EPISODE 02

[INTRODUCTION]

[00:00:00] JM: A large social network needs to develop systems for ingesting, storing and processing large volumes of data. Data engineering at scale requires multiple engineering teams that are responsible for different areas of the infrastructure. Data needs to be structured coherently in order to minimize the data cleaning process. Machine learning models need to be developed and deployed and iterated on at scale.

Areas of the company which produce data need to be decoupled from the areas of the company which consume data so that engineers throughout the company can reliably build tools on top of these different large datasets. In our previous episodes bout LinkedIn's data engineering, we covered two major components of LinkedIn's systems. The Kafka infrastructure, and the LinkedIn data platform that's used by engineers to productively build data applications.

Kapil Surlaker is a senior engineering director at LinkedIn, and he joins the show to discuss the bigger picture of LinkedIn's data infrastructure. Kapil works with teams across LinkedIn to understand the requirements for the products and the internal tools and translate those requirements into team structures and software platforms that let LinkedIn use data more productively. We discussed a wide range of topics, including engineering management, the modern data platform and LinkedIn's adaption of public cloud.

Full disclosure, LinkedIn is a sponsor of Software Engineering Daily.

[INTERVIEW]

[00:01:35] JM: Kapil Surlaker, welcome to Software Engineering Daily.

[00:01:37] KS: Thanks, Jeff. Great to be here.

[00:01:39] JM: So you're the senior director of engineering on the data side of things at LinkedIn. Is that correct?

[00:01:45] KS: That is correct.

[00:01:46] JM: Okay. So LinkedIn generates a ton of data every single day. Can you describe some of the major sources of data that are being generated across the product?

[00:01:57] KS: Absolutely. The first one is obviously when hundreds of millions of users are actively using the site and the mobile app on a daily basis. Through their interactions with the site and they're doing all kinds of activities, they're updating their profile, they're forming connections, they are liking and sharing posts, they're creating professional updates of their own. All of these creates usage generated content, which mostly gets stored in our distributed databases and also get served out of those distributed databases, right?

So this is what I would call maybe mostly transactional kind of data. The other sources of data which are in some ways much more higher in volume is one is what you would generally call as telemetry data. This is data that LinkedIn applications, including the frontend are instrumented to collect on how the applications themselves are behaving and how that's impacting user experience. So that tends to be an order of magnitude more data that are generally created as logging events and the funneled into Kafka and then flows form there onwards.

The third big class of data is what we would call as derived data, right? So this is data that is not generated as source of true data from either their databases or the event logging systems, but data that all our data produces and consumers, all your data scientists, and AI engineers, and other data users, they are processing these datasets to produce new datasets, new insights, new machine learning models and so on and so forth.

That tends to be really a source of data that is like truly there is no upper bound to it, because that's limited only by how much you can innovate. Broadly, I would say these are sort of the three classes of data; your transactional data, your logging data and your derived data.

[00:04:00] JM: Some of the previous conversations I've had with LinkedIn data engineers emphasized the structure of LinkedIn data engineering infrastructure on that event data. So when services are interacting, when users are doing different things, events are being

generated and they're being written to Kafka. You can do a lot of work with that event data. The event data is very rich. It's proliferate. There's a ton of it. Can you explain how LinkedIn evolved this architecture with an emphasis on the events? Because not every company has this kind of focus on event data.

[00:04:46] KS: I think that's an excellent question, and I think to some extent it kind of follows the progression and growth of certainly LinkedIn in this case, but in general, you can think of it as any applies to maybe any web company. As it grows through its usage and becomes sort of more sophisticate and mature, right?

The first phase is really much more existential, where you're kind of building your core functionality. Even to some extent, proving that it works, that it's a viable application that people are actually going to use, right?

At some point, you start to sort of go to that next phase where it's not existential anymore, but much more in terms of how do you make that experience much more relevant for users? How do we make it a more intelligent experience? How do we sort of generate that almost, if you call it magic, of when you see that user experience you go, "Wow! I didn't realize that could be done."

To some extent, getting to that phase really requires you to start understanding user behavior and your applications and how your users interact to those applications so that you can actually generate those insights and intelligence and magic. Without that, certainly, you'll have a user experience and a viable application, but certainly not going to be probably a great one, right?

