EPISODE 1093

[INTRODUCTION]

[00:00:00] JM: As a user browses a webpage, that browser session generates events that need to be recorded, validated, enriched and stored. This data is sometimes called customer data infrastructure or CDI. The data requires a full stack of different tools. System on the frontend to collect the data, middleware to transport the data, and backend systems for storing and loading that data into data warehouses and other analytical systems.

Snowplow analytics is a data collection platform for storing events. In Snowplow, modules called trackers send data to collectors, and this data can be then validated and enriched and then put into a customer's data warehouse via ETL jobs.

Alex Dean is the CEO of Snowplow and he joins the show to talk through the business model, the management and the engineering of Snowplow analytics, as well as the overall data engineering landscape.

If you are interested in sponsoring Software Engineering Daily, you can send me an email, jeff@softwareengineeringdaily.com. The show reaches more than 30,000 engineers daily, and if you are interested in reaching those engineers as well, we'd love to hear from you. You can also become a paid subscriber to the show by going to softwaredaily.com and clicking on subscribe. That would help us support the show and it would mean a lot to us. Thank you.

[SPONSOR MESSAGE]

[00:01:27] JM: SAP is a company that touches $23 trillion of consumer purchases around the world. You've probably heard of SAP, but you may not know about SAP's open source investments, such as SAP Data Intelligence. SAP Data Intelligence connects and transforms data to extract value from the distributed data landscape and embraces the best of open source technology.

SAP Data Intelligence brings together data orchestration, metadata management and powerful data pipelines with advanced machine learning enabling close collaboration between data scientists and the rest of your infrastructure teams. SAP Data Intelligence runs on Kubernetes. SAPs contribution to Kubernetes includes the Gardener Project that helps to operate, monitor, manage and keep Kubernetes clusters alive and up-to-date. To learn more about SAP Data Intelligence, you can visit sap.com/sedaily. That's sap.com/sedaily.

[INTERVIEW]

[00:02:38] JM: Alexander, welcome to the show.

[00:02:40] AD: Great to be here, Jeff. Thanks.

[00:02:42] JM: Snowplow is a tool for data collection and validation and enrichment, and the data that we're talking about here is event data. Explain what these events are.

[00:02:54] AD: Right. Yeah, good question. The way to think of it is you've got these kind of customer-facing practices, like places like websites, mobile apps, email systems, things like that. Your customers, your users are in those systems and they're behaving in certain ways. So they're generating activity.

The activity that they're working through, that's giving off these kind of tiny signals, these tiny little records, these tiny facts that are tracking what they're doing. So a good example is if you're in a website, you're viewing pages. You're staying on a page and we can send out an event that says the user is still on that page and maybe they've scrolled this part down in the page. Those tiny signals, we call those events. You can think of them a little bit like records in a database, but the small facts about that user at that point in time. When we talk about events, Snowplow being kind of like an event pipeline, that's what we mean. We mean it's a pipeline where these kind of tiny little behavioral signals flow through these events or records.

[00:03:55] JM: Does Snowplow itself handle the event collection? Like if I'm building a mobile app and I want to track events, am I creating those events through Snowplow?

[00:04:08] AD: Yeah, exactly. We put a bunch of SDKs, common platforms like web and mobile and server side as well. Snowplow users, Snowplow customers will embed those SDKs in their apps, in their websites and so forth, and those SDKs will basically generate the signal, generate the events. Then the Snowplow pipeline kind of picks those up and then further processes them. Yeah, you can think of it as like you put the Snowplow SDKs in and then that starts to kind of generate the data, the event data.

[00:04:39] JM: What would be an example of when I would need to get that data through the pipeline and do some additional work on the raw event data?

[00:04:49] AD: Yeah. This kind of data is – Well, we call it behavioral data now. 10 years ago, it used to be called click stream data. Now it's called behavioral. And there's just loads of different use cases for it. Businesses will kind of go through various stages and they'll get started with a lot of kind of transactional data and data bit sitting inside SQL databases and things like that. But there comes a point where you come up with a use case that just needs way more granular data, way more kind of granular information on the users, the customer's behavior at that point in time, they're activity.

So good examples would be maybe you're trying to do fraud detection. Maybe you're trying to understand customer behaviors, so deep understanding in the customer journey and they run up to a purchase, trying to understand what's sticky about a product and what's keeping your customers with you or making them more likely to churn.

There's many use cases as there really are kind of industries and departments inside of teams. But what brings Snowplow into an organization is that decision that, yeah, like we can't solve this problem properly without having that super granular kind of behavioral signal.

[00:05:58] JM: So when I create an event let's say with my ecommerce application, let's say a customer logs into an ecommerce application and they start clicking around. They're adding things to their shopping cart. They're seeing ads. What kinds of events might be generated from this workflow and what would happen to those events as they're being pulled into Snowplow?

[00:06:30] AD: Basically, you want to kind of capture all of that signal. You want to capture the customers or the perspective buyer paging through the side, experiencing catalog pages, looking at detail pages. Maybe you're kind of interacting with specific products. How far down the description they're reading? Whether they're clicking through on kind of recommendations to other pages and things like that. You want all that kind of granular information. Once they start getting into a kind of a buying cycle, maybe adding things to cart or adding things to wish list and then going through checkout, you want all of that granular information as well.

