 5 p.m. on a Friday and then crossing your fingers hoping for the best. We've all been there. We've all done that, and that's where Octopus Deploy comes into the picture.

Octopus Deploy is a friendly deployment automation tool taking over where your build or CI server ends. Use Octopus to promote releases on-prem or to the cloud. Octopus integrates with your existing build pipeline, TFS and VSTS, Bamboo, Team City and Jenkins. It integrates with AWS, Azure and on-prem environments. You can reliably and repeatedly deploy your .NET and Java apps and more. If you can package it, Octopus can deploy it.

It's quick and easy to install and you can just go to octopus.com to trial Octopus free for 45 days. That's octopus.com, O-C-T-O-P-U-S.com.

[INTERVIEW]

[0:07:19.7] JM: Justin Kan is the CEO of Atrium LTS. Justin, welcome to Software Engineering Daily.

[0:07:24.5] JK: Thanks for having me.

[0:07:26.1] JM: In the 1990s, the biggest financial cost to starting a company was often servers. Today, the servers are pretty cheap, but something that remains expensive is legal. What are the common ways that a startup has to interact with a law firm?

[0:07:42.1] JK: Well, basically every company has to interact with lawyers for every major financial transaction. It's kind of the operating system of business in America, and so you always have to interact with it whenever you do anything really big. If you raise money, you generally pay lawyers. If you sell your company, pay lawyers. If you do a major commercial transaction and you want someone to review it, you can pay lawyers. Pretty common, and I would say it's an inescapable thing, kind of like death and taxes.

[0:08:17.4] JM: The legal industry itself has not been transformed by technology in the same way that some other industries have. Why is that?

[0:08:26.6] JK: Like you alluded to earlier, legal cost have actually only risen in the past 20 years pretty much across the board for big corporate law, but also you could say localized startups as well. One of the questions might be why? Why is that the case? Why they're doing the series A versus 20 years ago? It's mostly the same thing. Generally, prices fall. Why have prices not fallen in legal? I think there's a bunch of different reasons for it. One of the main reasons is that there's very little process or technology improvement. The legal industry has been an innovator's paradox, legal innovation paradox, which is that there's no incentive. When you bill on an hourly model, there's no incentive to improve efficiency overtime. It's not that lawyers are trying to figure out how to charge the most or bill clients to handle the most number of hours. It's just that the incumbents don't have any incentive to figure out how to reduce cost through the adoption of better software practices or better software or even better process. What happens is they don't actually adopt any software, and so there's a very little market for it.

[0:09:44.4] JM: What is the technology stack inside of a law firm?

[0:09:48.4] JK: Outlook. It's like Outlook. Lawyers do all their work in Outlook and Word, which are pretty robust tools as you know, but pretty static. There haven't been very many improvements in the past 20 years and so I think lawyers live in documents and email and that's kind of the tech stack they use, and then there's often times a document management system as well.

[0:10:15.8] JM: You've started a bunch of companies and you've engaged with lawyers throughout those different companies. Is there a particular event that stood out where you said to yourself, "This makes absolutely no sense," or was it more of an accumulation of just minor inefficiencies and inconveniences?

[0:10:36.2] JK: I would say it was just interacting with the industry overtime. I've been — Over the past 12 years in Silicon Valley, I become what I call an involuntary power user of corporate legal services. I used them for all those events that we talked about, whether it was selling company or raising funding. I wouldn't say there was — I actually wouldn't characterize my experiences as bad at all. I would say they were just okay. Some were better. Some were worse. I like using legal services when I felt like the lawyer was an expert who's giving me their expert opinion and advising me through something that was very complicated or that I needed confidence that was going to go well.

I think that at various times I felt like it wasn't going as well or I didn't like the service as much when things were opaque, I didn't know what the process was or I would be billed what seem like a random amount, a random high amount. That's kind of like what led me to the idea for Atrium LTS, was kind of asking, "Isn't there a better way to do this?" It wasn't really like a — It wasn't one event. It was more of like, "I'm sure this industry could be better."

[0:11:54.5] JM: We'll get into the two businesses, one of them is legal technology services, which is the set of services, software that you're building, and Atrium LTS which is, as I understand, the first law firm that is dog-fooding these services. Do I have the right framework?