That's where I think that focus on kind of events, and when we talk about events or logging, it really comes from that motivation of really making those experiences for our members truly delightful and magical experiences. To do that, you need to have the base data to work from. So you understand what the users are actually doing with the site. Then your data scientists and AI engineers can go to work with that data to appropriately create those member experiences.

[00:06:46] JM: Describe in more detail how that event data gets turned into these derived datasets that you're talking about.

[00:06:53] KS: Okay. So let's start with the event or the data generation itself, right? So let's say you have users going to the site and they're interacting with the site, and as a result they're creating all sorts of source of truth data, like you change your profile to say, "Oh! Here's a new degree in something that I received, or a new certificate that I received," and you're also navigating this site. You're reading through people's blog posts, you're liking it, you're sharing it, you're scrolling through stuff. All that is generating "the event data" as well through different sources.

So event data largely will flow through Kafka. So these events, as they're being generated in the various applications and services, they're being emitted into Kafka. The source of truth data is basically being generated out of the source of truth databases. All these data is getting collected periodically into the data lake, right?

So we built a framework called Goblin, which continuously ingest this data into a Hadoop-based data lake. These datasets are then available to all the data scientist and AI engineers and other applications engineers to work with, right? So for example, if you're an engineer in the PYMK team, which is People You May Know, which is one of the very well-known applications that you find on the site. What those engineers are trying to do is to look at all these data and figure out which users should they be recommending to you to connect so that you can expand your professional network.

If you look at it in a different way, that objective is to create a machine learned model to take the datasets as input and suggest new members for you to connect with. So that's an example for derived datasets, where you would take all the source data from the databases. You would take all these event logs and generate a model and then suggested pairings for those suggestions to be then surfaced to the users. This is just one example of derived datasets that gets generated. It's literally the function of the users that we have. So we literally have hundreds of thousands of derived datasets that are currently in the data lake.

[00:09:12] JM: I'd like to understand the problems that you see across the data infrastructure. So we've gotten a sense at this point for the fact that there is a lot of event data that's being created across the platform, and this event data is the raw data that data scientists are going to

analyze and build machine learning models on top of. They're going to build applications like People You May Know. They're going to build other recommendation systems, maybe a newsfeed recommendation system. All kinds of things they're going to want to build on top of that.

In order to facilitate that, you as the director of engineering working on the data side of things, you need to build the right frameworks. You need to put the right platform teams in place. You need to make data discovery a priority. Tell me about the biggest challenges that you've had to solve in building a data platform that allows data application engineers to be productive.

[00:10:18] KS: So when we talk about data application engineers, I think one important thing to note there is there isn't a single persona for it, right? There's an intense amount of diversity in kind of personas which would use the data that would kind of fit that description. For example, we have application engineers who are trying to build inside-based experiences into their products that go on the site.

You have data scientists who are trying to understand the user behavior to better kind of predict what kind of products and how the features should get built. You have AI engineers who are trying to use their data to build machine learning models, which are going to make your products more relevant. You have your sales analysts and marketers trying to understand their audiences to figure out in a data-driven way what are sort of the next best actions, if you will.

One of the biggest challenges in kind of building this data platform is really to understand the diversity of use cases that exist, right? Because each of these personas not only has a different use case, but they also differ significantly in sort of their technical expertise or the toolsets they're comfortable with. Just as an example, take something like SQL. There are some personas who are much more comfortable expressing all their work in SQL, and there are personas where that's just not enough and you need to do much more sophisticated processing.

One of the challenges, and we have built a portfolio of different tools and platforms and applications and infrastructure to cater to this wide and diverse audience, right? I think that's kind of one of the big challenges, because there isn't a single answer for all. It's not one size fits

all, but you have to kind of really tailor the solution to the audience that you're catering to, if that makes sense.

[00:12:25] JM: How much do you try to standardize on tools that people can or cannot use?

[00:12:31] KS: That's really a good question. I think there are certainly aspects of the infrastructure in the platforms where it really helps to sort of standardize. On the other hand, there are some aspects and layers of your tooling and the rest of the stack where diversity might be not only okay, but it might even be anchoraged to kind of satisfy certain needs of that user base, right?