Then once you kind of got that, then you can start to use that in super interesting ways. Kind of a classic example in retail is that signal is going to give you great building blocks to figure out things like abandon shopping carts is a great example. There's no signal that says this shopping cart has been abandoned. But if you're collecting all that behavior that the shop is going through and then you see it stopping, you can figure out, you can do some downstream processing and figuring out, "Yeah, this is a perspective shopper that we want to get in touch with," and use that kind of behavioral understanding to then trigger something else downstream.

[00:07:39] JM: Snowplow was founded around 2012. How has the data landscape changed since that time?

[00:07:48] AD: It's changed vastly, really, really kind of to massive extent. Snowplow was founded – Yali, my cofounder and I, we founded it back in 2012. We had kind of the original idea of building this event data pipeline. I think the thing that was a little bit unusual at the time was we built it on top of AWS. We said, "Look, Snowplow is a behavioral data pipeline. You need to be running this on top of AWS. We're going to use AWS services, and that's going to make it super scalable."

I think looking back, it's hard to remember what the landscape was like in 2012, but like cloud was not in this kind of dominant position it was in now. Today, Snowplow supports AWS and GCP, but like it just wasn't such a big thing. It was a bit more of a bet back then. I'd say a few things. I'd say that the rise of cloud is a huge, huge difference. So many data engineers today spend so much of their time working with the native, cloud native data services inside of AWS or Azure or Google Cloud. We've seen so much like innovation on the streaming side of things, the rise of the cloud data warehouses.

Back in 2012, Red Shift was not a thing. And that's not even to start on the whole kind of people and process and methodology side of things. Data engineering was still a pretty niche career back in 2012. Now it's super mainstream and we're seeing kind of follow-on movements, like data ops and data meshes and all sorts of interesting things. Yeah, I would say it's changed vastly. Some things have stayed the same, but there's just been constant change on people and processes and tech.

[00:09:27] JM: The rise of the cloud data warehouse, how did that change your business?

[00:09:34] AD: It changed it a lot. The original version of Snowplow in 2012, we just landed the data in Amazon S3. So we gave you all these kind of granular events in S3 and we said, "Well, you better write some Hive queries." So Apache Hive was big back then and it kind of gave you a SQL layer over Hadoop. He said, "Well, you better write some Hive queries to work with that." That was just super difficult.

We were looking around the different options and trying to figure out could we get the Snowplow data into a data warehouse, and we heard from the guys at AWS that they were working on Redshift, and so we kind of behind-the-scenes started putting Snowplow to work with Redshift. I think our open source support for Redshift came out just a few weeks of the – Redshift came out of private beta. So that was just an awesome time for us, like as Redshift adaption went up, Snowplow adaption went up. People were searching for things like my web analytics data in Redshift and they were finding Snowplow, and Snowplow wasn't just a blog post tutorial. It was like doing Apache 2 licensed code that could do this thing. Yeah, the advent at Redshift was really big for us.

[00:10:44] JM: As Snowplow grew into that cloud data warehouse era, what else did you see in the customer use case patterns? How was the – Obviously, people were collecting more and more events, and I'm sure you had to figure out how to handle those, that [inaudible 00:11:06]. But what other kind of changes did you see as the cloud data warehouses rose to prominence?

[00:11:15] AD: We saw kind of the build out of this kind of modern data stack with the cloud data warehouse is kind of a lynchpin. We saw companies were starting to hire these kind of data teams, build these data teams, bring in maybe some sort of head of data engineering or similar role, and we saw them bringing in the cloud data warehouse and then thinking about, “Well, how am I going to use this data?” So thinking about the next generation of BI tools, people like Looker, and they also started thinking, “Well, how am I going to populate this warehouse?” So they were finding tools to kind of bring data in from their SaaS system. So finding things like Stitch and Fivetran, and then they were finding kind of behavioral data pipelines like us to bring in the kind of the rich activity.

I think the cloud data warehouse has changed a lot around, yeah, just kind of adaption patterns. And I think it kind of made it okay for people to start bringing all these data together and building this kind of central capability.

I think in what we saw, we’d started with web analytics and we pretty rapidly saw people wanting to bring their mobile app data in and then we saw users and then customers wanting to bring server side data in as well. I think people got more confident that these warehouses could be like kind of a central source for all these stuff.

[00:12:36] JM: There is an angle to the Snowplow story where you were looking at Google Analytics, and considering how Google Analytics gives you a vast quantity of data, but is siloed in Google Analytics, and people wanted a more flexible way of accessing that kind of analytics data. Can you describe how the Google Analytics was kind of an inspiration for how Snowplow developed?

[00:13:10] AD: Yeah, definitely. To think back to 2012, like Yali and I, we’d been doing a lot of consulting projects in London for UK clients, and they’ve been able to give us really good offline data. Data coming out of ERP and point of sale and CRM systems and things like that. But when it came to the kind of clickstream data, the behavioral data, they didn’t really have anything. They would give us a login to Google Analytics and say, “Hey, you can log in here and get what you need.” That wasn’t kind of good enough. That wasn’t what we wanted. We wanted access to all the granular, all the events, all the behavioral signal.