[0:12:14.1] JK: Atrium LTS is our legal technology services company that I'm the CEO. It's a startup. We run it like a startup. We have engineers and they provide services to law firm owned by Augie Rackow and Bebe Chueh, who are two lawyers. It's called Atrium LLP and they serve clients in Silicon Valley startup clients. That's kind of the relationship between these two entities.

It stems from the fact that there's a bunch of regulation around who can own a law firm and who can provide legal services. I'm not a lawyer. I've just been, like I said, a power user. Our goal was to, within the bounds of — Was to like have shown the feedback cycle and actually remove the legal innovation paradox by having a firm that was built around the idea that you could improve the deliver of legal services. Atrium LLP is that firm.

[0:13:13.8] JM: Got it. Describe some of the services that you're building at Atrium LTS.

[0:13:21.1] JK: I would say it's a bunch of things. The interesting product vision that we're building towards is what we called no documents. I think that what I mean by that is that if you look at your company's data, it's all — The corporate data of your company, like who owns the company, the owners who's on the board, what the board has done, who are the employees, what the employees are paid, all of these data. What the contracts are of the company and who's owed what. All of these data is PDFs and Word documents that are in maybe a box folder or a Dropbox folder. Most people — It's like not accessible to people.

Even to the company owners, they don't necessarily understand how to parse it. They can't really extract the structured data very easily from those documents, but those legal documents are the source of truth. They describe exactly what your company is more than anything else.

What we're trying to build, ultimately, the way I think about it, is something like a system that basically sucks in all of those documents and turns them into structured data. We understand your company. If I put my engineer hat on, I think that's like a very interesting problem, because there's a bunch of scripting and machine learning tools that we use to build to extract that data, and then I think of that data as a platform on top of which you can actually build a bunch of interesting applications. If you need to render a cap table for financing in Excel, that should be one button, because you already understand the existing capitalization structure of the company. You can build a document management system or contract management system on top of it. You can build a dashboard for the company owners that shows exactly — Allows them to do a lot of things on top of their company, whether it's initiate a board action or send an offer letter. All of these things are basically actions on top of the existing data of your company and ways to visualize or render that data. I think that's pretty powerful and like in a platform that hasn't existed for companies yet.

[0:15:27.3] JM: Can you take anything off the shelf, like a DocuSign API or a e-shares, these other —

[0:15:35.0] JK: Absolutely. I think that e-shares is a great example of like we could use this data extraction to basically automatically onboard you to e-shares. I love e-shares as investor and as a user. I like e-shares a lot. The onboarding process is manual, because somebody has to put in all the data of like the capitalization structure of a company, right? For DocuSign, for example, of a HelloSign, we could build on top of those and actually allow — We would use the APIs to send out these board consents or something like that, or an offer letter or whatever. These could all be built on top of the existing e-signing tools.

I think that, ultimately, I see it as like we're building this platform where we understand the structure data of your company and then we're building applications on top of that. Those applications probably have. Interact with other APIs and applications out there, another software that's out there.

[SPONSOR MESSAGE]

[0:16:36.7] JM: Auth0 makes authentication easy. As a developer, you love building things that are fun, and authentication is not fun. Authentication is a pain. It can take hours to implement and even once you have authentication, you have to keep all of your authentication code up-to-date.

Auth0 is the easiest and fastest way to implement real-world authentication and authorization architectures in to your apps and APIs. Allow your users to login however you want, regular username and password, Facebook, Twitter, enterprise identity providers, like AD and Office

365, or you can just let them login without passwords using an email login like Slack, or phone login, like WhatsApp.

Getting started is easy, you just grab the Auth0 SDK for any platform that you need and you add a few lines of code to your project. Whether you're building a mobile app, a website, or an API, they all need authentication.

Sign up for Auth0, that's the number 0, and get a free plan or try the enterprise plan for 21 days at auth0.io/sedaily. That's A-U-T-H.io/sedaily. There's no credit card required, and Auth0 is trusted by developers at Atlassian and Mozilla and the Wall Street Journal and many other companies who use authentication.

Simplify your authentication today and try it out at A-U-T-H.io/sedaily. Stop struggling with authentication and get back to building your core features with Auth0.

[INTERVIEW CONTINUED]

[0:18:26.7] JM: Could you maybe talk through what's an example document that you looked at and you said, "Okay. Here's how we're going to architect an algorithmic way of looking at this and extracting data from it."