Let me take an example, right? So one of the first principles that you do want to have is make sure that all your data is easily discoverable, right? So when you are a data applications engineer or a different persona that we talked about, when you start to solve a problem, you go through that phase of ideation where you want to discover datasets. What else exists? Then try to form that hypothesis around what it is that you want to do. If you don't have all your data in a way that is easily discoverable, that really sort of impedes any further progress that you're going to make, right?

So one of the things that we did is standardize on a few layers where you absolutely do need that standardization. For example, all our logging data has to flow through Kafka. All our data that gets then ingested into the data lake will be available on the Hadoop-based grid that all the data users use. Our data catalog, which is called Data Hub, makes it possible for all these users to easily discover and share datasets and reason about them in a very consistent way.

So you pick up a dataset, you want to know, "Hey, who owns these datasets? What are the SLAs that are associated with this dataset? What are the operational health metrics? What are the quality issues? What is the lineage of these datasets? Which other datasets does this dataset come from?" Because most of what the datasets that we have turn out to be derived datasets.

So in these layers, it is incredibly important to have standards and have that tooling, which is "allowed", versus others which are not. When it comes to certain other parts of their ecosystem, it is perfectly fine to sort of anchorage some amount of diversity, right?

An example I think is processing frameworks. Today, if you look at what are the tools that are available to these data users in order to interact with and process their data on the data lake, they have your legacy MapReduce-based toolkit. So you have your MapReduce, of course, but then you have your Pig and Hive, which as time goes is getting less and less adaption, because these are mostly legacy at this point.

On the other hand you have Spark, which is just exploding in popularity, right? Literally, most of the new jobs that the users are writing will overwhelmingly tend to be in Spark. Then for interactive tools, you have – Especially with the focus of SQL, you have tools like Presto, which is used for interactive querying and dashboarding and so on. We have a very fast OLAP engine called Pinot, that we built at LinkedIn and then we opened sourced it. That is used for surfacing extremely low latency in sites and the ability to slice metrics by a very large number of dimensions, if you will.

Users are free to choose any of these processing frameworks based on their needs and preferences. Even when it comes to SQL, somebody might have a preference for using Presto. Another person might have a preference for using Hive or SparkSQL. Those are perfectly fine, because as long as we have the standardization in the other layers that we talked about, they are still able to kind of share those datasets effectively.

If you process something in Spark and you create a dataset, that is still just as discoverable and usable by your other data users. So I think that's really kind of the guiding principle that we kind of use to decide which ones to kind of standardize and which ones to kind of open up and have a bigger menu of options, if you will.

[00:16:57] JM: In the previous two interviews that I've done with LinkedIn engineers, we talked about the Kafka infrastructure and the Hadoop infrastructure. As you've said, these are two things that are standardized. They're really the backbone of how data is created and stored at

LinkedIn. Again, we covered this a bit in the previous two episodes, but I think it's worth rehashing.

Now, the extent to which I understand it is that data that is occurring across the platform is getting logged in the form of events. There's massive number of events that are being written to different topics along Kafka, and periodically maybe it's over the course of four days or some interval that's shorter or longer than that. That data, the event data, is being flushed out of Kafka and written to the HDFS cluster for long-term storage.

Can you tell me more about the interaction, the usage of Kafka and HDFS and how that forms the backbone of your data infrastructure?

[00:18:09] KS: Sure. So Kafka, and the way we think about it in LinkedIn is certainly access kind of like a central lower system, right? Another way to think about it is it's really the flows and the streams that connect variety of data sources and collect all the data ultimately into a single location, right? There are multiple different ways that people sort of publish their data into Kafka, right? So it could come through your web applications. It could come through your mobile application. It can come through internal services. It can come through your internal apps.

Kafka acting as that backbone or the central nervous system really makes it easy for all these different applications and services in a way to talk to each other, right? Kafka has been always designed as kind of your low-latency data collection system, if you will. Hadoop on the other hand really is designed for much longer retention of your data. So all these data that goes into Kafka – And the retention of data in Kafka tends to be configurable, but given its use as more of a pipe, the retention typically tends to be on the order of days. You can certainly configure to have a longer retention, or if you have Kafka topics that are very well luminous, then you would typically keep it to a relatively small number of days.

