EPISODE 1259

[INTRODUCTION]

[00:00:00] JM: Apache SPARC is a popular open source analytics engine for large scale data processing. Applications can be written in Java, Scala, Python, R and SQL. These applications have flexible options to run on like Kubernetes, or in the cloud, or using Databricks. The company Data Mechanics is a cloud native SPARC platform for data engineers. It runs continuously optimized Apache SPARC workloads on a managed Kubernetes cluster within the user's cloud account. They boast a 50% to 70% cost reduction from cloud providers by dynamically scaling applications based on load and automatically tuning app configurations based on the historical SPARC pipeline runs. Their Kubernetes clusters are deployed within user accounts. So user data never leaves the environment and they handle the cluster management.

In this episode, we talked Jean-Yves Stefan, co-founder and CEO at Data Mechanics. Jean-Yves previously worked as a software engineer then a tech-lead manager at Databricks. We discussed the big data engineering in SPARC and the unique advantages of using data mechanics to make SPARC development easier and more cost effective.

A few announcements before we get started. One, if you like Clubhouse, subscribe to the Club for Software Daily on Clubhouse. It's just Software Daily. And we'll be doing some interesting Clubhouse sessions within the next few weeks. And two, if you're looking for a job, we are hiring a variety of roles. We're looking for a social media manager. We're looking for a graphic designer. And we're looking for writers. If you are interested in contributing content to Software Engineering Daily, or even if you're a podcaster, and you're curious about how to get involved, we are looking for people with interesting backgrounds who can contribute to Software Engineering Daily. Again, mostly we're looking for social media help and design help. But if you're a writer or a podcaster, we'd also love to hear from you. You can send me an email with your resume, jeff@softwareengineeringdaily.com. That's jeff@softwareengineeringdaily.com.

[INTERVIEW]

[00:02:07] JM: Jean-Yves, welcome to the show.

[00:02:09] JYS: Thank you so much, Jeffrey. It's great to be here.

[00:02:10] JM: Yeah, it's great to have you. We've done numerous shows about SPARC. And I'd like to start off by just talking about the applications of SPARC in 2021, especially in light of other perhaps competing environments, such as a data warehouse infrastructure. Tell me about the applications of SPARC today, and how it fits into the overall ecosystem?

[00:02:39] JYS: Yeah, that's a great question. So I think SPARC has a central place as a general framework for big data, distributed computing. Main use cases are the engineering, ETL, also SPARC streaming. And then also as a general purpose tool for advanced analytics and data science and for business intelligence. So yeah, these are the most common pillars. In terms of volume, definitely, the data engineering workloads, like ETL is still – Batch ETL is still number one use case. And in terms of how it evolves versus new technologies or data warehouses? Well, using SPARC in the cloud over an object store using, which we call a data lake, is still very popular. I think, if you need the flexibility of a programming language, since you can use Python and Scala with SPARC, it's definitely still the go-to-tool.

And the fact that's – And maybe we'll talk about this in the future, but the fact that now you can deploy SPARC on top of Kubernetes, and that's some pain points that we used to have are going away. Also make it a pretty standard architecture for big data workloads who wants to work with data sets over 100 gigabytes.

[00:04:01] JM: Can you draw more of a distinction between the applications of SPARC and the applications of a data warehouse?

[00:04:09] JYS: Yeah, so data warehouses, typically, you would interact with them only with SQL, while with SPARC you can use both Python and Scala, also R. So when you want to insert maybe your own business logic in a pretty complex way, then SPARC definitely gives you additional flexibility. Also, when you start managing a lot of ETLs and you don't want to go pure SQL because you want to have – At some point, SQL queries become hard to manage, while

SPARC lets you create functions and modularize your code. So that's one reason why people like SPARC over data warehouses.

In terms of cost, you can also be more effective. You don't want to store your entire data in a data warehouse necessarily. For the raw data, you would typically put it in an object store like S3, GCS, Azure Data Lake, and then queried with SPARC. And then maybe the data warehouse, you're going to still use it, but more for low-latency business Intelligence use cases. And so you're not going to put your entire data in there, but only maybe a subset of pre-aggregated data. So that's what we see a lot when people use both, a data warehouse and SPARC, is that SPARC will be used for really the data lake on an object store, and then it would only load a subset of their data onto their data warehouse.