[0:18:39.5] JK: One very simple, like a trivial example. Kind of one that we've built pretty early on, is Safe Notes and Convertible Notes. These are instruments where someone is investing in a company in a non-equity realm. They're very common in Silicon Valley. Y Combinator popularized this idea of a safe simple agreement for future equity.

The idea behind is basically you're agreeing to buy equity in the future so you can basically invest in a company without actually having to go through the process of doing an equity financing which is expensive and takes a long time.

These notes are all based off of very small set of templates, and so it's actually very easy to build something that extracts, like takes these PDFs, breaks them open, extracts all the texts from them, compares them against known templates, extracts the real metadata from it, things like how much this person invested, what's their entity name, what's their address, email address, and then obviously store those into a structured database. Then take all that data, and like when you're going to do an actual series A financing, one of the processes is understanding who's going to own how many shares at the end of the financing and building an Excel model that you can share among all the constituents, the lawyers, the company owner and the founders and management. You can share this model so they can see exactly who's going to own what.

That whole process of extracting, like taking these saves, reading them, understanding them, and then turning them into an Excel model is done by an attorney right now. That attorney bills $500 an hour to basically make this Excel model. Now, obviously that's something that could be done programmatically. That's one example of something that we're doing. Basically, extracts those saves in the method I described and then takes that structured data and then renders an Excel model based on it.

[0:20:32.4] JM: Sure. The safe note, that is something that's very templatized, because every startup in Silicon Valley that uses a safe note is going to use one of these templates that looks very similar. Even operating agreements I think are — Some people have pretty off-the-shelf operating agreements that are, but nonetheless do have data that you would want to extract in somewhat a standardized dashboard format that would be useful.

Across the board in terms of how people engage with legal documents, it's a wide of array and that's certainly outside of Silicon Valley. I assume it's less standardized. Is it or is it still standardized when you step outside of Silicon Valley? Is it still templatized where you can have these well-formatted documents and have expectations around where certain variables are going to be?

[0:21:28.4] JK: Yeah, I would say that. Obviously, the safe is very kind of very constrained example. There's other types of documents that we're building machine learning classifiers to actually categorize and requires a little bit more advanced pattern matching than just like comparing its known templates, right?

I do think that depending on the practice of law, there is more or less standardization. Other practice areas, like lease agreements for commercial real estate. It's very similar thing over and over and over again; fund formations, same thing.

There are many other practice areas where I would say the same types of techniques can be used to understand a structured data. Those are the ones where I think we would have the most ability and advantage in actually trying to into and building applications for them.

[0:22:16.2] JM: Yeah. I think as I understood preparing for this, LTS is not just about machine learning for law. It's also about making the lawyers more productive in their job, because I think for the foreseeable future, there's going to be a lot of subjectivity in dealing with legal documents. You can't just unleash the algorithms and have them take care of everything. What are some of the ways that you can make the job of the lawyer easier and automate a way some of the painful processes?

[0:22:53.9] JK: People are always going to want a lawyer. When we were selling Twitch, we didn't want the Uber for lawyers. We don't want a marketplace of lawyers. We didn't want an AI algorithm lawyer. We wanted lawyers who had experience selling billion dollar transactions. That is going to be common and immutable, in my opinion, for the most valuable work. You might incorporate your company on deals that are more clerky or something like that, but ultimately when it comes to the big transactions, which are the valuable lucrative transactions, people are going to want a real attorney who has experience. Someone with a great background, who went to a top law school and went to a top firm and has worked on this type of transaction, and I don't think you can get away from that.

The goal with Atrium LLP is that Atrium LLP would be a top firm like that that has that type of talent, and the software is really reducing work and also providing transparency and speed behind the scenes or maybe even parallel as well.

I think that's the core philosophy. It's like creating an AI that replaces lawyers. It's maybe creating some AIs in machine learning and scripting and software that does some of the base level work, the like trivial work, like document classifying, like understanding all the default structured data. All of that stuff is things that you don't need someone who went to Harvard Law School to do. So I think that's kind of the way I think about it.

