EPISODE 347

[INTRODUCTION]

[0:00:00.9] JM: Healthcare is a complex business. Oscar is a company that wanted to build a new insurance provider but they realized that healthcare is so interconnected that in order to build a new insurance provider they actually needed to build an entire healthcare business too, complete with patient management and facilities.

Since Oscar is a modern technology company, they focus on customer service and engineering and data management, and this company offers an optimistic view into what healthcare might look like in the near future. Every time a patient interacts with the healthcare system their insurance provider has an opportunity to collect data on that interaction.

Isaac Councill helped architect the infrastructure at Oscar that manages and analyzes this patient data. In this episode we talk about the healthcare system; the data engineering of Oscar, and Apache Mesos, which Oscar uses to manage its applications. We have an interesting discussion of Apache Mesos versus Kubernetes, and I hope to do more shows about the evolving healthcare technology space. It's really intriguing me right now.

[SPONSOR MESSAGE]

[0:01:23.9] JM: Artificial intelligence is dramatically evolving the way that our world works, and to make AI easier and faster, we need new kinds of hardware and software, which is why Intel acquired Nervana Systems and its platform for deep learning.

Intel Nervana is hiring engineers to help develop a full stack for AI from chip design to software frameworks. Go to softwareengineeringdaily.com/intel to apply for an opening on the team. To learn more about the company, check out the interviews that I've conducted with its engineers. Those are also available at softwareengineeringdaily.com/intel. Come build the future with Intel Nervana. Go to softwareengineeringdaily.com/intel to apply now.

[INTERVIEW]

[0:02:19.4] JM: Isaac Council is the VP of engineering at Oscar. Isaac welcome to Software Engineering Daily.

[0:02:24.5] IC: Thanks. It's great to be here.

[0:02:25.7] JM: I want to begin by discussing health insurance and why this is a technology problem, and then we will work our way towards the actual engineering. Let's start with a naïve question; what is health insurance?

[0:02:41.1] IC: Yeah, that'd be a great question. It's absolutely not just a technology problem, and technology is we think a part of the solution here. Health insurance is really — It's an interesting entity that is a really crucial player in healthcare. Just thinking about what we are as a health insurance company, I think I have to discuss a little bit about the Affordable Care Act and how that changed the setting. That's really about why we're here.

You can look at health insurance as being traditionally in America. It's been a products that is sold mostly to employers in order to cover the insurance fees, the medical costs of the employees of a company. That's been the traditional model in America. Companies get tax [inaudible 0:03:35.5] they're incentivized to provide these benefits and in face legally responsible for providing these benefits.

Now, that insurer is responsible for taking claims from the medical care that is encouraged by the employees of these companies and paying according to, usually, negotiated fee schedules with the doctor networks, the doctors, the hospitals, or the pharmacist in the case of drugs. We're the payer, typically. When you hear the word single-payer, which has been a buzzword that's been going on for the last year. You heard a lot in the election cycle.

Right now, we're one of the multiple payers because there're multiple insurance companies. We're all competing with each other to get the best rates with the providers. We're the payer in the system. We have a relationship with traditionally health insurance, we'll have a relationship with an employer, so they'll have the secondary relationship with the employees, the people that

are actually are generating the bills. Also have a secondary relationship with the providers because they're the once who get paid. That's it in a nutshell.

I should go on to talk about how that relationship has changed through the Affordable Care Act, where the Affordable Care Act is really shaking things up so that's we're still the payer, but the relationships are entirely different.

Now, because of the individual markets for the first time, you have a broad swaths of the public who are able to engage directly with the insurer. Now, we've gone — The industry has gone for selling basically enterprise insurance to the enterprise and so now we're selling consumer product, almost. We're selling product directly to customers. That really changes things in a very very fundamental way, and that's where technology starts to creep in, in a very obvious way because now we can talk about technology as the industry is currently thinking about mobile applications, desktop dictations, portals for actual member experience, which was a really secondary concern before.

We really benefit from having that deep relationship with members, which wasn't necessarily the case before. It used to be a case, remember that the employers were the ones who — We want to be cozy because they're actually paying us, and so the people who are engaging with healthcare, they cost us every time we interact with them. Now, it's completely changed with Oscar, with all the other insurance on these individual markets. We're engaging directly with the members or the ones who are paying. They are also the ones who are receiving the benefits from our service. We provide them a great benefit while they can just go and find another insurer, which really should be the way that it works after all.