I guess what was – And the reason we wanted access to all that data was because we knew if we had it, we could do any analysis we wanted. So we could grab it, grab that data, join it to the offline data. We could do single customer view. We could think about customer loyalty holistically, all of those kinds of things. We knew how to build the kind of the engine that underlie something like Google Analytics or [inaudible 00:14:12] that was then. Kind of the original concept of Snowplow was what if we just sort of take the engine out of something like Google Analytics, give that away for free, open source it and say, “Look, you can kind of build what you want on top of this engine.”

I guess Snowplow, it was interesting because we were kind of in the web analytics category and that we cared deeply about your web behavioral data, but weren’t a web analytics vendor. Funnily enough, when we started Snowplow, we thought we were kind of disrupting. We thought, “Oh! We’re open source. We’re disrupting Google Analytics.” Google Analytics is doing just fine. Lots of customers and users use Snowplow alongside Google Analytics. We did something different, which is we actually kind of built this behavioral data engine, if you will, that data teams could then build up stuff on top of. It ended up being quite different from Google Analytics even though frustration with GA was kind of where we started and why we started to build out.

[00:15:11] JM: You also worked for OpenX for a year and a half a certain point of your career, and OpenX is an adtech company I believe to exchange. And I know that adtech has a lot of opportunities around data management. I’m wanting to go through a little bit of the high level before we dive a little more into the architecture and how you got to that architecture and the engineering behind it. But when you look at the adtech space and the amount of data collection that goes on there, what are the opportunities for building products there and did that at all play a role in what you ended up doing with Snowplow?

[00:15:58] AD: Jeff, you’ve done your research. Yeah. Yali and I met at OpenX. And so we met at an adtech business that had kind of an open source pedigree, heritage. So I think that gave us two things. It gave us a kind of an earlier understanding about open source business models and how you could potentially build a community and then go from a community to commercializing. Then almost more importantly, getting started with Snowplow, it kind of showed us how to build these kinds of pipelines. You’re right, the adtech industry was very early into building click stream data pipelines.

I remember in 2008, like OpenX was starting to build out kind of re-platform, and I was spending time in Pasadena and I got to work with like early original Hadoop engineers. You start to see how these components are going to work and how you're going to be able to build these very kind of high-fidelity pipelines on top of them, and then that kind of turned in – Later, that turned into Amazon putting out Elastic MapReduce and kind of packaging up Hadoop. And that was one of the early thought we had at Snowplow, which was, "Oh! We can build the initial version of Snowplow on top of Elastic MapReduce, and that's going to give us a lot of the kind of the fabric to build Snowplow on top of."

Yeah, adtech was a big help in the early days of Snowplow, and adtech's had its ups and downs since, but adtech and martech have always been catchphrase that have really, really cared about very granular event stream data and working with – My sound just went strange.

Yeah, adtech continued to be like major consumers and major thinkers in the whole kind of the event data space. They used technologies like Snowplow very widely still. Some adtech vendors used Snowplow under the hood, I mean the open source side of Snowplow.

[SPONSOR MESSAGE]

[00:18:00] JM: Airtable is a new way of creating software. It's a low-code platform with the ability to become code-heavy. And through years of development as a modern approach to a spreadsheet, Airtable has now evolved to become a modern approach to a database backend. If you've ever thought of your database as a spreadsheet and wanted to treat it that way, that's how Airtable got started. Of course, Airtable is now much more than that. Air table has introduced blocks, which is a system for building rich application components that sit on top of the Airtable backend. And more recently, custom blocks, which is a system of JavaScript and React components that work through the Airtable SDK to enable developers to make their own functionality on top of Airtable.

You can make a modern, authenticated, real-time cred app for users or admins or for yourself and you can make really anything that you would want to make out of a database, and it really saves you time if you're looking to get started in a low-code fashion. You can enter the Airtable custom blocks hackathon at airtable.devpost.com and you have a chance to win up to $100,000 in cash prizes. Get started by defining your database backend in Airtable and then move up the stack to building a fully-fledged custom block using the Airtable SDK. Win your share of the hundred thousand dollars cash prize at airtable.devpost.com. Try out a new way to build software with Airtable. Thank you to Airtable for being a sponsor of Software Engineering Daily.

[00:19:44] JM: Now, let's get into the architecture of Snowplow. We've talked about the fact that you can collect these events, and these events eventually might want to make their way into your cloud data warehouse so then you can query them, you can build machine learning pipelines around them, things like that. But there is this intermediate step of cleaning up or enrichment or these things that can go on in the Snowplow pipeline. Give me a description of the Snowplow architecture. Maybe take me through like the life of an event.

[00:20:22] AD: The big insight when we were building Snowplow or designing Snowplow and something we've stayed true to ever since is the idea that Snowplow is really a unidirectional pipeline. The data goes in one direction, and as it goes through, it gets enriched, it gets validated, all that kind of stuff. We're constantly kind of augmenting and improving the data, but it's a single directional flow, and that made the whole thing a lot easier to build and it's actually made it a lot easier to evolve overtime, because we can say, "Well, we'll work on this part of it now. We'll improve this part of the pipeline."