[00:05:35] JM: So are you saying that there are applications where both SPARC and a data warehouse makes sense?

[00:05:42] JYS: Yes, definitely. We see this a lot. SPARC plus Snowflake, or SPARC plus BigQuery. Typically, as I said, when people have both technologies, it’s because they use SPARC to manage their object stores and to ingest data into maybe pre-aggregated. And then they would only load a subset of their data onto their data warehouse, which is then used maybe by analysts, maybe by business users, maybe by dashboards, and so on. Typically, when you're looking for a low-latency and zero management, but SPARC is still the tool of choice for general purpose data engineering in these architectures.

[00:06:25] JM: You work at Data Mechanics, and you build a SPARC platform, a hosted SPARC platform. And the really obvious question that I'm sure plenty of people are going to want to know the answer to is Databricks already exists. Databricks is widely known as the SPARC company. Why is there room for another company based around Apache SPARC?

[00:06:48] JYS: Yeah, that's a great question. And prior to Data Mechanics, I was a software engineer at Databricks. So obviously, I know the differences between the two platforms. Also, by the way, the SPARC ecosystem is even bigger. On Amazon, you have EMR. On Google, you have Dataproc. On Azure, you have HDInsights, Synapse.

In general, I would say Databricks is great if you have a data science heavy use cases. They're not just selling you a SPARC infrastructure. They're selling you a development environment with hosted notebooks, hosted job scheduler, etc. On our side of Data Mechanics, we focus on data engineering workloads. We focus on making it simple to manage big data pipelines and making them easy to develop, stable and cost effective.

Slightly different value prop, our average users, I would say are a bit more technical than the average users of Databricks. But yeah, we're basically the SPARC backend for these companies who are not really interested in buying the Databricks environment, but who just want to have a serverless experience with SPARC. Also, we deploy SPARC on top of Kubernetes, which Databricks doesn't do. And that's kind of the big technological change that allowed us to start doing this even though we were already competing with these big players.

[00:08:11] JM: So tell me a little bit more about the ideation for thinking of Data Mechanics when you were at Databricks and you hadn't started Data Mechanics yet?

[00:08:20] JYS: Yes. So when I was a Databricks, I was a software engineer, I was leading their SPARC infrastructure team. At the time I so the SPARC and Kubernetes project come up. It was a push by some very large companies, but not particularly with Databricks. I always had this personal plan that I wanted to build something from scratch. Originally, I wasn't going to build a competitor. I was more building a complimentary tool that would automatically tune the infrastructure parameters and the SPARC configurations, because that's a big pain point that we saw within Databricks customers and, in general, in the community.

But then – So with this idea, we went to Y Combinator. And gradually, during the program, we realized that if we really want to solve these pain points around SPARC, we actually need to control the infrastructure to make these automated decisions for our users. And so that's how we actually ended up being a big data platform and alternative to Databricks. So yeah, it was a two-step process. I didn't really quit Databricks and saying, “Well, I’m building a competitor.” We gradually got there.

[00:09:29] JM: Knowing what you know about the Databricks ecosystem versus the overall addressable market for SPARC, how do you see your company diverging from Databricks?

[00:09:42] JYS: Yeah. So we really focus on companies who have primarily a data engineering use case. So people who have to build and manage large ETL pipelines that process large volumes of data and they're pretty technical. They're maybe comfortable calling an API or they're comfortable using Docker. But what they're not an expert at is managing SPARC. And so that's what we provide. It's definitely a sub-segment of the market. And we're obviously still very tiny compared to Databricks. But that's the only way as a startup that you can compete with these big players is to have focus and to give it 10x better experience for one sub-segment. So that's data engineers.

We have customers in the US and in Europe. They're mostly startups, mid-market companies. We don't really target enterprise yet, but we do have a couple of enterprise customers. And yeah, we're having fun. I think we're filling a gap. A lot of people didn't want to use Databricks. They were not interested in the development environment that they sell. And so they were going to Amazon EMR or Google Dataproc. And then these platforms are pretty painful to manage. And so I think there's a space for our current positioning.

[00:11:04] JM: So maybe like you're offering SPARC without the bells and whistles of the Databricks platform.