[0:06:40.2] JM: You discussed this difference between the model where I get health insurance through my work place where my insurance is tightly coupled to my workplace versus the idea of health insurance being provided or being purchased by me. I'm not precisely sure how Oscar works, so I guess I didn't quite understand that aspect of your explanation. All the companies that I've worked at in the past, I've gotten health insurance through the company and it set me up with this tight coupling to the company where it might be hard to — If I switch jobs, then I might not get to keep my doctor and it raises the question why is this healthcare plan even being provided by my employer. Since then, since leaving the workplace and becoming

employed by my own company, now I get individual health insurance that I purchase outside of the company that I worked for. The health insurance quality is probably not as good, but at least I am completely in control of it can.

Could you contrast those two models a little bit more and explain what Oscar does relative to those models?

[0:07:57.3] IC: Sure. I don't understand why it wouldn't have to be as good. Hopefully it would be at least as good. The model being fundamentally different, I think, is really the key. First off with, with your employers, have you ever gotten to choose which company you want to be insured with? Which doctors, doctors' network, that sort of thing? It's fairly unusual.

[0:08:19.3] JM: I don't remember.

[0:08:20.3] IC: Yeah. Oftentimes, you get to choose the plan, the EPO, the PPO, all that stuff but you don't necessarily fundamentally get a choice between Aetna or Cigna or all the various options that are out there. All of that negotiations, all that choice was done by the employer, and so the employer is basically trying to get the best policy that they can for the maximum or the best considering the population of employees. That's not necessarily done with you individually in mind trying to do the best they can on the budget that they have, but maybe there is a better product for you. Maybe you're interested in something of a different offering that some of the company or some other plan provides. That's been that market for us which has been just absolutely absent.

Really, opening up that individual choice is actually normalizing the insurance market or at least that's the hope, that's the promise, is normalizing the insurance market to be a fair and free market in a much broader notion than what it has been before. You have individuals making a choice for them, for themselves, rather than that choice being made on their behalves by the employers.

[0:09:37.7] JM: When a company sells you health insurance, so when I buy health insurance from a company, they are making a bet on how my health is relative to the price of care that I may need to receive if I have a health incident. Can you explain how a health insurance

company calculates the expected value of that bet and how they price insurance for a given customer?

[0:10:07.0] IC: Sure. First off, we don't price for individual customers under the Affordable Care Act. It's actually one of the really nice things that's under heat right now. You might've seen if you're following what's happening in DC right now, there's a lot of exciting talk about removing some of those provisions from the Affordable Care Act. Right now, our plans have a fixed price, and so that fixed prices is for everybody who wants to choose those plans. We don't price you up based on some pre-existing condition, some pre-existing set of factors. That's one of the things that the Affordable Care Act has actually removed.

Getting back to that in general, I've talked about the Affordable Care Act actually changing the nature of the relationships in this industry and also removing individual pricing based on pre-existing conditions. Those are really wonderful things that the Affordable Care Act has done. There is I don't think anybody, you'll find anybody who's really studied the Affordable Care Act that doesn't come to the conclusion that it has some significant flaws and has some places where it could benefit from improvement, but it's fundamentally changed the expectations around insurance, and that's really what a lot of the things that we're here trying to fully realize, to help realize those visions.

Getting back to your question, we price based on actuarial risk, the same as any other insurance, auto insurance. Then there's a lot of actuarial tables that can be used to calculate the likelihood or what's the statistical probability of getting in a crash and what's the distribution of expenses for those crashes. It's a really a statistical problem. The same goes for health insurance. What's the statistical likelihood steps? Are you going to need care, and what is that care going to cost? The distribution of cost, it can go anywhere from just a routine checkup to being diagnosed with some kind of horrible cancer which would be very costly.

We have to keep in mind all the probabilities and the cost distribution and at the end of the day you have to be taking in more money than you pay out in terms of the medical expenses for your membership base. That's basically the key metric for us as an insurance company. It's called the medical loss ratio. The medical loss ratios basically what we're paying out to providers, to pharmacies, medical spend, versus what we're taking in revenue.

The idea is to have that medical loss ratio be around 85%, something like is considered — Anything under 90 is considered very very healthy. We have to fit all of our operations, a lot of our operational expenses in the area of the MLR up to one. That's basically what we have to keep in mind with the pricing. We have to protect the overall medical expenses for the membership if we're trying to hit that target as precisely as we can, that our medical spend is going to be right around those numbers, somewhere in 80 to 90 range is what we're hoping for.