In terms of what we get the actual works, like saving. I think it's in many different things and it's like kind of small savings and all these different areas, whether it's rendering the first version of documents or whether it's giving more transparency to the customer so that they send less emails and texts, like transparency into what is exactly being worked on. Almost like giving an interface on to the legal work that's similar to like a legal Asana or Jira. I think that's something that even in the industry today could adopt, but they don't really, because there's no incentive to it.

[0:25:02.1] JM: At Atrium LLP, are they adopting stuff like Asana? Do they use project management tools?

[0:25:08.7] JK: Yeah. Using project management tools, yup. It's innovative for the industry, but I think it's a good step forward.

[0:25:16.7] JM: They do use the — Atrium LLP, they use these project management tools.

[0:25:20.9] JK: Yup.

[0:25:21.2] JM: Awesome. Had they said anything like, "Oh, this revolutionizes how we do our legal work," or anything like that? What's the feedback you're hearing?

[0:25:32.0] JK: The feedback is good. The feedback is good. I think we've managed — Atrium LLP is comprised of many entrepreneurial lawyers, especially younger lawyers who are excited about innovating and adopting new technology. That is kind of reflected in how they approach it. You can compare it — Most firms, the people who are in charge of the firm are the people who are most senior and often times very engrained in doing things a certain way, and so often times that's one of the reasons why there's this lack of adoption of new software, even for commercially available software that might be adopted in many other areas outside of the law, like in Asana.

[0:26:17.6] JM: How is the engineering org at LTS, Atrium LTS, structured?

[0:26:24.4] JK: The engineering org is we have a bunch of different engineering teams I'd say that correspond to different legal functions, work with layers very closely on different areas of the business, kind of on different types of work that's done. Atrium LLP has a team that does financing, for example. Then there are engineers that work with that financing team as consultants to basically build software for that financing team that just works on kind of Silicon Valley style finances. Then there's another team that works on documented taken processing that's doing some of that document extraction and metadata extraction I talked about. It's very cross-functional from an organizational standpoint.

[0:27:10.2] JM: The difficulty seem more in product management than really hard engineering problems. Is that the case right now?

[0:27:17.6] JK: I think that document extraction and structured data side is actually a very interesting engineering problem. I think there are two types of engineers that would — There's probably three types of engineers that would want to work at Atrium, at Atrium LTS, and those three engineers — The first is like kind of product engineers who want to build, kind of like you were saying, product engineers who want to work very closely with their customer, in this case, it's lawyers and paralegals, to build products that are used every day and they can have a very short feedback cycle.

So I think that's one set of people that it would appeal to. You're sitting nearby your best customer, and it's possible for you to really get feedback in a very short term basis and roll something out and see change and see people like really be thankful. You're talking about building stuff for an industry that really doesn't have a lot of like engineers building things for it.

The second type of work an engineer who might be interested is really — I would say people who want to build that platform, that data platform and maybe worked on machine learning. It's people who — We have this interesting dataset of corporate documents where there's, like I said, all the structured data of your company. It's like it's described in this dataset.

The other thing about it which I think is very interesting is that the dataset is like — I think it's a very good dataset for — it's like as a machine learning problem, because it's like very constrained actually in a lot of ways. I think it's interesting to think about how you can build a platform, extract this data, and then you build application on — Like structured in a way that you could build applications on top of it. That's like the second type of engineer I think who might be interested.

The third type is someone who's either been a founder or a very early stage engineer at a startup, because we are — Atrium LTS's mission is really to empower legal services for startups to be faster, more transparent and more with upfront pricing. People who have had that pain would be interested.

[0:29:27.7] JM: Why is that upfront pricing model so important for Atrium LLP to be different?

[0:29:36.4] JK: Yeah, I think it's very important for the adoption of new technology for us to have a reason to adopt it. Atrium LLP is a — When you're on the hourly business model, you don't have any — There's no incentive for you to actually reduce cost, and part of reducing cost and being more efficient is like adopting new technology. One of the reasons you see these law firms not adopting new technology is there's no internal incentive. By having fixed pricing, it doesn't really matter whether it's higher margin or a lower margin at first. What matters is your internal incentive is to innovate and reduce cost.

[0:30:13.8] JM: How similar are the operational day-to-day work, the cases for Atrium. How similar is it to a normal law firm?

[0:30:24.9] JK: I think that Atrium LLP's workflow is different behind the scenes, but the work product output ultimately is very similar to an existing law firm. The output is documents. It's review of documents. It's talking on the phone or email advising to clients. Really, a lot of the interactions are powered by a software behind the scenes.