[00:11:12] JYS: Yep, without the bells and whistles. And then on the particular pain point that we solve, we go much further. So on our platform, when you start running a SPARC pipeline on a schedule, say every day, or every week, or every hour, our platform is going to – Whenever your pipeline runs, it's going to process some logs to automatically figure out, "Oh, is there a memory issue? Or is there an infrastructure problem? Or do we suffer from a lack of parallelism?" and so on. And then automatically on the next run of the pipeline, we're going to turn some SPARC configurations for you, like the size of the containers, like some SPARC configs around parallelism or on shuffle. And this has a big impact on stability, on performance. And that's something Databricks doesn't have.

Similarly, we deploy SPARC on top of Kubernetes. And so our users, they can control the Docker image. And it's a lot more flexible than a Databricks platform. So it's just a DataBricks light. It's a different positioning. And the problems that we solve, we do go further. But in

general, it is not as broad or a value prop as Databricks in the sense that we don't have hosted notebooks. So we don't have a hosted job scheduler or an ML flow integration. We really focus on providing a serverless SPARC infrastructure and making it easy to use and cost effective.

[00:12:36] JM: That point on the cost effectiveness. In the average SPARC deployment, in the average SPARC application, do you see opportunities for big cost savings?

[00:12:46] JYS: Yes, and this was actually the starter point of my project when I was at Databricks. I was pretty horrified to realize that on average, people's infrastructure are not used – SPARC infrastructure are not used effectively enough. That can be because people provisioning fixed size clusters, or because people put too many resources. Whenever something fails, they're like, “Oh, let's give bigger instances and more memory, because maybe they don't know how to actually figure out the root cause.” And so that's something people try out.

But on average, when we looked at our customers, back then at Databricks, we realized that 80% of the total compute infrastructure is actually idle is not used by SPARC. So 80% of your bill is for servers that are just idle. And so this is something we wanted to solve. And so the way we solved it is by having this automation, the auto scaling, the auto tuning, and that's how we can typically reduce our customers cost by 50% to 75%. So divided by two, to divided by four. So yeah, on average, there's really a lot of cost savings when people use SPARC.

[00:14:00] JM: So when people are coming to Data Mechanics, presumably they're already using SPARC. They've got their SPARC application built. Are they just managing it themselves? Or they're using Amazon, typically, and then they migrate to you. What's the typical path to a customer coming to you?

[00:14:24] JYS: Yeah. So customers would say maybe two thirds of them already use SPARC on some infrastructure. And then there's a last third that are just starting from scratch. Our platform is only available on Amazon, GCP or Azure. So they must be in the cloud. We cannot deploy on an on-premise infrastructure. So let's say they're migrating from EMR on Amazon. Well, then, typically, we will do a two to three-week proof of concept where we deploy the platform in their Amazon account. We help them migrate a couple of applications. We can make

this pretty simple because it's in their cloud account. The same data access rules work. It's still SPARC, obviously. So the same code works. Our platform will automatically tune some configurations to kind of adapt from running on Yarn to running on Kubernetes. And yeah, in two to three weeks, they can go already from zero to having a couple of pipelines or more running in production. And then if they were really heavy users of EMR, there might be a bit of a migration. But once you've migrated a first few apps, then it's pretty easy to extend it to all the others. And then if people build from scratch, there isn't even the problem of the migration. And it's more about helping people get familiar with SPARC. We have some monitoring interfaces that make getting started with SPARC easy. And it's a different work. But yeah, we work with both kinds of customers, whether they're already using SPARC or whether they're new to SPARC.

[00:16:01] JM: And for your infrastructure, you're built entirely on Kubernetes?

[00:16:05] JYS: Yes. Indeed. And this is pretty new. Until 2018, you couldn't deploy SPARC on top of Kubernetes. And since SPARC released 2.3, there was initial support, but it wasn't very stable. Now, we were at a version of SPARC 3.1, and SPARC on Kubernetes is officially generally available and production-ready. So when we deploy on top of Kubernetes, indeed, that's – The biggest difference compared to Yarn or compared to Databricks is that you're going to have a single Kubernetes cluster for all your SPARC applications, whether they're ETL, whether they're notebooks, whether they're business intelligence queries, whatever. And then, inside this communities cluster, you can run many applications with different SPARC versions, with different version of Python, Hadoop, Scala, all your dependencies. And that's because, as a user, you control the Docker image. And you can build your own Docker image with your own dependencies. And then that's the level of isolation, the container, and not the underlying infrastructure.