[0:13:39.7] JM: You could build a health insurance business that is better than the existing competitors by just building a better UX, because I I've interacted with my own health insurance provider's website and it's really frustrating experience. I wish I could just pay $10 more a month to have a better UI.

I haven't purchased through Oscar, but just from browsing the website and clicking through things, I can understand that the UX of the product is much better. Let's start to talk about the engineering. Describe the basic architecture of the user facing application. If I'm a user who wants to buy health insurance through Oscar, what is the application that I'm dealing with?

[0:14:26.3] IC: Sure. I've got to say, I'm really happy to hear what you're saying now, that's the UX is such an important consideration. That's a particularly entertaining thing because that was one of the things that we heard a lot when we're just getting started around 2013 or early 2014 from these other insurance companies or from other academics in the field. It's like, "Wow! Oscar, that's really cute." What they don't realize is that nobody in this industry cares about the websites. They're talking about single-digit engagements to websites of insurers out there that they don't get excited about maybe five or six.

We saw early on that providing a really nice experience, really providing some very valuable functionality drove engagement order of magnitude more than what folks were telling us we could expect the in the best case. We started seeing numbers like 70% engagements month over month. I think what you said really bears out.

To get back to your question about what that the tech stack actually is. Recently, we've gone really all and in React. We just absolutely love React when we moved over to it. Just see that

the unit test coverage skyrocket and the code base which makes me super happy. It's also just enabled a ton of really interesting stuff which we could talk about if you're interested.

In the JavaScript layer, typically, all the new stuff that were moving to is in React. There might be some legacy that isn't there that's going away. It's on a flask application. The flask application is being launched. It depends on which product you're talking about. In general, we launch all of our applications inside of Mesos. We have a particular zone in our Mesos cluster just for user facing applications. We're typically launching into that, these flask applications that serving out rehab to clients. We're using HAProxy to route these things. In a nutshell, that's kind of what's going. We're using AWS right now, if that's of any interest.

I also want to emphasize this thing that you see, the websites and what a member is actually going to engage with. It's really just actually the tip of the iceberg when it comes to engineering at Oscar. We're doing a lot of exciting stuff on the front-end for members and I think we have a very exciting program going forward. We also have a ton of application development for internal use cases for our member service representatives. For example, our internal operations, and then we have hundreds of applications that are just moving data around collecting data from the outside. It's actually a wonderfully complex space, and so that member website really is just the tip what you can see.

[0:17:36.2] JM: Of course.

[SPONSOR MESSAGE]

[0:17:44.2] JM: Your application sits on layers of dynamic infrastructure and supporting services. Datadog brings you visibility into every part of your infrastructure, plus, APM for monitoring your application's performance. Dashboarding, collaboration tools, and alerts let you develop your own workflow for observability and incident response. Datadog integrates seamlessly with all of your apps and systems; from Slack, to Amazon web services, so you can get visibility in minutes.

[INTERVIEW CONTINUED]

[0:19:09.1] JM: I want to get into the customer service backend applications as well as the data engineering applications, but let's talk real quick about Mesos and your infrastructure orchestration layer. You said you're launching these flask applications on Mesos. Are these being launched on VMs or in containers?

[0:19:31.6] IC: In VMs right now. The containerization thing is kind of a fun topic. We've been avoiding Docker and a lot of the containerization out there for a few years now mostly because I think a lot of us just didn't feel like it was quite ready for prime time. Keeping the Docker one point over these was built on dot releases. We have some substantial underlying parts and it's been really exciting to watch it. We've really set back for that.

We're starting to go with containers now. I think that the space has matured quite a bit, so we've got a lot of reliability going on. You see other options going on with using Docker images in other ways, like the Mesos Containerizer for example is something that I'm very excited about where we can get the benefits from containers but without necessarily extra services and extra complexity going on. We're going to be using Docker.

Basically, what we decided really early on as we're going to keep it as absolutely simple as possible and we're going to be disciplined as developer. A lot of us were at Google before. I was at Google before. It was helpful to be kind of brainwashed in the same space.

At Google, the mantra is sort of single statically links binary is really the key to beautiful deployments. You build. You make sure that your image is common. Your build is done on a machine that mimics the production environment and when it gets time to deploy, you just downloaded thing and you dot slash around it, and it's great.