The way it flows end-to-end – So we have the event generation in the SDKs, in the trackers that we call them. Those events then get sent into a collector, which is just a basic kind of HTTP server that picks them up. That collector then puts the raw events, so the events that basically just come out of the tracking SDK. Puts them on to a raw queue, so like a Kinesis or Google Cloud pub/sub event stream, or queue, or topic, call it what you will. And then the heavy lifting component which we call kind of the rich component picks those up and it does – Even it's called enrich, it actually does two super important things. The first thing it does is event validation, which I'll come back to in a second. Then the other thing it does is enrichment. So those two things are really, really core. They're really big parts of Snowplow.

The validation is really important. When we send in the events from the trackers, we get the developer to essentially annotate what schemas those events should follow. By schema, I just mean like what's the data structure of those events. What are the entities that are part of that event? What is the definition? What is the structure then? That's a little addressed, if you will, like I'd send in with the events. Then in the validation process, we check those references against the actual data structure definitions. We created a component called Igloo, which is a schema registry. Basically, that's just like a tiny store of these data structures.

We got a schema registry. The guys at Confluent have got a schema registry. It's a pretty important part of this architecture, because you want to very quickly essentially check the events that are coming through against the structure they should have, and that's really important in our space because those events that are coming in, they could be unreliable. They could be somehow corrupted, maybe in the build process, maybe someone – When the developers has changed the fields that are coming through. So we need to kind of [inaudible 00:22:57] check all that. Think of it as kind of an unreliable numerator that we need to crosscheck.

Validation is a really big part. Then enrichment is the other big part. Basically, we had to start building out enrichment really fast really early in Snowplow's history, because we were sending these events data pipeline and analysts were saying, "This isn't rich enough. I need to know the geo location of these events. I need to know kind of like who was the referral. What was the marketing campaign that got someone to land on this page?"

We built out a bunch of enrichments. We added some abilities to kind of create custom enrichments so you can write them in like JavaScript and you can do SQL database lookups and things like that. Those two things together mean that the data coming out of the enrich component is really high quality. It's like been validated and it's richer. So it's got more interesting stuff on it.

We did another thing early on, which is really important, which is we embraced the idea of bad data, or sometimes we call it kind of failed events now. One of the things that frustrated us about the packaged analytics platforms was that they just would sort of sit on bad data. They wouldn't like flag it. They wouldn't quarantine it. They wouldn't do anything with it. That can create all sorts of problems in your reporting.

We had a strong idea of like, “Well, think of it like Unix pipes.” You got your standard in. You got your standard out, but you need your standard err as well. You need a bad bucket where you can put stuff. So anything that fails this whole process goes into a bad stream, if you will, and a stream of failed events. That’s really powerful, because that means you can then go and fix that. You can then look in there and say, “Oh, okay. I get it. There was a problem with that schema or there’s a problem of that enrichment or whatever. I can go and fix that data,” and reflow that data through the pipeline once you fixed it. That was really important.

What comes out of that component is just enriched events. Validated enriched good quality events. Then we have the question of, “Well, what do you with those?” Like I said, the original destination was S3. That was a bit clunky to work with. So then Redshift came out and we built a loader to get the data into Redshift. Over the years, we built other loaders. We wrote a loader for Snowflake, a loader for BigQuery, a loader for Elastic, and they’re used by different people for different reasons. But fundamentally, the heavy lifting there is you got to get this incredibly rich event data into the warehouse in a format that’s kind of usable and convenient for the analyst. So there’s a fair bit of kind of clever logic that happens in those loaders to make the data very kind of analytics ready. But fundamentally, what’s going into the warehouse is the same super granular, enriched, rich data.

The last piece after that for the warehouse is, well, what do you do with that really granular data? So one area that we’re planning on doing more in is kind of helping users and customers to kind of build that very granular event data back up into kind of more meaningful, kind of more sort of business analytics-ready records. A good example of that if you got Snowplow tracking, like video consumption, you’ll probably send out a heartbeat every 10 seconds and say user is still watching the video. And those heartbeats will all end up in your warehouse, but actually an analyst doesn’t really care about the heartbeats. The analyst wants to know user 1, 2, 3 watched the video for 23 minutes. So that whole piece is called data modeling, SQL data modeling.

We’ve done a bit of work in that area. Other people have done work in that area as well. There’s a DBT, which is a very hot kind of data modeling tool. That’s grown up over the last few years. The guys at FishTalent had built that community really well. There’s almost now kind of a job role dedicated to that part of the pipeline called the analytics engineer who’s doing a lot of work with SQL to kind of rebuild up those layers to really understand kind of the customer behavior and make that kind of usable by the business. That what's happens. Then of course you plug in BI tools, things like Looker and Snowplow bindings into Looker, LookML and stuff like that.

It's a multistage thing. I think the thing that makes it kind of not too difficult to reason about and the thing that's allowed us to keep evolving it over 8 years is the fact that it's single-directional. So we've been able to kind of swap out and improve individual components as long as we don't break the kind of protocols, the source and the target end of those components.

[00:27:38] JM: It's kind of like you're trying to preempt the ETL process, because you're basically trying to ingest the data, clean it up on ingestion and then throw it straight into the data warehouse, if I understand it correctly. Rather than having I think a lot of traditional or a lot of alternative data workflows, they'll dump the data straight into a data lake like S3. And then they'll have periodic ETL jobs, and then the ETL jobs do the cleanup and put them into the warehouse. Then maybe you have DBT doing additional cleanup in the warehouse. But it seems like your architecture is more around cleaning things up before it gets into the warehouse. Is that right?