[0:30:55.0] JM: Something that I have enjoyed about engineering is that we learn the tools for breaking down pretty much any concept in the world and understanding it and it can take a long time, but as an engineer you figure out that almost anything is possible to understand. This is the approach that you're taking to the law. You have no formal education in it. There's no reason why you should be starting a legal company. What has been your personal process for learning about the law?

[0:31:25.8] JK: Sure. Similarly, that's the reason I like engineering. I was always a pretty shitty programmer. I was a self-taught programmer. I was actually a good web developer. I say I was a good web developer and a shitty programmer, but I love programming and building web apps because it unlocked my creativity. You could feel like anything is possible. You could build anything as long as you can think about how to structure it and define the problem and define the outputs and the inputs. That's kind of the process.

I would say we've taken very engineering process towards the creation of this like legal software and services, and my thinking around it is really — The process I've gone through I would say is like, first, I was like gathering data, talking to tons of partners and lawyers out there and clients and investors. I'm like, "What are the problems in the industry? Are the problems that I experience the same as the problems that other people might have experienced," and really gathering like what are the reasons why there's lack of innovation. Why aren't they using software? Why don't they even use CRM software or Asana or a project management software?

Then once I had like a really good mental framework for how the industry operated there, I was like, "What are the entry points where we could create something that affected this industry and changed it for the better to like provide these values of speed, transparency and price predictability to startups?"

That's how I approached the problem, and then building a company, I think building this legal company is no different than building any other company, which is just to say, "What are you thinking about? What are the problems? What do I want to be? We want to get to a certain milestone, certain revenue milestone, certain number of customers, whatever it is." Then working backwards to like what are the biggest step barriers from getting there today and what are the things that I needed to do to affect those? Often, it all breaks down to — Pretty much, it always comes back to hiring great people and making sure they're focused on the right problems.

[0:33:36.3] JM: Are there any areas of the law that you've tried to study but they remain too difficult and too obscure and you just still feel like you don't get them?

[0:33:47.1] JK: The funny thing is I actually — I'm one of the easy clients. I think I was one of the clients who never really read any legal documents and didn't really dig in that much. I'd say legal for me was always a barrier between what I actually wanted to do, which was fundraising or selling a company or whatever. It's just something that I had to get through.

It's funny because I don't actually — I think I understand a lot now through working on Atrium LTS. I've understand the steps of a series A financing. I had actually raised millions and millions, tens of millions of dollars without actually understanding what the steps of a series A financing were. I do — It's funny. I have like more knowledge, legal knowledge now, but that's not really my interest. My interest is — I would say the content of like legal work is not my interest. My interest is applying engineering and product to affecting the delivery of that content to make it more efficient and make it a better experience for all the participants, whether it's the lawyer or the client.

I think that's funny, because I actually like — I'm learning much more about the delivery of legal and business of legal and the technology of legal than I am about the content of legal, which that's what I'm interested in.

[SPONSOR MESSAGE]

[0:35:14.1] JM: Do you have a product that is sold to software engineers? Are you looking to hire software engineers? Become a sponsor of Software Engineering Daily and support the show while getting your company into the ears of 24,000 developers around the world. Developers listen to Software Engineering Daily to find out about the latest strategies and tools for building software. Send me an email to find out more, jeff@softwareengineeringdaily.com.

The sponsors of Software Engineering Daily make this show possible, and I have enjoyed advertising for some of the brands that I personally love using in my software projects. If you're curious about becoming a sponsor, send me an email, or email your marketing director and tell them that they should send me an email, jeff@softwareengineeringdaily.com.

Thanks as always for listening and supporting the show, and let's get on with the show.

[INTERVIEW]

[0:36:16.2] JM: Totally. I was looking at some of the clients that Atrium LLP has worked with so far. One of them — That was Protocal Labs, which is they've made Filecoin and IPFS and they had an ICO. Have you learned anything about how the law views ICOs?

[0:36:35.8] JK: Yes, which is the Silicon Valley that law firms are mixed on ICO. Some of them are very, "Hey, we're going to support this and do them," and other are like, "We don't want to touch that, because we think there are massive legal — Security law problems around it."