Then we have — With Mesos, we get nice things like C groups, so we can allocates just the right amount of RAM, CPU, and disk and you hold your process to that. That means we basically have all the containerization that we need that we have — We have far resource isolation. We have our artifact isolation. We get the benefits of a straight up UNIX security that way. It's very very simple.

The only time that we really get into trouble is when we need something weird, so like I somebody who wants to use a LibreOffice, like PDF rendering engine just to pick on a recent case. Then, yeah, it's kind of annoying to build a few Mesos nodes that have just — That have the special stuff on them so that we can use the shared libraries. A container would be lovely for that to be able to still load up a Docker images with LibreOffice and just go to town. Really, discipline first, and so Mesos really, I think, helped us develop some very very good habits early on.

[0:22:18.7] JM: The reason that I've heard people move from virtualization to containerization, at least one of the reasons, is that when you go to containers you can break up the VMs into units that are more properly sized to the jobs that you are allocating to those units. If you do everything in terms of VMs, a VM has a floor to how small you can make it and a lot of times you're going to spin up a VM, you're going to use it for some kind application and that application is not going to take advantage, take full advantage of the resources available in that VM. Whereas if you sliced up into containers, the containers can fit more exactly to the size of the jobs. Is the —

[0:23:09.9] IC: It makes sense. Mesos really is the way we get around to that, because with Mesos — Mesos is lovely. It basically handles resource allocation by Mesos itself and the scheduler puts your jobs where it sees fit.

Just to talk about the Mesos model a bit, it really takes your virtual machines and turns them into globs of CPU, disc, and RAM. Now, the VMs typically are going to be consistent. We don't really care about them. We don't launch a VM with a particular role. We launch a VM with a Mesos role and then we use a scheduler on top of Mesos. Right now we're using Aurora which was developed in Twitter. We love it. It's been just amazingly reliable for us. The schedules allows you to use the globs of resources that Mesos made for you and it just slices them off.

If you tell it your job, it's .5 gigs of RAM and one CPU. It's a hardcore, non-threaded sequential process to run fast. So you tell your scheduler to launch this job with those resources, it asks Mesos to basically carve that amount of resources out and then create a C group to container, if you will, to launch your job end. You can get all those benefits of very high utilization. We actually have a wonderful utilization with Mesos because we're launching maybe a ton, maybe 100 jobs on a single VM, and because everybody built their jobs off of the same image which that VM is running. We're using Mesos or a scheduler in order to slice off little pieces of that VM and launch the job within it. Actually, it's wonderfully cost-effective and also just incredibly convenient. Mesos and Aurora together give us beautiful orchestration that the same kind of thing that Kubernetes is going to give you with these all stack underpinnings. Yeah, we're definitely abstracted away from the VM. We just didn't need containers to get there.

[0:25:44.0] JM: What are the operational differences between somebody running a Kubernets cluster with Docker containers and somebody running a big Mesos cluster, where Mesos itself is just abstracting away the collection of VM's and allocating jobs that are across this uniform address space of Mesos stuff?

[0:26:06.0] IC: Not much. Not much really. It's the different way of getting the same thing. Kubernetes will give you Docker. Mesos gives you the option of having Docker.

[0:26:15.5] JM: Okay. Interesting. All right. Now that we've kind of covered the orchestration layer, let's talk about data engineering and then maybe we can get into the customer service stuff. The data engineering, it really starts to get interesting because you've got all these data points that you can collect. If people are buying health insurance and then you can gather data points along their entire health journey because every time they engage with a healthcare

provider you're getting feedback about that. What are some of the data points that you're able to collect on the data platform?

[0:26:53.8] IC: Oh my goodness! Our general data space is made up at the very high level of — Whether there's members, or the people. Those people could be either members or they can healthcare professionals, providers. There are plans which are descriptions of the plan configuration that the people have signed up with. There's eligibility, which turns out to be a complex space because of regulation. That's just determining records to determine who is eligible to receive benefits under a plan at any given point in time.

Then there's our claims, of course, and those come in the discreet records that describe care that was those provided or received and there's authorizations that people are requesting, sending a pre-authorization to say, "Hey, this is going to be okay, or we're going to cover these." All these things. We also have agreements, opt-in agreements that we can detect healthcare events, like somebody gets checked in to an emergency room. We can actually get notified about that so that we can do our best to help in some meaningful way if somebody needs it.

