1:40

Alexey

**This week, we'll talk about DataOps. We have a special guest today, Tomasz. Tomasz is a DataOps who lives in Poland, in Poznan. After working in product analytics, data engineering, data science, and machine learning, he fell in love with operations. He finds peace in fixing poorly-written IAM roles and teaching people. I really love that line. So welcome – welcome Tomasz.**

2:06

Tomasz

Hello, everyone. Thanks for joining us on this kind of niche topic still. I hope after this podcast/vlog, it will become less niche because it deserves attention in my opinion. Yours probably as well.

# Tomasz’s background

2:25

Alexey

**Yeah, indeed. I'm surprised that you worked in so many different roles. Probably we'll start our interview with that. Before we go into our main topic of DataOps and becoming DataOps, let's start with your background. Can you tell us about your career journey so far?**

2:42

Tomasz

Sure. Well, university time should count to the career time. I studied econometrics, which is kind of exotic, or at least it was back then. Then, kind of accidentally, because a friend of mine shared the information with me that “Yeah, this company OLX is actually hiring.” I was like “Never heard of it.” So totally by accident, I joined the company as a machine learning trainee. Then I was working as a junior data engineer, then as a data scientist, and so on.

Basically, alongside all these roles and positions, I believe I touched on a lot of steps in that whole cycle of – analyze data, create the model, publish the model, create the product, yada, yad, yada. So because I get easily bored with stuff, I try to touch as much as possible and don't become an expert on one particular domain or only one step like only building models or only doing the analysis. I wanted to know what the other folks are doing and why and how this connects to my own work. That's why the scope was rather broad.

4:31

Alexey

**So you tried all these positions – all these roles – while working at the same company? At OLX?**

4:38

Tomasz

Mostly, I didn't work only at OLX. I briefly worked at the Central Statistical Office (Główny Urząd Statystyczny) in Poland to see what the government’s statistics look like. There were also some brief episodes with university and some other companies and yada yada yada. But mainly OLX.

5:03

Alexey

**You mentioned that you got into OLX as an ML trainee by accident. I think many of our listeners, or people who are watching this, might be wondering what this accident was. Can you tell us more about that?**

5:20

Tomasz

Sure. I mentioned “by accident” because I haven't actually searched for it specifically. Yeah, I definitely wanted to be into machine learning and stuff. I just received a link that said, “Hey, some company is hiring for a machine learning internship.” I was like, “Why not?” Why this might be a little bit awkward to some of the listeners because right now machine learning is a hot topic. Back then, it's probably hard to believe but, at least in Poland, nobody pretty much heard of machine learning. Back then it was called multi-dimensional analysis at university.

I basically took the course, where I was doing all the principal component analysis, classification, clustering, and all that other stuff, but it wasn't even called machine learning. It was something like “multi-dimensional analysis” because a lot of folks there were from a statistical background, and they just had different naming conventions, let's say. So that was the “accident” [chuckles].

# What Tomasz did before DataOps (Data Science)

6:34

Alexey

**So you worked as a machine learning trainee, then you worked as a junior data engineer, then you also worked as a data scientist. After working as a data scientist, you became interested in DataOps and you became a DataOps. But before you became a DataOps engineer – I don't know how to properly call this role… Before you started doing DataOps, you were a data scientist. Can you maybe tell us what you were doing? What kind of tasks did you have? What kind of questions and responsibilities did you have?**

7:08

Tomasz

Sure. I was working mainly in the under the models customer unit – OLX has different units, so I wasn't in the core data science team, rather, under one of the business units. I was doing mainly behavioral analysis, like, analyzing clickstream data, trying to do some models on that to capture some interesting signals, trying to catch people who might perform actions that we are actually interested in and trying to grab them basically. Also, I was doing a little bit of product analytics, alongside the work plus, obviously, some operations because that's how the DataOps journey started for me.

A lot of people might believe that if you're a data analyst, or data engineer, or data scientist, you don't need a lot of operational skills. That's a little bit of a misunderstanding, because even if you are working in data – if you are a programmer, you are perfectly aware that they are SREs. DevOps and stuff like that – in data, not really. But that might be a little bit of a context story, but I guess it's relevant for the discussion. So let's say you are a data engineer, and all of a sudden you start a new project and you need to create a new S3 bucket or Kinesis stream or whatever. You believe that you won't be deeply involved in that because there is a platform team or there is an infra team or whatever other team – the central core staff. So you go to them, like “Hey, folks! I need a new S3 bucket.” And they are like, “That's cool. Here's the link to the repository. Create the merge request, and we'll do the review on priority because we like you.” Then you’re like, “Oh, crap. I need to learn Terraform, Terragrunt, Atlantis and obviously a cloud provider.” So that's about “not needing to learn operational skills”. Or maybe as a data analyst, you also believe “Ah! I don’t need that stuff.” But then all of the sudden you need to actually understand the dataflow in the company.