All these data is then sucked out of Kafka, and we have that framework that we talked about called Goblin, which periodically ingest that data and stores it in HDFS, right? That ingest mechanism is actually fairly frequent of the order of tens of minutes. So Kafka would operate at much lower latencies in terms of the end-to-end flows. In Hadoop we would have that data being collected and stored every few minutes. Then you have these processes that sort of

compact that data. You have often datasets that get frequently updated. So you need to kind of dedupe it and compact it so that other applications can sort of use it.

That data then becomes available for all these use cases to use for an extended period of time, right? So in some sense, it is accumulated constantly, and as those datasets continue to accumulate and then the users in their various applications would continue to process the data to generate whatever it is that they're doing.

[00:20:48] JM: Okay. I know that LinkedIn is going to start moving some of its data infrastructure into the cloud. This is partly due to the Microsoft acquisition and just partly due to the fact that the cloud is quite useful.

Can you give me some insight into what parts of your data infrastructure you're planning to migrate into the cloud?

[00:21:09] KS: So there are two aspects of it, right? One is there are already a few pieces where you kind of use pieces that are available to you on the cloud, because it makes sense for you to do so, and you can do that at cost and be able to kind of scale it rather than having to kind of build everything from scratch instead of leveraging what is already available.

The other part is the transition that we are kind of moving into the Azure cloud, which is more of a longer term path for us to kind of move pretty much the entirety of our footprint into the Azure-based cloud infrastructure. There are sort of multiple reasons for it. One is – The foremost is actually not the fact that Microsoft acquired us. I'm pretty certain that we would have ended up in a cloud infrastructure even without it. Certainly, being part of Microsoft, it makes a lot of sense, because we can work with those teams to kind of build it in a way that makes it easier for something of LinkedIn's scale to move there as well.

Part of it is just the economies of scale. As large as LinkedIn itself is, being in a cloud environment just gives you economies of scale that are just at another level, if you will, right? Whether it's hardware, and especially with innovations that are happening in hardware, with GPUs and FPGAs, and a larger cloud vendor can obviously keep pace with it much better.

The fact that you have much larger pool of hardware that cloud vendors have also means that elasticity becomes much more feasibly, and this is especially useful for workloads that have a pattern that goes up and down at different times of the data, at different times of the week, when you have your peak periods, your holidays. The demand for storage and compute can obviously change.

When you're in a cloud environment where you have that elasticity, if you will, it's much easier to sort of grow and shrink your footprint to sort of optimize your users as supposed to on-prem deployment, where you're constraint by the limitations of a physical hardware, right? There are several other reasons for it, but that would certainly be – Those would certainly be one of the more common ones.

[00:23:41] JM: This thought just came to mind. I'm not exactly sure if LinkedIn is the right vintage, but do you know if LinkedIn was the – Was LinkedIn making its decision, its infrastructure decisions, around the same time that Netflix went all in on the cloud? Was there a period where LinkedIn was at all considering earlier to go aggressively into the cloud, or was it never really a serious consideration?

[00:24:08] KS: On the Netflix one, I have to admit, I'm not entirely certain on the timeline. I think – Certainly for the period I've been here, which is since late 2010. I believe Netflix was – The transition was a little bit after that, if I'm not mistaken.

[00:24:23] JM: After. Okay. Was it 2011?

[00:24:26] KS: I could be wrong. I'm not entirely certain about Netflix. But from LinkedIn's point of view, and even in 2010, LinkedIn had been around already for quite some time. It was already operating out of its own data centers. You kind of build that expertise on how to operate that entire stack on your own, right?

So it was never an issue for LinkedIn in terms of, "Hey, you don't really have that muscle in order to do it." I think periodically we would always kind of look at in terms of just doing the due diligence. This is a case where sometimes you see sort of the cost parameters sort of changing, but you look at how the cloud is evolving and how you are evolving. In the past, when we looked

at it, we kind of decided that it wasn't quite the right time for us yet just because of where different things are. I think now where things are and where we see things going, it's definitely much more sort of opportune time to make that call.

[00:25:28] JM: One thing about now is – One thing I'll say is I do feel like we're getting to a point where the cloud providers are really starting to come out with tools where there is no open source – I mean, we're still in the early days of this, but there is no open source tool that is the same as X. Something like – I don't know, BigQuery, right? Or like maybe managed TensorFlow from Google. I mean, these are not the most perfect examples, but it's getting very easy to imagine, like CosmosDB. I don't know. It's getting very easy to imagine cloud providers offering software that is simply just better than anything that's out there in terms of open source. By the way, partly because the operational burdens of some of these things are just so heavy that you really just want to outsource it. So that seems like something that's a newer development.