All that gives you kind of a rough overview. There's tons more, but there's always so much to talk about in that space because there's so many problems inherent within it to solve. It's wonderful, like a provider data is a wonderful examples with the projects that at first glance might seem easy if you haven't had to deal with it before. It's like, "What's the problem?" You get a bunch of data about providers. You stick in a Solar elastic search or something like that and you search it. It's great.

The problem is that all of these data sources that are coming in, there's the classic problems of missing data, incorrect data, just bad formats, and there's duplicate, conflicting information. It's all just wonderful classic data pipeline stuff where you have to have a really big kind of interesting pipeline to not only do the records aggregation but to do entity reconciliation deduping, splitting different providers with the same name, that sort of thing. It's really interesting. It's really tough. We're going to have, as our membership, we're going to have exactly the same issues with incoming member signups. We're looking forward to that. The claims coming in need to be sometimes corrected, there's typos. You have to match the incoming provider information to providers that are networked, because if they're not in our

network, then we don't have a negotiated fee schedule. We don't know what to pay, or we don't know whether we should accept or deny the claim. It has to be this fuzzy matching process. The same thing with member information. These stuff that people are just typing in and all that stuff has to be handled correctly by our systems.

Oftentimes in the insurance industry, that's actually handled partially through technology but with people, with human actually entering that. We're just trying to get the accuracy up to as much as possible. All of these data is highly highly relational. It's sometimes hard to decouple and to figure out when member information is more clinical, more operational. It's really wonderfully interpretive at times. Anyway, lots of really interesting data pipelining issues. I'm not sure which part of that you'd like to dive into more.

[0:30:49.2] JM: Let's start at the beginning. Your first two years at Oscar you were focused on this full- time. What was involved in setting up the data pipeline early on?

[0:31:01.5] IC: It's not much except just parsing CSVs and trying to do fairly coherently. It's a wonderful thing coming into to healthcare is as fairly seasoned software engineer and previous research scientist research scientists came. into this space with a bunch of wide-eyed ambitious people and started staring at the problem and wondering what we should start typing into our terminals. It's just a wonderfully humbling experience because none of us really knew what we were doing. We were all learning for the first time.

I guess we know how to build data pipelines. We know how to setup databases, but we're all learning what this data meant, what the problems were going to be and how it all relates to each other. More so, how can we lavage this to actually let our customers have a good experience at the least and not contribute to a bad experience, which is always just a horrible thing if you see it.

Really a lot of the first couple of years which is really trying to get our heads around it and trying to get as many of these feeds hooked up as quickly as possible because there are tons of them. They're like 70 or 80 just in the first half year that had to get models fit in. we have to set up the connectivity with various places. There wasn't a whole lot of thinking about how we're going to

make some beautiful elegant system. It was just get that data in and get it somewhere where it's going to be useful as quick as possible.

Nowadays, we have that luxury of thinking about, "This would benefit from our graph databases, or this would benefit from JSON-B storage and PostgreS. Now, we can be sophisticated. Back then, that was a scramble. Not only were we ingesting all of those data, but we are doing everything else that was involved with making an actual insurance company that works, which is a fairly complex thing. Yeah, go ahead.

[0:33:04.1] JM: That has stabilized. The product itself has stabilized. What has happened with the data pipeline?

[0:33:10.8] IC: Oh! Now, you see wonderful things starting to happen. Right now, we were thinking in terms of what are the characteristics that you want to —

[0:33:20.6] JM: Could you maybe give me an idea of data — You've got lots of different streams that could potentially come in. Are they all hitting Kafka and getting fanned out to some different places? Maybe could you walk me through the life cycle of that data?

[0:33:34.9] IC: Yeah. I'll talk to you about some of the properties of the storage engines that we're doing as well. Yeah, there's a pattern that's starting to emerge where you have to set up the perimeter connectivity in some way. Oftentimes it's FSTP. FSTP is often like the replacement for our PC or Rest in this industry. We really like sending files to each other. Those files are in lots of various formats and so we have to —

[0:34:02.8] JM: Sorry. Not to interrupt you, but is that because a hospital is sending you this FSTP stuff?

[0:34:08.4] IC: It could be a hospital. It could be a provider network that's sending us a roster of people who are currently in the network every week and we have to figure out the differences, that sort of thing. Pretty much everybody, it's the de facto way of sending information unless there's some very compelling reason that this has to be a synchronous process.

That would be, say, a synchronous process might occur when you're connecting with a remote EHR, like a electronic health system. Then maybe you're doing — You're trying to let your member schedule a doctor's visit, so that sort of thing has to be done fairly synchronously.