[00:28:23] AD: Yeah. I think that's quite a good way of look at it. Funnily enough, quite a few engineers and data engineers come to Snowplow having kind of outgrown or got frustrated with classic ETL approaches. You're right. It's a good way of looking at it. ETL was kind of a very classic model of saying, "Well, we've got kind of data in certain places and parts of data and we want to bring that into the warehouse and we wanted to extract transform load." Recently, there's been a lot of like ELT and the idea that like you can do the transformation in the warehouse using something like DBT or Matillion. I think that kind of Snowplow's kind of adjacent to that, but different, which is we built – The early days of Snowplow, you kind of think of it as a little bit of an ETL style process. But as we sort of reengineered it and made all the components real-time, you're right, like essentially a lot of the power happens in real-time in these processing components before it hits the warehouse.

So like that actually turns out to be incredibly helpful when you're trying to bring on the real-time use cases. The real-time use cases have been fairly slow coming in general in the data industry. It's always been this is the year of real-time and like it's sent out that every year people have built more and more like SQL jobs inside of data warehouses. But real time use cases are coming online. More people are thinking about kind of operational analytics and forward deployments and how do we turn this into real-time decisioning. In that world, you're right, like kind of ETL is kind of too slow and too backward-looking and too kind of maintenance-oriented, whereas a lot of the architecture of Snowplow kind of gets you that real-time, gets you that very granular high-quality, high-richness event data basically in real-time so you can do those use cases without having to route it through a warehouse and do lots of clean up and further processing. It's sort of packaged it up more, if you will.

[00:30:28] JM: Now, in Snowplow, if I'm collecting events, I might have really high-volume of events coming in. If I've got a user that's scrolling through a page really quickly, it's collecting lots and lots of events, are there any engineering issues with handling a high-frequency of events like that?

[00:30:47] AD: I mean, we have users and customers doing billions of events a day on the lockdown, like some of our customers and users, their volumes have gone crazy. I think it's kind of – There are two different parts to it. There're absolutely volumes, then there's actual kind of spikiness of traffic, and they're kind of two separate things. The absolute volume side of things, the good news is that the core components of Snowplow, the underlying services in AWS and GCP, they do basically like all scale horizontally.

What you tend to find is that there are kind of orders of magnitude inflection points where you need to step back a bit. And we've had to do this over the years and think, "Well, actually, okay, at this volume, this component inside of the cloud provider starts to breakdown. How do we fix that?" But in general, those horizontal scalable services are pretty robust, and we've built somehow on top of them in a robust way. I think the spikiness thing is interesting. If you think about a lot of the kind of cloud components from AWS, for example, they're good at elasticity, but sometimes you have to kind of pre-warm them. Sometimes they're a little bit slow. And you can end up in a world where – You can end up in an interesting world where you're either kind of having to scale ahead of the spike in anticipation, or sometimes people have like overprovisioned these things just to avoid having the – To make sure they're ready for the spike.

We've been doing quite a lot of engineering recently to deal with spikiness on AWS and we've kind of been bringing in kind of an SQS kind of overflow pipe to handle Kinesis being a bit slow to spin up sometimes and things like that. We're always looking at ways to improve. The nice thing is because we're picking up these larger and larger users, everyone kind of else gets those benefits. Yeah, the direction travels only one way. People are sending more and more event volumes through these pipelines.

[00:32:48] JM: The high-quality data is only attainable if you do this schema validation process. Can you tell me more about how the schema definition and the validation works?

[00:33:05] AD: Yeah, definitely. When we're starting Snowplow, we were focused on web analytics data, and that typically have very defined structures. As we started to bring in support from mobile analytics, we realized that every mobile app is different and app developers want to send in very different structures. We thought, "Okay. We want a way of structuring this, making it predictable so that we can load it into the data warehouse, for example, in a structured way." We looked around at different schema technologies. The one we landed on was JSON schema. Pretty much all frontend developers or application developers are very comfortable with JSON. JSON schema is like very powerful description language that allows you to essentially strongly type a JSON object, and it's really flexible and you can do some really cool stuff with it. We were like, "Okay. Well, let's create a registry where we can store these definitions of these JSON schemas, and then we'll come with a kind of an address formats so that the developer who's instrumenting Snowplow can just send in a JSON and send in a little URI essentially that describes what the schema is.

So we did some thinking on like vendoring it and versioning it, and all of that was really, really helpful. I think the nice thing about all these is it means that if an event fails validation, we get very granular error messages on what fails, and that makes it very useful to go back and figure out kind of what's gone wrong in the tagging and the instrumentation?

I think the validation sort of touches on a big topic that kind of the data folk had gone backwards and forwards on over the last 10 years, which is kind of schema versus schemaless, structured versus semi-structured, versus unstructured data. I know there are different schools of thought on it. Like at Snowplow, our kind of basic tenant for us from our perspective was that worked around understanding the data structures has to happen somewhere. So you can bring it right up to the front of the pipeline, like we do, and say, "Well, we got to have an understanding of what we're sending through at the time generation." Well, you can push it all the way down to like the other end of the scale. You're putting on structured JSONs into something like MongoDB and the analysts are figuring it out.