Are there any particular pieces of cloud infrastructure that you look at and you're like, "That's pretty appealing and I don't think we would be able to roll our own?"

[00:26:29] KS: That's a really interesting question, because I think there are definitely different considerations for where you are in the journey, right? One of the biggest differences, and I think for LinkedIn versus, say, a smaller company or a startup, the considerations stand to be rather different in terms of, "Hey, what can you use as simply managed service," or where your needs and your scale is really at a different level compared to where most of the market is.

For example, if you look at our usage of Kafka or even the general size of the grid footprint and so on, it would really tend to be – Like I don't even know at what percentile where you would imagine that maybe 99% of the users out there would have a much smaller scale, right? That sort of becomes sort of a forcing function for you to really innovate and push the boundaries in that space.

When there are problems at a different layer where – Certainly, when it comes to sort of just hardware resources or just kind of the virtualization layers where you just benefit from the innovations that are either happening in the cloud windows themselves, or happening in the

more broader sort of open source ecosystem, right? Where even the cloud vendors, they're sort of adapting these open source tools and customizing it to their cloud, right?

Things like on the virtualization and the container layers, you have things like Kubernetes, obviously. As you go higher up in the stack, you have both your proprietary querying systems, but then you also have your open source systems, like your Presto and your Spark and systems like that, if you will, right?

You definitely have that broader menu of options in some cases to choose from. To our earlier point, you will have use cases that fit into these different buckets. You won't have one thing that necessarily satisfies everything, especially in something else diverse and complex as LinkedIn. But that certainly becomes an added advantage.

[00:28:46] JM: Let's talk a bit about management. The director level role is somewhere in the VP territory, or like depending on what kind of organization you have. So you are a manager of managers, to my knowledge. So you're probably spending a lot of time with other managers. You're probably spending a fair amount of time seeing team level presentations. Tell me about the role of a director. How are you spending your time? What are your – I guess, your objectives and key results and the metrics that you're measuring for yourself?

[00:29:25] KS: Let me maybe talk about that in the context of – I mean, as you put it, sort of that growth and responsibilities, if you will, right? So when you are sort of managing, you're managing sort of a team of engineers, and then you're sort of managing multiple different, but related teams of engineers, which is when you're kind of a senior manager role and so on and so forth. As a director, then you're managing sort of managers. As a senior director, you're kind of managing teams that are going to manage themselves for multiple directors and so on,.

As you sort of go through these, I mean, ultimately the thing that doesn't really change in any role that you have, you're always focused on, "Hey, what are LinkedIn's priorities at this point and what is it that you can do to make sure your broader organization is aligned with those goals and make basically LinkedIn successful?"

How you approach it and how you do it day-to-day obviously has to change based on your role and kind of that function and the size of the organization that you deal with? In some sense, as you're kind of broadening your responsibility, you're much more focused on achieving those results through people and focusing on sort of coaching the folks that are obviously directly reporting to you. But also finding opportunities in kind of the broader organizations to sort of collaborate and achieve things that you may not be able to do simply by yourself within your organization, right?

I think that's sort of I think one way to kind of look at it and evaluate that. I'm not entirely sure that that aspect of fostering collaboration is captured in sort of metrics, if you will. But qualitatively and subjectively, that's certainly what you strive for.

[00:31:22] JM: And that collaboration, that's probably important, because there's a lot of – At a company as big as LinkedIn, there's a lot of areas of the company that are not necessarily known through our the organization. There's tribal knowledge.

So that stuff gets captured in your brain. If somebody mentions some problem to you, often times you're going to say, "Oh! Sharon can help with that. Why don't you go talk to Sharon?" They're like, "I have no idea who Sharon is. Can you intro me?" So, yeah, it seems like – In-company connector seems like an important role. What have you learned about management since becoming a director?

[00:32:02] KS: Just to go back to your previous question, and I think as you were segueing into this one, part of it is also I think how we think about as a company, right? Because there are application teams that are directly building products at LinkedIn, then then there are more what we call horizontal teams that are focused on sort of enabling and sort of had wider impact on the organization.