With the de fact way of communicating is over FSTP. The whole industry is kind of cultured into a calendar-day information cycle. That's sort of way. It's easy. It's robust. We know how to do it. Yeah, it's great.

We're kind of used to that. They'll send us things in various formats. I can talk about this for probably too long because I love this stuff, but there's thing, EDI, electronic data interchange. Wonderful, wonderful thing. If you read the Wikipedia entry about this, you'll find out after following a couple of links that it got its start during the Berlin airdrop after World War II. There's this amazing logistical problem getting supplies into West Berlin for the allies and they were using these old modems to actually communicate. They need a compressed format to send it, and this thing has evolved over the decades and really took hold in the health insurance industry, the healthcare industry I should say.

We have EDI. There's U.S. dialect. There' European dialect of it. Of course, we pretty much deal with the U.S. thing. It's this wonderful, wonderful protocol which it can you can look at it — It's just a way of transmitting structured data but it's actually — You need a stack to understand it. It's a context sensitive parameter. When you look at it, it's really just amazing, then developing a parser which we've done because we didn't just love all the ones that we found out there. Developing a parser is one of the most fun things I've done in the last — I don't even know how long I can remember. Great, great stuff.

There's these old formats. There's like HL7 is another one which is less interesting than HTML. We get lots of CS fee, like stuff in various formats. We have to convert that into raw data, upload it to S3, stick an event on Kafka. We have a thing that this subscriber might be interested in, then you might download the file, transform it in some way, stick it in a database, use MySQL. We're kind of moving more towards PostgreS.

As time goes on, we can talk about why in just a little bit. Yeah, that's sort of how this stuff goes. In the opposite direction, it's pretty much exactly the same. We have to communicate with

people by dropping things down in whatever format they need in outgoing FSTP drop, or they could actually get it. Maybe we're going to send it to them over Rest or Soap. We just have to be prepared for just about anything.

[SPONSOR MESSAGE]

[0:37:47.2] JM: Spring is a season of growth and change. Have you been thinking you'd be happier at a new job? If you're dreaming about a new job and have been waiting for the right time to make a move, go to hire.com/sedaily today.

Hired makes finding work enjoyable. Hired uses an algorithmic job-matching tool in combination with a talent advocate who will walk you through the process of finding a better job. Maybe you want more flexible hours, or more money, or remote work. Maybe you work at Zillow, or Squarespace, or Postmates, or some of the other top technology companies that are desperately looking for engineers on Hired. You and your skills are in high demand. You listen to a software engineering podcast in your spare time, so you're clearly passionate about technology.

Check out hired.com/sedaily to get a special offer for Software Engineering Daily listeners. A $600 signing bonus from Hired when you find that great job that gives you the respect and the salary that you deserve as a talented engineer. I love Hired because it puts you in charge. Go to hired.com/sedaily, and thanks to Hired for being a continued long-running sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:39:18.2] JM: You're talking about a lot of ways that data can come in. Once it's in to this, system what are you doing to extract value from it? You could deliver it to data scientists, you could somehow integrated back into the customer service platform, you could integrated into the customer facing application. What are the different things you can do with that massive data once it is ingested and indexed and normalized?

[0:39:43.3] IC: Just unifying it and presenting a cohesive view that [inaudible 0:39:47.1] is really kind of the dream just so that you — Developing a truth is actually quite challenging. You got missing information or like incorrect information that are coming in. Also when you have — We can transform things in such a way that business decisions are made based on data that was buggy. What do we do about that? Actually constructing a storage layer that lets us have that kind of unification, the kind of flexibility, but also lets us really have a very tight understanding of data overtime is really kind of the first step.

We could look at this as being a nascent effort at Oscar, but you see a lot more of our stuff ending up what we call — It's a standard term, bi-temporal storage. What we mean by that is that we store things, generally as relational. It doesn't have to be, but we store it with two different dimensions of time. There's the valid time which is just when was a particular assertion true. If it's in unbounded, maybe it was true from today till the end of time, or maybe it was just true. Maybe we're talking about a membership, or a member's eligibility stand. Then it has an end date. This was true from January 15th to March 1st when they called in and canceled. I don't know.