I would say that right now it's a lot of unknowns and people have not tested this thoroughly against regulators and perhaps like against a legal framework that exist today. I think it's mostly remains to be seen. I think this is all compounded by the fact that many ICOs right now — I, by no means, an expert in crypto. I am an investor in one crypto fund polychain and I have some Bitcoin, but I'm not like — I'm not someone who's like cutting edge and like up-to-date. I rely on my friends for that mostly.

I would say that I think I believe that many of these ICOs that are happening right now are purely gold rush and there's no reason for their software to be a distributed ledger. That makes sense only for certain things, like certain user cases. Not every use case doesn't make sense for it to be a distributed ledger system. Not every use case doesn't make sense to have like a separate token of value.

I think in Filecoin's case, it makes a lot of sense, but there's a lot of like companies, like the Kin Token, for example, the Kik's token, like Kik Messenger has its token. It's like I don't understand what the economic point of creating a cryptocurrency is except to raise money for a corporate vehicle, which I think is just — If that's the case, then it seems like purely a way to circumvent securities law.

[0:38:22.7] JM: Yes. Okay. Yeah. Are you just speculating or have you had a sense from talking to people that —

[0:38:29.9] JK: No. I'm just speculating. This is not based on my — I have no — This is my business hat, business person, Silicon Valley investor hat. It just seems like a lot of these things — It's like kind of a gold rush — If not frauds, there's no like actual technical and economic business model reason for them to like actually be a cryptocurrency.

[0:38:52.3] JM: Sure. I see a new Twitter add for a new ICO every day.

[0:38:59.1] JK: Yes. To me, that's very sketchy. I get emails that are like cold solicitations that participate in these random ICOs. Yeah.

[0:39:12.0] JM: This upfront pricing model that Atrium has where you keep prices fixed on work. I'm a startup founder, I come to Atrium LLP and I say, "Hey, I need this task done," and Atrium tells me, "Here's the price," upfront, and that's great. That's a weight off my shoulders where otherwise I'm already worried about how much cash is in the bank, and if I have to also worry about, "Okay. Hey, random old world law firm, I need to get this task done." They say, "Okay. Cool. We'll let you know what it costs when we have it done in three weeks." That's a burden, and I don't need any extra burden.

Getting the prices fixed on work that is going to take a variable amount of time from case-to-case, that seems challenging, but it also reminds me of — We did an interview with Gigster a while ago, and they do something kind of similar with their contracting model where they have big software projects that come to them, and they learn overtime how to price these basically by tracking the data really closely. What's the approach to getting fixed prices on what have been variable priced purchases in the past?

[0:40:26.3] JK: I think there's probably two types of work. There's work that varies like within a certain bell curve every time. It's like more known, and there's lots of data points. That's work like series A financings or series B financings.

Silicon Valley law firms who have been around for a long time will have like thousands of data points and they should be able to build a model that says, "Okay. Given your parameters of the things that are coming, like the firm, how much money it is. How many shareholdings, incumbent shareholdings you have, all these stuff, they should be able to build a model that says, "Here's the distribution of likely outcomes of how much work it's going to take," and we're going to just price that at the percentile mark or something like that, but they don't, because there's no incentive too.

So I think it should be possible to build a model for many kinds of work that is based on the data that actually is accurate enough that you can basically absorb the risk of variance. That's all we want to do.

[0:41:25.8] JM: Actually, the model with Atrium LLP is this is the proof of concept of a set of practices and technology services that a law firm can adopt. What's the model for deploying that at other law firms? How much data do you need to collect within Atrium LLP to get other law firms enticed, or are you already in conversations with people who are interested?

[0:41:52.8] JK: No. I would say it's pretty new, and we need to prove — Because lawyers are very resistant to changing their workflow. I think there's going to be like many prove points that we have to hit before other — It's even functional at other law firms.

Our goal really right now is to like learn and get data from how you can improve processes and what technology platforms we can build to actually prove lawyers work, and that's kind of where we're at right now. Yeah, our goal is to like just be — Right now we're in the learning phase, I think.

[0:42:27.4] JM: Has any part of the product development been harder than you anticipated?

[0:42:32.2] JK: It's a good question. Has any part of the produce development —

[0:42:34.7] JM: Because I remember. I saw an interview where you were talking about how you've been thinking about this — Building this company for a while.