That's a leverage decision, right? We only have one Kafka team at LinkedIn. We only have one Hadoop team at LinkedIn. We have only one team that builds experimentation engine. We have only one team that builds data portals and data hub, for example, right? Which is the discovery portal that we talked about. This is an organization choice in some ways. You certainly have many organization that don't necessarily leveraged it in the same ways, and then you end up

with siloes where you have multiple teams sort of doing overlapping things, or sometimes doing conflicting things, sometimes doing redundant things.

Part of that organization design also means that horizontal teams like my own have a ton of different partners. So it really places the emphasis back on collaboration where it's extremely important to drive a result for the company as a whole and to enable the organizations which depend on you to achieve the results that they want to achieve, right?

So I think there's certainly a ton that I've learned as a manager over the years. I think the two things I would probably call out as things that would stand out is, one, certainly, focus on the larger aspects of collaboration, because I think I'm certainly misquoting it, and we can correct that if I got that wrong. Then I think, say, if you want to go fast, you go alone. If you want to go far, go together, right? Something like that, right?

[00:34:04] JM: No. No. [inaudible 00:34:04].

[00:34:06] KS: That's really something that you have to really internalize, because a lot of the times your inclination as engineers is certainly being sort of [inaudible 00:34:16] to you don't get things done very, very quickly. But when you focus on the long-term, it's really about not just focusing on the relationships, but thinking about how do you achieve the right outcome in the long-term by influencing a number of other teams and organizations that can then partner and go along with you, right?

The second one I would say probably kind of in the similar way is really to focus on the people, right? Certainly, the collaboration is the other aspect of it. But, certainly, when you are a manager at any capacity, but certainly as you grow, the results that you can achieve in that role are directly dependent in some ways. That's the only way you would get those results is by investing and growing in individuals, right?

Certainly, at LinkedIn, I've been lucky to work with folks who are really absolutely the best in their fields and just being fortunate to work with them for several years now. When you do that, you kind of realize what you get through by constantly investing in those individuals and also those relationships.

[00:35:34] JM: To return to the technical. Much of what we talked about earlier was about building a platform to allow data application engineers to be productive. One of the modern systems for building data applications is the realm of machine learning. Building and deploying machine learning models is not easy. Back testing those models is not easy. Updating those models is not easy. Tell me about the process of developing machine learning infrastructure and machine learning standards within LinkedIn.

[00:36:14] KS: So machine learning is certainly a very rapidly evolving field, and the pace of innovation is just staggering, where you have new frameworks coming up constantly, new frameworks for deep learning and new frameworks pretty much through the entire lifecycle of machine learning. Now, given the space that it touches, we actually have an effort at LinkedIn called ProML, which stands for productive ML, which really aims to kind of standardize the tooling across the board through that entire lifecycle of machine learning. So how do you kind of build models in a consistent way? How do you deploy those models? How do you monitor and then update those models?

Now, I'm not necessarily the best person to kind of go into the details of like ProML itself. That's an effort that is a big priority for us across the company, right? That touches various parts of the ecosystem, certainly, including all your offline infrastructure, your online serving systems, your deployment tooling and so on.

That is something that we've been working on for a while. It has seen fantastic results in production. If we haven't talked about it publicly, I'm pretty sure that we would. Yeah, I think someone like [inaudible 00:37:35] is probably a much better person to talk about it productive ML.

[00:37:40] JM: From 2010 to 2014, you are working on some internally-built data systems at LinkedIn, and I've had previous conversations about Pinot and with Carl. I just talked about Dali. So LinkedIn has a history of building new data infrastructure tools internally. Now, we're in this time where there's so much great open source stuff you can take off the shelf. You're starting to use cloud services. There're so many cloud services one could use. Why are you still building brand-new tools?

[00:38:19] KS: That's a really good question. So let me start with talking about open source, right? At least certainly for the time I've been here, LinkedIn has been a very, very strong advocate of open source, and that's super exciting for a number of reasons. But being an open source advocate, it goes both ways. So there are definitely cases where we have built a lot of systems. Then for the most part, open sourced them. In some cases when we haven't, it's largely because we didn't necessarily prioritized it upfront, and then it became sort of much harder to open source it rather than a desire not to open source.