So what this means is that you stop having to think about clusters. You just submit Dockerized application, and then it's our job to make sure the Kubernetes cluster automatically scales to make sure we bin pack your containers onto the nodes efficiently to increase your infrastructure and efficiency. And so it decreases your costs.

[00:17:32] JM: What have been some of the most difficult engineering challenges of building Data Mechanics thus far?

[00:17:38] JYS: Well, I would say, getting SPARC on Kubernetes to work, to be stable, to be performant. Yeah, it's not a production-ready. But still, there were a lot of things to patch. There were a lot of gotchas to make it really easy to use and to actually get it to production with many customers. So for example, we use the SPARC or Kubernetes operator. You have to pay attention between the SPARC Kubernetes operator and the Kubernetes version. And so make a lot of tests to make sure they work well. We wanted to make it easy to use some spot nodes on the platform. And so that required a lot of integration between SPARC and between the cloud provider and Kubernetes.

Another example is the performance of shuffle. To make shuffle performance in SPARC, you need to mount certain type of disks that have very good throughput. And again, the way to do this on Kubernetes is slightly different than when you do it on Yarn. And so all of these things, we kind of had to figure them out. And we actually wrote a lot of blog posts to also help people who just want to do it open source. So I would say that was the biggest challenge.

And then maybe the second biggest challenge is just that our platform is deployed inside our customer's cloud accounts. So that means you don't just manage your own infrastructure, but you're managing many customer's data planes. And so to be able to do this, to be able to automatically push updates to our services to be able to monitor all these data planes, that's a lot of work. And definitely, that's the other big challenge of building these kinds of platforms.

[00:19:23] JM: Let's zoom out for a second and talk about the overall data infrastructure landscape. The thing that I found most interesting about the Databricks pitch in recent memory is the whole data lake house idea that you would have a unified data lake and data warehouse basically negotiated by SPARC. To what extent do you see that as desirable? And is that something that you want to enable for customers at Data Mechanics?

[00:19:53] JYS: Yes. So today, already, what we enable is the use of the Delta Lake, which is basically an abstraction on top of S3, which brings a lot of new features, and also some performance improvements. And I think that's a great addition to SPARC. With respect to the lake house, basically combining using SPARC both as an ETL tool over S3, and then as a business intelligence tool as if you were querying a data warehouse, is also something I'm

pretty excited about. I think in our platform, we still have a few features in mind to really make this vision a reality. But I agree, that is a common need. And SPARC and SPARC SQL are built to address these challenges. So I'm pretty excited about these developments too.

[00:20:45] JM: What are the other major projects in the open source data infrastructure landscape that you see as important or evolving?

[00:20:55] JYS: So first, if I really talk about the kind of open source project that we work with on a regular basis, there is the SPARC on Kubernetes operator, there is Delight, which is a monitoring tool for Apache SPARC that actually we open sourced. If we want to zoom out and talk beside SPARC, I mean, in the data engineering landscape, I'm pretty interested in Dask. I'm pretty interested in that Ray. I'm pretty interested in Presto. Definitely, it's a fast moving market. And yeah, it's an exciting space to be in.

[00:21:36] JM: How much have you looked at Ray? Have you seen many applications that are using it seriously?

[00:21:41] JYS: Well, not really. I mean, mostly, I looked at it out of personal interest. Our customers, they mostly use SPARC for data engineering workloads and less on the data science machine learning, deep learning side. So I don't have natural exposure to it. But yeah, I think they're gaining steam. And probably we're going to be hearing more and more about Ray in the future years.

[00:22:04] JM: So when I spin up a SPARC cluster on Data Mechanics, walk me through what happens. What are you using under the hood in terms of cloud infrastructure?

[00:22:16] JYS: Right. So when a customer wants to get started with Data Mechanics, they give us a set of permission, like an IM role on their cloud accounts. So let's say they're an Amazon customer. So we're going to create and manage an EKS cluster for them in their cloud account. They don't need to have any expertise with Kubernetes. They just give us this role. We create the EKS cluster. And then they can interact with it just through our WebUI, or through our API, or through our Airflow Connector, or Jupiter Connector, and so on.