Those two extremes work in different ways at different points in like product lifecycles and different teams and stuff. But fundamentally, what we found at Snowplow is that the kinds of more kind of data mature companies that use Snowplow, they've just gone through the pain of having data analysts and data scientists struggling with data that is un-schemed or semi-structured and they don't understand what properties it has when. So they've kind of said, "Okay. Well, we don't want to have that pain anymore. So we're going to switch and we're going to put more work in to structuring the data and just have that quality flow through the pipeline. Really, data professionals can talk about schemed versus un-schemed all day. But I think the most important thing is to be explicit in the decisions you're making and why.

[00:36:04] JM: What I'd like a little bit of clarification on is why you need the schema validation? Because if I have my data infrastructure, or if I have my frontend infrastructure all defined however I want to, the user-facing events, like when somebody is scrolling through my mobile app, I get to define the schema of those events as they're being created. Then they created. They get infested on my side of the backend. Why do I need to validate them further if I control all of this event infrastructure?

[00:36:45] AD: It's because there could – There's very often drift in the source systems. In the website or the mobile apps, whatever, there'll be a change in like – In Java language, there might be addition of a new field into the [inaudible 00:37:00]. There might be an addition of a new property into a struct or whatever. That's added to data. There can be like changes of IDs from strings to numbers and back again that are just so common and things like that. They don't sound like a lot when I describe them like that. But if you think about the downstream processing, a good example is if you're trying to stream this data into a table in Redshift, and I think the order IDs are numbers. I've got a numeric column of that order ID and then suddenly there are strings, like my whole process breaks down. I can't keep loading. I can't load that row.

Basically, it's because the schemas exist to create this kind of contract, if you will, from right upstream inside the original applications that are generating the behavioral data all the way downstream to the data warehouse. If you're building real-time applications, again, you're going to have like – You're going to be reading these events. You're going to be working with them and you expect them to be a certain way. You expect the order ID to always be the string or always be the number. It's quite scary if you don't have those kinds of controls in that kind of strongly typed system, if you will. It's quite scary how often it breaks down just because, yeah, data quality as user's source are always tricky.

[00:38:25] JM: What happens when there's a failure in the event validation? Like when my schema finds that one of the events that had been created is invalid.

[00:38:39] AD: Right. Basically, that event will be taken offline by the enrichment component I talked about earlier. They'll be taken offline and it will be written out to a separate stream. Think of it as like a separate Kinesis stream or Google Cloud pub/sub topic, and it'll be written out and it'll be like, "Here's the real payload. This is what we tried to process. And here are the failures that we had processing it." Here are like the kind of the – We tried to validate it against this schema, and we couldn't. And here are the failures. So, order ID was meant to be a string and it's a number kind of thing. Kind of every single failed event goes out there. Then different people do different things with them. Sometimes it's just interesting for reporting, people are like, "Okay. I get it. I've got increased, elevated error rates. I'm going to go back and fix the instrumentation." Sometimes those events are like really important for business reporting or some sort of operational data product. So we'll work with the customer to kind of fix those problems and get them back into the pipeline. And an open source user can do the same thing, because you have access to all the failed events.

[SPONSOR MESSAGE]

[00:39:53] JM: GitLab Commit is GitLab's inaugural community event. GitLab is changing how people think about tools and engineering best practices, and GitLab commit in Brooklyn is a place for people to learn about the newest practices in dev ops and how tools and processes come together to improve the software development lifecycle.

GitLab Commit is the official conference for GitLab. It's coming to Brooklyn, New York, September 17th, 2019. If you can make it to Brooklyn on September 17th, mark your calendar for GitLab Commit and go to softwareengineeringdaily.com/commit. You can sign up with code COMMITSED that's, C-O-M-M-I-T-S-E-D and save 30% on conference passes.

If you're working in dev ops and you can make it to New York, it's a great opportunity to take a day away from the office. Your company will probably pay for it, and you get 30% off if you sign up with code COMMITSED.

There are great speakers from Delta Air Lines, Goldman Sachs, Northwestern Mutual, T-Mobile and more. Check it out at softwareengineeringdaily.com/commit and use code COMMITSED.

Thank you to GitLab for being a sponsor.

[INTERVIEW CONTINUED]

[00:41:22] JM: The data, once it's in the pipeline, it can be consumed by multiple places. I could save it in a data lake. I could also save it in a data warehouse. I could use it to maybe trigger something. Can you tell me about the actions that developers might want to take once an event comes in to Snowplow?

[00:41:51] AD: It's a really interesting space and it's kind of evolved with the wider kind of data tooling ecosystem. The first things first, the bread and butter, so many Snowplow users and customers will do the kind of crunching in the warehouse that I described. That is kind of – That's sort of table stakes. You get the granular data into the warehouse and then you do data modeling on top of it and you might do like business reporting or product analytics or whatever. You can do loads of analytical use cases. That's very standard.

But like you said, you've also got these events in stream. You've got a very high-velocity stream of all of these kind of behavioral data and it's sitting inside your own Amazon or Google Cloud account. I mean, different technologies have come about over the last, whatever, 10 years, for doing stream processing. And we have lots of users and customers who plug those in.