In many cases, we have adapted tools that have been built elsewhere and open sourced. We certainly kind of looked at those and then decided that leveraging those was much better, right? I think as the years have passé, I think I don't necessarily – I wouldn't necessarily say that that balance has changed, but I think the philosophy of making the decision I think has stayed roughly the same, right? Which I don't think we have – Or rather in every case where we have built new infrastructure and new tools, it's very consciously having looked at the ecosystem and then placed bets carefully around where we see that ecosystem going and whether we see our needs being well-addressed in that open source ecosystem always.

Just going back in time, you referred to the early days of LinkedIn infrastructure in 2010 and so on. If you go back at the time, and our immediate problem at the time and one of the first projects I worked on at my time in LinkedIn was to build a distributed database layer that would help us reduce our dependency on Oracle, right? Certainly, for cost reasons, but also for operability and uptime reasons and so on.

When you look at it today and you go, "Yeah, you have CosmosDB on the Azure cloud and you have a ton of other options." If you go back to 2010, there were much fewer options that were out there. Certainly, the options that were there weren't necessarily built for LinkedIn scale. We actually kicked the tires on a few to make sure that when we were making those decisions, we were making them very consciously, because these are important bets, and you would need to invest time and resources for multiple years to get the benefit.

If you look at data processing systems that we've looked at recently, you would look at something like Spark or something like Presto, and there really isn't a reason why you wouldn't

adapt those with the intention of then enhancing them for your users, right? Every time we picked up something from the open source ecosystem, we have invested in those to first meet our needs. Certainly, you have scale, but also other aspects, and then giving it back into the open source community, right? So we've done it with Spark. We've done it with Presto, and we'll continue doing it with the other tools that we would adapt.

Even so, you would find gaps where they're not very well-addressed in any of the existing solutions. I think Pinot was a great example of it, right? There's a ton of analytics or OLAP engines out there. But the specific problem that we were looking at when we decided to build Pinot was to have an extremely low-latency OLAP system, because we wanted to build side-facing products using those. When I mean low-latency, we're talking about milliseconds. We're not even talking seconds, which most of your interactive analytics databases talk about, right?

So when we looked at the ecosystem and we said, "Okay, let's try things that are out there and see what works." There really wasn't anything that gave you guarantees of millisecond level performance when they were dealing with OLAP. You found tons of systems which would do queries in seconds, but that simply wasn't good enough.

So that is an example of where you make a very conscious choice of investing and building a solution because there isn't anything that meets your needs. In Pinot, that's exactly what we did. After we built it, we scaled it internally to literally hundreds of use cases, and then we put it back in the open source, right?

That allows a lot of other companies in and outside the valley to use a system like that to satisfy their own analytics needs. So I definitely see that sort of balance staying that way, because we'll always have LinkedIn being somewhat unique in its setup as a professional graph and because of the scale and because of that network aspect. We'll always have use cases that are not entirely addressed by what is out there. When we see those, it's not building infrastructures for the sake of building it, but because we do see a gap in their ecosystem.

[00:43:30] JM: What's been the hardest part of scaling LinkedIn's data infrastructure?

[00:43:33] KS: From a scalability point of view, I would say there are two aspects. One is when you look at the infrastructure itself and you think about, “Hey, what is it that you need to do to scale it?” The other one is more organizational. So I’ll talk a little bit about both. When I talk about infrastructure, what I mean is we are at a scale where we don’t have the luxury of running into problems that a lot of people have run into and sold already, right?

When you are, and it often happens when you’re adapting some new tool for the first time. As you go through that scaling aspect, it’s great, because other people have sort of used it and you have their luxury of, “Hey, some of those issues are already taken care of.”

But when you get into the scaling aspects where very few companies have, you start to discover problems that really there is no other way of discovering, right? We ran into this with Hadoop. We ran into that with Yarn. We ran into that with Spark. All these cases, the interesting aspect is because those problems show up only at a certain scale, you don’t even have a very good way of kind of testing it, because to test it, you would need a large enough footprint to just test it with and nobody has few tens of thousands of machines just lying around for free for testing, right? So we actually invested in a lot of tools that kind of helped us kind of simulate those results and test it with a smaller footprint.