We have those validity periods which can't be overlapping for a single assertion because that would be a conflict. But then we also have the second dimension of time, is the transaction time. That's basically what did the system think was true at a given point in time, which you can get a very different answer from what was true at a given time. That would be, say, for a bug. Let's say we improperly cancelled somebody's membership on March 1st then retroactively a week later we instated that we instated the plan. The validity period was at one point, say, January 1st to unbound it. Then it got truncated to January 1st to March 1st, and then it got retroactively corrected to January 1st unbounded again.

There's that week of time where we might have made business decisions and it's incredibly helpful to be able to go back to that week and say, "Hey, what did the system think was true on March 2nd, and what happens?"

Going back to trying to understand some of the decisions that were made. Some of the payment decisions, the care decisions. We can now go back and understand, "Ah! That's because we have the wrong data. Now we can rerun these events using the validity period rather than the

transaction period. Wonderfully useful. We developed that first just for thinking about new claims engine, which is actually a wonderful thing to talk about but I don't know if we'll have time.

This idea that we're going to need these two different dimensions kept on popping up. Once we started talking about it with the claims, like, "We talk to other teams now, like, "Yeah, that's totally makes sense for a member data and for plans, signs ups, like all these—" Now, we're like, "All right. Let's just make everything in general by temporal." That's kind of this layer of integrated truth that we're trying to where getting really nicely transformed data, really nice approach to time and that data and make sure all their relationships are covered.

That's sort of the goal with the data engineering piece right now is developing that foundation. On top of that foundation, many many things could be built. That's next. You can see some of the predictive learning, statistical learning coming out of all of these relationships. Right now, just building that really solid foundation, getting our operations tight is really the focus of our data engineering work.

[0:44:04.1] JM: Let's start to move towards that conversation. What are the moonshot goals that Oscar could accomplish in the next five or 10 years once this data is closely integrated with customer facing products?

[0:44:18.7] IC: Sure. You could just imagine, say, making an appointment through the Oscar web app with a doctor and then you walk into a doctor's office geo-fencing and so you get automatically signed in. Transfer your personal, your demographic information, fill out all those forms. Automatically, you can look over them, sign them manually, but you'll have to do that work, so yeah, it's still a matter of convenience, and we could make it your electronic health records with your consent available to the doctors so they understand what you're coming in with, your context.

Doctor could — The receptionist, even, before you see the doctor, just a quick check to make sure that you're eligible, nothing new there. That's great. See how you're coming in for a particular reason. We might need to do these tests. Let's see. Are those things going to be covered? We should be able to not only tell you this thing is covered, but we should be able to tell you exactly what your copay is going to be and how this is going to work towards your

deductible. Is your deductible already been met? What's going to be the exact payment situation?

Hopefully you could be able to make a decision before you actually get the service. Exposing those costs upfront very precisely would be really amazing. Then, say, you get the care and you're walking out the office. Why not at that point just have the doctor submit the claim right there and then? If they can do that, we've taken a couple of hops of extra inefficiency out of the system. Also, we could, if we're going to owe the provider, why not just pay them right then? If you want the money in your bank account tomorrow, then submit your claim today.

That whole integrated experience got so much confusion out of the process, so much inefficiency out of the process. You see it's not just then we're providing a convenient case, convenient experience for the member, we're also providing convenient experience for the provider. We want to be mindful of both of those relationships being very important and doing what we can to broker not only those individual relationships with us but the relationship between the member and the provider.

Just as a member experience, that would be kind of a moonshot, the kind of technology that we need to build all that doesn't really exist in this space right now. That's what we're creating. We're expecting some really exciting stuff to come out once we've finally done the work of building all of the basic infrastructure and applications that we need to construct an insurance company.

We've had to outsource certain parts of it just because it's not possible to build a fully running insurance company in just a few months or even a couple of years. We keep on insourcing, in-housing more and more of it as time goes on. We're looking at replacing some pretty major pieces, like our internal claims engine, our billing and payment systems have already been done, our eligibility engine. All these are the stuff — On top of the integrated data platform, the integrated logic platform, to make some of these really beautiful customer experiences become reality. That's what we're shooting for.

[0:47:57.4] JM: That's one thing that's cool about the opportunities in what I sometimes hear being called digital health rather than when people — Because when people think about,

“What’s the intersection of healthcare and technology?” People think about like a new robotics that’s going to do surgery on you. You don’t even need that necessarily, you just need computers doing what they do well and you can get a vastly more efficient system.

[0:48:26.3] IC: That’s right. You can see that happening. Just stepping back from Oscar a little bit. It’s wonderful to see the kind of focus that I’m perceiving now in venture capital and also just what entrepreneurs are really interested in and tackling right now.