[0:42:41.7] JK: Yeah. I think that we are trying to do a breadth of things right now, and so that is — Things are — I think we're working on many different types of legal work for startups, like tools for legal work for startups. There's a little bit of — I think we can make more progress probably if we constrain the scope to fewer things that I might work on, and I think we might want to do that.

Overall, I wouldn't say it's like harder than anticipated. It's a pretty straightforward. I think there are interesting engineering challenges, like I said, on that kind of extracting structure data side, but a lot of the other software, it's like fairly straightforward in terms of how to develop it.

[0:43:24.0] JM: Yeah. That issue of focus, that comes up a lot when you talk to different startups, and doing multiple things at once is really hard. The problem is you're in a space where you want to sort of change the whole mindset of the space. It's not like where you're Amazon and you're like, "Okay. Let's focus only on books first," and then it's very obvious how to laterally expand to DVDs. It seems like you need to go whole-hog.

If you were to constrain the focus to — So one thing specifically, would it be Safe Notes or something? It seems hard.

[0:44:04.0] JK: I think there are two major pieces of software that are very important. One is the data extraction and platform, and the second is the client facing portal that gives them insight into what's going on in their legal work. Those are probably the two things that we really need to make the most progress on, I think.

[0:44:27.1] JM: Interesting. What's the most grandiose vision of what LTS could evolve into?

[0:44:35.4] JK: I think LTS could power over the biggest law firm in the world. I think Atrium LLP could be the biggest law firm in the world and LTS could power it. It could power many other law firms outside of that, and it's kind of the big vision. I think that this is a new model for unlocking the $160 billion of outsource corp and legal spend. That's my goal.

When I decided to go back into starting a company, I had been investing and incubating companies for a couple of years, partner on Y Combinatore. Before that, started a bunch of companies. Twitch is the one that people know about.

What really got me to go back and say, "Hey, I should be the CEO of a company again," and really dive all in, is that I really believed that this was a big market and a big opportunity and I really want to try to build something that's bigger than Twitch. Yeah.

[0:45:32.3] JM: Okay.

[0:45:34.1] JK: I think this market supports it.

[0:45:35.3] JM: It's interesting. I thought that the model was to build technology and then give it to lots of other law firms, but I could see it being equally plausible where you just grow Atrium LLP.

[0:45:46.5] JK: I think we try to grow Atrium LLP and also like try to figure out how we can sell the software to other law firms as well. I think those are parallel paths. The goal is that legal revenue out there, like the most — We can have the largest amount of legal revenue flowing through this like LTS software as possible. If that makes sense.

[0:46:11.1] JM: Yeah. Definitely. I do want to talk a little bit about Twitch, just because I've heard some random interviews where you were talking about some of the engineering issues at Twitch, but I had never heard anything where anybody went super deep. I know there was a lot of custom software. That's a serious tech company. What were some of the engineering problems that you recall from Twitch?

[0:46:37.1] JK: I haven't engineered anything on Twitch in a long time. Caveat everything with that. Twitch was built on top of the infrastructure that we used for Justin TV. Most of my engineering work was from the Justin TV days, and then eventually it was rescanned into Twitch, and I think a lot of that course infrastructure stayed the same.

We were one of the biggest Ruby on Rail sites for a while, maybe still are, but that was a big challenge actually. There were two probably big challenges. One was just the building of video infrastructure. Live video infrastructure was like not mature and so we had to build a bunch of basically software around like the live video servers that would route the video to various points of presence and kind of like grow the number of like — Basically, the video would come in to an origin and then go out to like various slaves like all over, sometimes in the same data center in San Francisco, sometimes in multiple data centers if it was a popular stream.

The problem was like because we were supporting streams, some would be like 10 people watching. Some would be like 100,000 people watching. We actually didn't know Apriori, unlike an Akamai, which pre-provisions. We didn't know Apriori, which streams would be super popular or not, right?

You had to have this system that could be very reactive, and if stream was getting really popular, would populate it to many different servers all over the world, right? That was a huge engineering problem. That took a long time to get right from a stability and being robust standpoint. That was one big engineering problem.