The last couple of years, lambda functions have been very hard. So we have various different users and customers who build lambda functions to read that stream and then do something with it. Maybe they're looking for abandoned shopping carts. Maybe they're looking to make content recommendations or something like that in the real-time. There are a lot of cool use cases like that.

We see other people using things like Flink, Spark and so forth. There are lots of different tools out there. One thing we put out to make those use cases a little bit easier is kind of, again, a bunch of SDKs, which we call the analytics SDKs. So they kind of make it easier to work the Snowplow data in one of those environments. We have a few of those.

But yeah, I mean there are loads of different use cases and there are different tools, because fundamentally it's just a stream and pretty much all those stream processing tools support Kinesis and pub/sub. Yeah.

[00:43:42] JM: Do you have any perspective on which data warehouses are the most popular and the kind of breakdown of usage between Snowflake, BigQuery, Redshift?

[00:43:58] AD: People really like – We see it breakdown quite a lot by cloud, and like I said, we're on AWS and Google Cloud. So we kind of see those two the closest. On Google Cloud, people love BigQuery. BigQuery is a bit of a sort of a hero service inside of Google Cloud and a lot of people come to Google Cloud, because they or their developers or data folk want to use BigQuery. BigQuery is super popular. On AWS, I would say that there are still a lot of usage of Redshift and there's a definite kind of interest in using some of the kind of adjacent Redshift services; so Athena, Redshift Spectrum and things like that. But we see a lot of Snowflake usage with our customers and our source users. I think the thing that's Snowflake have really got right is it's just kind of a one-stop shop, like you get comfortable with Snowflake, get comfortable with the UI, the query language of that stuff, SQL, and then people, they like it. So, yeah. And the elasticity side of it I think, separating the storage from the queries, that was very important when they came out with that.

Yeah, we see a lot of Snowflake usage, a lot of BigQuery usage. But Redshift still has a lot of aficionados, still has a lot of fans out there and there's a lot of people who've invested pretty heavily in Redshift over the years. Yeah, it's not going anywhere.

[00:45:22] JM: As GDPR has become more prominent, has that affected your business at all? Because there's so much regulation around data management, and Snowplow is inherently a data management product. Did you have to make any adjustments to the infrastructure?

[00:45:43] AD: We took GDPR and similar things like CCPA. We took them and continued to take them very seriously. Like I think that it's easy to kind of get stuck on the kind of the technical side of those things and think about like specific features. I think the wider thing with those is thinking about what they're really going for, and they're really going through the idea that like all of us who work in data need to get way clearer on the basis under which we're collecting behavioral data from users and customers and other people using these platforms.

We put an extra functionality into Snowplow around kind of tracking consent from users. So we put that into the tracking SDKs. We did more thinking around like PII and around masking and things like that and we continued to take it super seriously. I think that Snowplow is interesting because the Snowplow pipeline is out there and it's open source, Apache 2 licensed. People can take it. Like I mentioned earlier, we have users in adtech and we can't control how Snowplow was used. But what we can do as a company is we can think really hard about how we want Snowplow to be used and we can think really hard about like the privacy-conscious persona. So like the data team that wants to do this stuff right and do right by the data subject so that that company is, yeah, collecting data from.

Yeah, we're thinking about a lot. We're thinking about what else we can put into the roadmap around it. One thing I should really point out is the way that Snowplow runs, it's open source. As an open source user, you take and run it in your cloud. Our paying customers, it's the same model. We run Snowplow in a private SaaS way, which means we run Snowplow for our customers in their own cloud accounts. Our customers are fundamentally running this as first party data collection. Almost all our customers, most open source users will have the Snowplow data going to their own subdomain. That's how this stuff is set up.

I think that we're hopefully on the right side of where all these is going, which is companies, if they think hard, have reasons to collect certain behavioral data from their customers, and Snowplow as a software platform can help and do that and those companies, less and less, do they want to be sending all these data out to a third party SaaS vendor and trusting that vendor to do certain things with it or not. I think, the industry is moving in a direction we're happy with, but there's always more we can be doing to think about kind of how we can kind of help users and customers look up to their data subjects.

[00:48:22] JM: We have done some coverage of companies like Segment. Segment is another data collection, data analytics, data infrastructure product. How do you see yourself positioned relative to Segment or to other – Anyway, there's a number of these customer data infrastructure products. How do you see yourself positioned relative to those?

[00:48:44] AD: Yeah, it's a great question. We being kind of stable mates of Segment for a long time. They got started around the same time as we did, and of course we've tracked them overtime and they've got some great stuff. I would say that there's a kind of a fundamental difference of focus. With Snowplow, like we've been creating for 8 years this kind of behavioral data engine, and it's open source core, and when we sell it to customers, we sell it in this private SaaS way. Really, it's an engine that our users and our customers build on top of, and they built very diverse things on top of it. They need a lot of different use cases.

I would say that Segment have been a little bit more focused on creating a very convenient end-to-end pipeline and meeting some very valuable needs around, for example, relaying and multiplexing that event data into lots of different marketing endpoints and different SaaS tools and things like that. That's not really an area we've gone into.