[0:48:45.1] JM: Is it getting easier? Is it getting easier for an entrepreneur that wants to build a business in healthcare?

[0:48:50.5] IC: In healthcare? Yeah, it’s still pretty tough. It depends on what you’re talking about. A health insurance company is not going to be easy, period. It’s just really hard, because not only because of how much work you need to do and how much competency you need to acquire from very early on, but the regulatory engagement that you need before getting out the door. The kind of cash that you have to have in the bank to cover thing and figure out in the sideways. It’s a huge moat to cross. We’re the first new insurer in New York in 15, 17 years, something like that. It’s just not something that happens very often because it’s super hard.

[0:49:30.1] JM: Why was Oscar able to do that when nobody else could do it for 15 or 17 years?

[0:49:34.3] IC: We have some pretty awesome cofounders. They’re well connected. They’re smart. They’re experienced. That they were able to pull a lot of these together, and really ambitious. They’re kind of crazy to pull this stuff.

[0:49:48.6] JM: There was no one weird trick to build an insurance company. They just worked really hard.

[0:49:54.9] IC: Yeah. We also benefited from the Affordable Care Act. We have that wonderful configuration of having a promise of a really shake up, or at least the indication that there might be a disruptive legal legislative happening. If we could position a company to take advantage of that, well that gets venture capitalists interested.

Then we had to have a bunch of crazy cofounders to think that — It's like, "Wow! Let's make a health insurance company and let's try to fix something that is ingrained and difficult as the United States Health Insurance or just healthcare industry. To find people who are crazy enough to think that it was a good idea to help them. It's tough and the Affordable Care Act really did get us the investment interests, but then the ambition and the competency of the people who really got this place started. It's something that I find pretty impressive.

[0:50:56.4] JM: Okay. Last question in the remaining five minutes or so we have. Why is it that the United States Healthcare System is so much more expensive and less efficient than the other developed nations like Singapore, or Israel?

[0:51:12.5] IC: I'm going to have to just put my opinion head on here, but I think it's an opinion that's shared by many of us here. I think getting back to the set of relationships that I was talking about just the very beginning, that really is the source of a lot of these expense because we have absolutely the wrong relationships to create a liable market and to put healthy market pressure on the right players and the right ways.

Just getting back to the initial customer relationship with the healthcare, with the health insurer is selling to the employer. The employer is now the source of money. They're the customer. They're your friends. They are the person who's engaging with healthcare as a cost driver. You got to try to minimize the amount of costs that they develop in the old model.

Also, probably even more diabolical, you have the people who are actually engaging with healthcare are completely cut out of the pricing of it. Basically, in America, you just want your health insurance to cover your expenses. You don't have to look at it. Have you ever looked at one of those explanation of benefits letters after you've been to the hospital? They're crazy. You see things like you have to have these incomprehensible services done to you, build 5,000. The insurance company approved $400, paid 300, you owe 100, something like that. It's like, "Well, okay. Cool."

Why this? Never did I get to shop around. It's like, "No. Even if you have the luxury of choosing which hospital you're about to go to, you're not going to be able to easily get the prices upfront."

Now, if you're actually paying, you might start thinking about that a little bit more. Then it's like, "This hospital," hospital A versus hospital B in the old world, it's like, "Well, I have this vague idea that hospital A is kind of higher quality than hospital B," which may or may not be the case, but you don't know. It's completely abstracted away from you. Hospital A is actually four times the cost of hospital B and it's not four times the quality.

You actually have to pay out from your own pocket. You might go to hospital B. That whole decision, that whole market pressure is absolutely cut out of the equation. That's one of the things that we're hoping to really take advantage of those new relationships to help basically put the right kind of market pressures back on and get the actual customer as much back into the conversation in the way that they should be as market participants.

[0:54:03.9] JM: Isaac, I want to thank you for coming on Software Engineering Daily. It's been a great conversation.

[0:54:06.9] IC: Thank you so much for having me. It's been a lot of fun.

[END OF INTERVIEW]

[0:54:12.7] JM: Thanks to Symphono for sponsoring Software Engineering Daily. Symphono is a custom engineering shop where senior engineers tackle big tech challenges while learning from each other. Check it out at symphono.com/sedaily. That's symphono.com/sedaily.

Thanks again to Symphono for being a sponsor of Software Engineering Daily for almost a year now. Your continued support allows us to deliver this content to the listeners on a regular basis.

[END]