So once this Kubernetes cluster is deployed, then they can start SPARC applications. They can either, as I said, connect a Jupyter Notebook or kick it from Airflow, or from an API call, and so on. And when you do that, under the hood, you make a request to what we call the gateway, which is our service running inside the customer's Kubernetes cluster. And the gateway will intercept the request. It will enrich it with some auto tune in the configurations with some additional configurations for log collection and so on. And then it will actually start the containers, so pods on the Kubernetes cluster. And then you can track this through the monitoring UI, the web dashboard, which is again served by the gateway.

So yeah, for the end user, there's a lot of magic that happens behind the scene, but they can just start submitting Dockerized SPARC applications, track it through our dashboard. And that's it. And under the hood, yeah, what we do is we actually launch the pods, configure them, collect their logs, persist their logs so that they're visible also over time. And then we have this optimization algorithm that is going to analyze their logs and automatically adapt some configurations for the next time their pipeline runs.

[00:24:11] JM: And I don't know if you can share this, but how does that compare to how you ran SPARC at scale at Databricks?

[00:24:18] JYS: Yeah, there are multiple levels to answer this question. On the infrastructure side, I think I can share at least what is publicly known about Databricks. Databricks actually will start virtual machines in their customer's cloud accounts. So let's say you are, again, on Amazon. So when you start a cluster on Databricks, they will provision EC2 instance. And then they will actually run your application on a container. It's not using Kubernetes under the hood. Really, there's a cluster manager that was built by Databricks appropriate to re-cluster manager.

So what are the differences? One is, on Databricks, you need to manage clusters. You need to create a lot of them. A single cluster will have a certain SPARC version, will have a certain set of dependencies. So there will be a bit of a tradeoff between having shared long-running clusters or having ephemeral clusters where you just run your code and then shut them down. On the Kubernetes side, you don't need to worry about that. The underlying nodes are automatically shared by Kubernetes. And then each application is kind of ephemeral. It has a Docker image. It runs in its Docker containers. And then when it's done, the containers are gone, and the space

can be used by other applications. The speed is also higher. On average, we get applications to start in a minute, a minute and a half. If for some reason, you already had capacity, and it's just about starting a container, then it can be no less than 10 seconds. So compared to Databricks, or just in general, compared to running SPARC on Yarn, like on EMR, it's a much better developer experience.

[00:26:02] JM: What are you focused on today inside the company?

[00:26:05] JYS: Yeah. So I'm the co-founder and the CEO, which is ephemeral. I get to wear many hats. So I work with customers a lot from their first call, and tracking their progress, making sure they're successful. So I manage some solutions architect who actually do the technical work of onboarding them and helping them migrate some of their pipelines and so on. Obviously, I also need to get involved with product, thinking about the roadmap, what we want to see in the future. And then one last piece of my role is also content marketing. So if you check our blog, we have lots of resources about getting started with SPARC on Kubernetes about the benefits, the pros and cons of running SPARC on Kubernetes. And, really, I think, I mean, hopefully, articles that are valuable to the community. And that's how people get to know about us. And that's how we meet our prospects and customers. And yeah, it's a roll that changes six months ago. I was not doing the same job, and probably six months in the future, it'll be different again. But that's just early stage startup environment. That's also true for our team. So yeah.

[00:27:21] JM: Tell me a little bit more about building the optimization engine for making SPARC applications more performant.

[00:27:31] JYS: Yeah. So the way we tackle this challenge is we didn't want to build like a black box machine learning algorithm that just tries something at random and see if it works. We cannot afford to do this, because we need our recommendations to work and be stable. We cannot afford to make a bad recommendation that would crash the customer's pipeline. So instead, the way we're doing it is we really do these automated recommendations based on data engineering heuristics, things like, "Oh, you should have at least 3X the number of partitions as the number of cores in your cluster for you to get a good level of parallelism," these kinds of rules. And then there is an optimization algorithm, Bayesian algorithm, that will look at

some optimization objective, like maybe reducing your costs within a certain SLA. And then we'll decide how to tune certain configurations. So we don't do hundreds of configurations. It's more a list of about 5 to 10. Things like the size of your containers. Things like how many SPARC executors do you have if you have a fixed number. Things like the default number of partitions. A couple of configurations around making data access fast using the right committer, using the right S3 configurations. But yeah, this is how it works. And so again, when you run an application for the first time, we don't have any prior knowledge. So you'll just get a default set of configuration. And obviously, you can specify your own configs. But then, once we see your application run on a schedule, we're going to learn from the previous runs and automatically figure out, “Oh, we should be allocating more memory. Otherwise, we're going to have a stability problem and so on.”