For example, we only, as a behavior related pipeline or engine, we only focus on behavioral data and we don't pull in data from SaaS systems like Segment does. We tell our users and customers to use something like Stitch or Fivetran for that, and we don't plan the Snowplow data out into a bunch of different SaaS tools. For sure, we have users and customers who do that who write connectors to get the data into different SaaS destinations, but that's not really been the heart of Snowplow. We've had loads of different use cases that different people use.

I guess on the kind of data quality enrichment stuff, that's been probably a big focus for us than Segment, but only they've done some interesting work recently on data structures and things like that as well. It's an interesting one. I think the other thing is they do a lot of work as a CDP. So they do a lot of work talking to marketers about how to get this data into their hands.

We've always been much more focused on the data team. So we've been much more focused on the data engineers, data ops, the business analysts, the analytical engineers. That's our home game. That's who we focus on and empowering them. It's a little bit of a difference of focus. Yeah, a lot of the same ideas running through the two products.

[00:51:01] JM: All right. Could you tell me a little a bit about how the company is managed? What's the structure of the different teams, and give me some organizational principles for how Snowplow analytics is organized.

[00:51:14] AD: Snowplow is kind of hybrid setup. We have about 70 people. About half the team have a desk in London. Almost all of which haven't seen that desk in a while on the count of lockdown, and the rest is globally distributed. We built out – Snowplow is interesting. We built out engineering and support for very early on, because that was kind of where the business started. First, it was Yali and me, and we're doing everything. Then as we started to figure out how to commercialize Snowplow and bring on the first customers into that kind of very SaaS model, we were able to hire. So we just really wanted build our engineering. We wanted to build that support so that our initial customers had a great kind of experience with us.

We've built out these teams early. We've built out these teams in a kind of a globally distributed fashion, because we were looking for talent. And on the support side, we were looking for people in different time zones to make sure that we have [inaudible 00:52:13] support. So we built it. Then of course because we'd sell it in open source, we had like open source contributors who are pinging us pull requests on GitHub and stuff like that. We had a lot of like remote DNA, distributed team DNA early on.

Then as we kind of got bigger, we started to build out some core functions in London. We started to build out a bit of an operations team. We started to build out recently sales and marketing as well, of course, and so forth. It's been really interesting. I mean, we've been navigating the whole COVID-19 thing okay as a distributed team. I think that one of the things that really brought it home [inaudible 00:52:54] for us is we in a way week twice a year. So we kind of get everyone together from all over the world and get to be together for a week normally in a city in Europe. We were meant to have one of those three weeks ago, and we couldn't have it because of lockdowns and quarantines. That was a real kind of like oh moment where we kind of had to go really remote, and we did in a way we did it remotely. We were together, but a part, and it was great. Yeah, that was really missed.

[00:53:24] JM: I just like to close off with a little bit of discussion of the future of data engineering. Do you have any predictions of what's in the near future for data engineering or the future for your business in particular?

[00:53:39] AD: I think data is only going in one direction. I think everyone's talking a lot about digital transformation, and that's a big part of this, what's happening at the moment. But data is really big as well. People are trying to make decisions faster. They're trying to make them in a more data-literate way. I think the macro trends are all going in the direction of data teams, data professionals, more in investment in data as a strategic asset. That's for sure.

I think that the data landscape, the technologies will continue to evolve. I feel like we're probably overdue a bit of a breakthrough somewhere in the data stack. I feel like there's got to be a bit of a breakthrough somewhere in the storage or streaming landscape. I feel like a lot of the tools have been the same for the last 5 years, and normally in tech that's a sign that something is going to change pretty quickly. I think that's going to happen. I want Snowplow to be ready for that when it happens.

I think we're finally going to see more of those like forward deployed use cases. I think people have got to be comfortable with the kind of backward-looking analytics. I think people want to move into the forward deployments into the operational analytics into real-time decisioning and things like that when people are ready for that. I think it's just going to be more use cases, more data. Yeah, I'm very excited about Snowplow kind of staying abreast of that and just helping these data teams kind of keep pushing forward on their journeys.

[00:55:01] JM: Okay. Alexander, thanks for coming on the show. It's been great talking.

[00:55:04] AD: Thank you so much, Jeff. It's been awesome being on the show, being your guest. Thank you so much.

[INTERVIEW]

[00:55:18] JM: Over the last few months, I've started hearing about Retool. Every business needs internal tools, but if we're being honest, I don't know of many engineers who really enjoy building internal tools. It can be hard to get engineering resources to build back-office applications and it's definitely hard to get engineers excited about maintaining those back-office applications. Companies like a Doordash, and Brex, and Amazon use Retool to build custom internal tools faster.

The idea is that internal tools mostly look the same. They're made out of tables, and dropdowns, and buttons, and text inputs. Retool gives you a drag-and-drop interface so engineers can build these internal UIs in hours, not days, and they can spend more time building features that customers will see. Retool connects to any database and API. For example, if you are pulling data from Postgres, you just write a SQL query. You drag a table on to the canvas.

If you want to try out Retool, you can go to retool.com/sedaily. That's R-E-T-O-O-L.com/sedaily, and you can even host Retool on-premise if you want to keep it ultra-secure. I've heard a lot of good things about Retool from engineers who I respect. So check it out at retool.com/sedaily.

[END]