The other was just getting this like Rail site to scale. Twitter had a huge amount of problems during the same timeframe, like 2009, 2010. Just getting this like Ruby on Rails site to scale, we ended up building this basically middleware on Twisted that basically cache, like it would statically cache all these Rails pages. The application service took forever to render. Then inject the — It would have like a mem-cache store of like all the user data and it would inject — It would have these custom tags that would like replace them with the user data. Actually, Rails would render the first, like the cache version and then this like custom server would like interject the actual user data, because Rails was too slow at the time. I think there have been massive improvements now. All of these is from — It's almost 10 years, 2008 to 2010 I would say, or 2008 to 2011.

That's how we scaled to, at the time, is like hundreds of millions of page views a month. That was big at the time. I'm sure it's dropped in a bucket now. It was fun. I learned a lot about engineering. I hadn't been really formally trained. I wasn't a CS undergrad. I also never — I never worked at a big company or anything. I started our first company right out of college. It was a good tutorial in kind of like engineering and engineering management for me.

[0:49:43.7] JM: What are some takeaways from that, from the management process?

[0:49:47.1] JK: I remember debugging — The site would go down. There's a live video site with like our peaks to value ratio was like 35-X to like one. It was very hard to like to scale. Most other sites are like 1.5, 2X and peaked at like to troff usage.

It was just crazy. Every time something would fall over, it was like always one of six things. It was like bandwidth to servers, like CPU memory. The site would go down and I'd be just like, "Okay. Let's look at Nagios," and it's like, "What's getting fucked here?" "Oh, it's like IO." It's like, "Why can't it write fast enough to disc here?" Then you just like debug it. Like it was so much live debugging of how can we like shut off a feature or like read-write something right then to like make it work right now, because it's like a live video site. This is probably a terrible way to actually build something, but it was so hard to predict what our usage was and we're also operating on a very lean budget from an infrastructure standpoint.

When we sold Twitch, we have like three network engineers or maybe five network engineers. Like Hulu, which was less traffic, I think network engineering of 50. We were operating, we were very under-funded the whole time. We raised $45 million over the life of the company, but that's over eight years. I would say that our infrastructure, we were very efficient at infrastructure, but it came at the cost of like we never had excess capacity. We were often times tweaking features in order to actually have the sites stay up, which is terrible from an engineering standpoint, but it was very instructive in terms of — I don't know. It's funny. I haven't thought about this stuff in years. It was fun.

[0:51:37.8] JM: Did you learn anything about managing your own psychology in that process?

[0:51:41.4] JK: Sure. I don't think I implemented it super well. I think I'm much better at it now, actually. At the time, it was so stressful, because sometimes the site would just fall over and we had a major event going on or like the Jonas Brothers was streaming or something, and we would be so stressed out. We would be just like looking at each other, like, "This is —" Like what can we do? We're like literally staring at like a 500 application error in Chrome and be like, "How do we turn off features so that this live event can happen right now, or how do we rewrite or cache more things or like what can we do to like fix this?" Yeah, that was very stressful — Oftentimes very stressful.

[0:52:23.4] JM: Interesting. Is there anything else you want to add Atrium LTS, Atrium LLP and where the project is going?

[0:52:30.7] JK: We're hiring right now. Yes. I think it's a pretty fun project. I think it's a pretty interesting and unique project, and it's going pretty well. Atrium LLP has interesting clients, a ton of clients right now and it's pretty cool to build software that people use every day. I think there's very interesting data play. That to me is the exciting part from my engineering side. It's like the thing that I get excited about. If you're interested, just reach out and let us know.

[0:53:02.2] JM: Great. Justin Kan, thanks for coming on Software Engineering Daily.

[0:53:05.1] JK: Thanks a lot.

[0:53:05.5] JM: All right.

[END OF INTERVIEW]

[0:53:08.8] JM: Simplify continuous delivery GoCD, the on-premise open-source continuous delivery tool by ThoughtWorks. With GoCD, you can easily model complex deployment workflows using pipelines and you can visualize them end-to-end with its value stream map. You get complete visibility into and control of your company's deployments.

At gocd.org/sedaily, find out how to bring continuous delivery to your teams. Say goodbye to deployment panic and hello to consistent, predictable deliveries. Visit gocd.org/sedaily to learn more about GoCD. Commercial support and enterprise add-ons, including disaster recovery, are available.

Thanks to GoCD for being a continued sponsor of Software Engineering Daily.

[END]