[00:29:24] JM: Is there more room for improvement in the optimization engine?

[00:29:27] JYS: Yes, definitely. I think already with the set of rules we have, we avoid a lot of the common mistakes people do. And they may not realize, but they have a very bad configuration that is just global share folders apps and that isn't performing well. And I think we address a lot of that. But there are still some things we're working on. Right now, we're working on automated remediation for some errors. So let's say you have a pipeline that runs into a memory problem and crashes in the middle of the night. But then you have your Airflow scheduler that is going to retry the application. And then on the second attempt, automatically, there's a new configuration where maybe you have more driver memory and then you're going to avoid this failure. So this is something where we're pushing right now, and I think we’ll make our customers sleep without getting paged.

And then besides the optimizations, there is also a class of features that we're building to give insights to our users, because not everything can be fixed through configuration. If there's a problem with the way the data is partitioned, we as a platform cannot take stat. But we want to build a new monitoring interface that also gives you high-level feedback and shows you, “Oh, you shouldn't be joining on this key because the data is very skewed.” And, “Oh, the way you partition the data isn't very efficient. “And so that's another class of features that we’re working on to help our customers make their SPARC code more performant, more stable, and make better decisions overall.

[00:31:04] JM: Do you think there is a competition between SPARC-based application development and data warehouse-based application development? Like is it mutually exclusive to be using SPARC versus Snowflake? Or do you think these are technologies that are used in tandem?

[00:31:26] JYS: Yeah. In this space, for many technologies, at the same time, some people use them in tandem, but at the same time, people hesitate and sometimes need to make a choice between two technologies. So there is definitely some competition. I think if you look at Databricks' recent announcements around Lake House and around their new pricing for analytics queries, they're definitely trying to be competitive to Snowflake. So in general, they are in competition. This being said, yeah, we have many customers who use SPARC and Snowflake. And so typically, when they do is they use the object store as their main data lake. And then they run some aggregations. They de-duplicate their data, etc. And then they only load a subset of their data onto Snowflake. And I think that can be a good compromises if you really need snowflake for some of your business intelligence or for powering dashboards or some real-time applications, and then still use SPARC for really the heavy lifting ETL. And I think that makes it also kind of cost-effective. So they can also work in tandem.

[00:32:39] JM: I'd like to come back to this question of growth and company development, building a data infrastructure company in 2021. I mean, there's never been more demand for data infrastructure, of course, but there's also never been more competition and more noise in the marketplace. Tell me a little bit more about how you plan to navigate the competitive dynamics of the market.

[00:33:05] JYS: Yeah, I think at any time, building startups is hard. And you need to find the right target, the right messaging, the right positioning, and then iterate on the product, and then gradually expand from your initial niche to become one of the big player and address the entire market. So on our side, as I said before, the sub-segment we target are pretty technical data engineers, and they're looking for a serverless SPARC backend that is going to make their SPARC pipelines efficient and stable without spending a lot of time managing them. And then I think as we grow, already, today, we're seeing a lot of demands for some features that are maybe features you would build more for enterprise customers when you have multiple teams

and you want collaboration and you want to fine-grained access control and so on. So we're building these features to start working with bigger customers. And I think, also, we're going to expand in terms of the user persona. Right now, it's really mostly for data engineers, though, obviously, anyone can use it, but we're also going to later make it – Also target the data scientists, then data analysts and so on. It's a fine equilibrium to figure out which features to prioritize. But overall, there is a big demand for running SPARC on Kubernetes. There are a lot of companies who are not very happy either with Amazon EMR or with Databricks. And we help them migrate and solve their pain points and reduce their costs. And so yeah, so far, it's an exciting start and pretty excited for the next 12 months and what it will mean both in terms of product and in terms of growing our team.

[00:34:55] JM: Do you take any lessons away from the Hadoop wars where you had MapR, and Hortonworks, and Cloudera all competing for the Hadoop market? Do you think that's a similar environment that you're competing in for the SPARC market?