On Hadoop, we built called something called Dynamometer that allowed us to test, “Hey, what would your cluster look like in its performance aspect if it was 10,000 nodes? What would it look like if your single cluster was 15,000 nodes?” You can kind of then get an idea of what it looks like well before you actually get there so you can start solving those problems, right? That’s more the technical aspects.

The other parts of scaling is scaling from an organizational point of view, right? That turns out to be true specially in kind of the analytics and big data ecosystem, because the scaling problems are, one, function of your user base, of your member base, and the traffic on the site. But the other aspects of it come from your internal user base, right?

If you have tens of data scientists that are trying to use your data versus you have hundreds of thousands, it’s a very different scaling aspect organizationally. As they become more productive, they’re also experimenting more and more. They’re innovating more and more. So if you go

back 10 years, we didn't really have an experimentation system. We didn't need an experimentation system even at that time, because the site was just so small, right?

Today, where we are, you wouldn't even talk about pushing significant changes to the site unless they were behind an experiment. We built an internal system called TREX, which also kind of illustrates a point that you were earlier asking about, "Hey, what is the reason to build such a tool?" There are very few companies that kind of do it at a scale that we do it and as rigorously as we do, right?

We built a tool called TREX, which is used for all of our targeting, ramping and experimentation needs. That cultural shift that comes with it of basically saying that, "We're not really pushing significant changes to the site, unless we have an experiment behind it and we can clearly see the impact of releasing those changes, the releasing of changes on your business metrics," right? So that is kind of an organizational and a user behavior aspect of scaling as well, which is something that we have also gotten much better at over the years.

[00:47:36] JM: Has cost management ever been a significant issue?

[00:47:39] KS: Cost management, I would say yes and no. Yes in the sense that, yes, there's obviously something that you pay attention to. Especially in the big data space, because of this multiple factors that affect your scaling, you always have to be conscious around how quickly are you growing. Knowing the sense of, I think, what really matters is not the cost itself. But what are you generating out of that cost? So the ROI of that investment becomes much more important to pay attention.

So one effort that we've invested in significantly, something that we call resource intelligence, right? The idea is really not just around reducing cost, but really raise the visibility of where are you investing your hardware resources. What are the systems or the applications that are getting used out of it? So that you can make rational decisions around what are the places where we may be getting lower ROI as supposed to places where we get high ROI. Then you can really invest your efforts into whether it might be cost reduction or it might be improved efficiency through investments in the software layers and so on and so forth. You can really then focus on where those efforts are really worth spending.

[00:49:02] JM: Tell me something new that you learned about data engineering in the last year.

[00:49:05] KS: I think the most interesting thing about data engineering, certainly in the last year, but also I think maybe in the recent past, is I think that role and that function itself is going through a transformation. If you look back sort of further back in time, the practice of data engineering might have involved using sort of very kind of your traditional tools, and in LinkedIn, we certainly had those, right?

So most of your function was really about, “Oh! You have your Teradatas and Oracle data warehouse before that, and you have your traditional analytics tools of micro-strategy and Tableau and so on and so forth.” The function really is about saying, “Hey, how do we use these existing tools to sort of meet the needs in terms of building those supports and dashboards and so on?”

As Oracle system has evolved to have more sophisticated needs, and certainly if you go back in time, using machine learning and using AI to the extent that we do today, didn’t certainly exist back then. So all these innovations push you towards, “Hey, what are the innovations that need to happen in the infrastructure and the toolings in the platform space?”

I think that pace is only going to increase, right? As the innovations and the need for it also, and in the AI and ML space increase, what it means is it’s going to drive innovations in the infrastructure and the tooling layer much more aggressively. It is certainly changing the function of how we think about data engineering. In the years to come, I’m pretty certain that we’ll continue to drive those changes.

[00:50:52] JM: Kapil Surlaker, thank you for coming on the show.

[00:50:54] KS: Thanks, Jeff. It’s been great chatting with you.

[END OF INTERVIEW]

[00:51:06] JM: LinkedIn is a software company with the goal of creating economic opportunity for every member of the global workforce. LinkedIn is hiring data scientists, software engineers, researchers and many more roles for its engineering team. To find out more about the problems that LinkedIn is solving and the teams that are solving these problems, check out engineering.linkedin.com, where you can read about culture, open source projects and hard problems that LinkedIn has worked through and blogged about.

Thank you to LinkedIn for sponsoring this show and for creating software that I use every day.

[END]