[00:35:15] JYS: Well, I don't know if – Let's see. I think these three companies, they were primarily deploying on-premise. And I think the main reason why today we don't deploy on-premise is because it's, it's harder to operate. It's less standard. So building a product takes more time. Pushing updates takes more time. So it would hurt our velocity. You need to have more human power to manage on-premise infrastructure. And so these companies also have a lower margin. So I don't think it's necessarily the high competition between these companies that put them where they are now. Just the fact that on the cloud and the cloud native side, we've been able to progress a lot, lot faster. And so yeah, today, there aren't also that many players. There's obviously Databricks. And then each cloud provider has their own service. But people, more and more, they're looking for cloud-agnostic solutions. And so I think we're definitely the only managed SPARC on Kubernetes platform available on AWS, GCP, and Azure. So, yeah, and also right now we're definitely not the size of these big players. But yeah, I think – By the way, it's also good to have a little bit of competition in such a big market as big data or just SPARC. So if it were not us, it would be someone else.

[00:36:47] JM: how has the SPARC open source ecosystem been evolving over the past few years?

[00:36:52] JYS: Yeah, I'm really impressed with the SPARC community, and still how intense the development is. So SPARC open source, over the last couple of years, there's been huge improvements in terms of performance. SPARC 3.0, for example, was huge. Also, huge improvements on the infrastructure side, because two years ago, you almost couldn't run SPARC on top of Kubernetes. And now, it's GA and production ready, and I think a year or two from now it will be very rare to still run SPARC on top of Yarn. So yeah, overall, it's a really active community.

I think also, outside of SPARC, there are lots of open source projects. We mentioned Delta Lake earlier. We mentioned this SPARC on Kubernetes operator. We mentioned Delight, which is a SPARC UI alternative that we developed and open sourced. And so yeah, you also see all these tools, and the overall ecosystem is really moving fast. I think the yearly conference, SPARC Summit, or now I think it's called the Data + AI Summit, it's really impressive to see every year what gets published there.

[00:38:03] JM: Any predictions about the near future of data infrastructure?

[00:38:06] JYS: Yeah, I think we're in a transition phase, where still a lot of people – Like the general software engineering world has evolved a lot, with DevOps, with Kubernetes, Docker containers, CI/CDs. And honestly the way you produce software engineer, traditional software engineering, compared to 10 years ago has completely changed.

On the data side, a lot of things still look like they looked 10 years ago, with Hadoop, with running bash scripts to download your dependencies, your JARs with running into horrible dependencies issues with not having very isolated environments. And I think with Kubernetes, with the ability to run SPARC in containers, and it's not just SPARC. We see this in many different big data or just regular data technologies. We're going to have a 10X improvement in the next few years in the way people develop with data. And really, I think that's great news. And also, that's one of the reason why we started Data Mechanics is we wanted to help with this transition. Bring the DevOps best practices to the data engineering world.

[00:39:16] JM: Awesome. Well, is there anything else you want to add about data infrastructure, or company building, or anything you feel would be appropriate?

[00:39:24] JYS: Yeah, I think, obviously, people are interested in running SPARC on Kubernetes. That's what our company is built for. They can check our websites. But besides this, we recently you developed an open source free monitoring tool called Delight. Delight works on top of Databricks, on top of Amazon EMR, on top of Google Dataproc, on top of – Even on-premise deployments, on top of open source. So whatever the SPARC infrastructure, you can just install the Delight agent and then you'll have access to a web dashboard where you'll see memory metrics, CPU metrics aligned with your SPARC jobs and stages. And what we hope is to make it really simple for people to troubleshoot the performance of their applications.

So yeah, we open sourced it. If you look up our website, or if you just look for Delight, Data Mechanics Delight, I'm sure you'll find it. And we love your feedback. And if you want to learn more about it, we'll be giving a tutorial about it at the upcoming Data + AI Summit. So yeah, that's it from my side. Thank you so much for having me on the show. I hope this was interesting to the audience. And I'm happy to answer follow up questions. They can reach out to me on LinkedIn, or Twitter, or email, obviously.

[00:40:43] JM: Awesome. Thanks for coming on the show.

[00:40:44] JYS: Thank you.

[END]