Why? Because we prepared an awesome report, provided a view on some part of the business unit – the results are pretty important. They are about to be shown to the leadership and your boss asks a really uncomfy question, “How confident are you with the results? Because that 10% drop in the revenue looks kind of suspicious.” So maybe some ETL jobs failed this and that. Maybe some servers were basically down. Maybe there's an issue with tracking or some other stuff. So then, all of the sudden, you need to understand not only your path but the whole pipeline. Or even as a data scientist, you might be thinking “Okay. My job ends pretty much at the level of a Jupyter notebook and that's it. Then there will be some almighty Big Data team or machine learning engineers who will take that stuff and put it on prod.” Yes, and no. Because, for instance, you created a model where the prediction time is one second. The product folks came through and said, “Okay, that's cool, but you need to go down to 300 milliseconds. Oops.” Or you create a model that is outputting, let's say, a list of cookies. Some folks from marketing are like “That's super cool. But we don't have any possible tool that is able to consume it. We can consume some rules.” So now you realize “Okay, I created a model that is pretty much useless because the output format doesn't match.”

Those are pretty hard tasks and it's pretty unexpected for a lot of folks, because they believe that there will be someone else who will be doing that. That's not necessarily true and that's pretty much where DataOps comes in, because people need help in that manner. Also, there is a possibility for a huge misconception here. Some listeners might be thinking now “Okay. Now I finally get what DataOps is about. This will be the person who will create the infrastructure for me or who will do the maintenance for me.” Not necessarily. DataOps is the person who will help you work effectively, who will help you design the solution, who will basically make your work less scary. He will not do something for you. He will teach you how to do it effectively.

12:40

Alexey

**So in your story, you needed a bucket, you needed the Kinesis stream, and then you asked somebody for help and they said, “Oh, we’re busy. Create a pull request (or merge request, whatever).” And you were expecting that these people would help you, but they kind of said, “Okay, just do it yourself. Here is the repo. You need to create a merge request.” Right?**

13:07

Tomasz

Also to, to maybe defend the platform teams a little bit, or the security or the SREs – they are not supposed to do that. Let's say you are asking the security team to create a service role for you. It will be not very responsible from their side to just throw out of the fence some relative stuff to someone who doesn't know how this thing is supposed to work. They should help you – they should do the review and they should guide you, but not necessarily do the job for you.

13:49

Alexey

**Okay. So in all these roles that you had over this time – as a data engineer, as a data analyst, as a data scientist – in all these instances, you needed to touch the infrastructure, right? And this is how you learn how to do this and this is how you fell in love with doing all this stuff. Is that right?**

14:08

Tomasz

That's totally correct.

# Why Tomasz made the transition from Data science to DataOps

14:12

Alexey

**So when did you realize that you actually enjoy doing this stuff more than your work as a data scientist? How did it happen?**

14:20

Tomasz

I'm glad that this question was asked. Because, again, there are a lot of misconceptions about which role in data is more important than the other role. Before answering the question directly – no role is more important than another role. Especially, you might be thinking “Do I need to have a DataOps in the company?” The answer is “No,” which might be surprising for some folks. It's not a mission-critical role. It's rather more of a support role. Imagine that you're playing a game and you are going to the boss fight. You are going to the particular boss fight with a broken sword and without potions. Is it doable? Yes. Will it be fun? Probably not. DataOps is kind of that buff, that fixed sword, that plenty of potions, and stuff like that. It's useful, but not necessarily mandatory. [chuckles]

Now answering the question maybe – Why? Because I wanted to solve problems and it turns out – which kind of correlates with what Andrew Ng (if I'm pronouncing the surname correctly) discovered or is trying to make people aware of – is that the whole domain went recently from the model-centric approach to the data-centric approach. Which essentially means that, if you are doing work in some large company and your job is not to create the model to explain the behavior of something and then forget about it, but rather you are creating a data product. Surprisingly, the majority of the work in creating the data product is in operations – it’s in data and not modeling itself.

Naturally, because we were working on data products, we're solving more and more and more of engineering problems, not necessarily scientific ones. So that's how I fell in love with that stuff. But DataOps is not necessarily more important than data science. Actually, with Alexey, we both know a perfect example of that – because while I was transferring from data science to DataOps, we know one person who was actually doing the opposite. One SRE with plenty of experience in operations was going through the same bridge, but in the opposite direction – from Ops to data science. So no career is worse or better than another. Don't get us wrong.

17:46

Alexey

**I think you said at the beginning that doing the same thing for a long time is boring for you, because you want to do a lot of different things. And I think many people are like that. Right?**

18:00

Tomasz

That's also a factor.

18:03

Alexey

**So, not everyone but some people are – it's just too boring to keep doing the same thing all the time. For our colleague, it was probably boring to do Ops stuff all the time, and he wanted to try something else. So you were doing data science, and then you realized that you need to spend a lot of time doing this infra stuff and this is where a lot of problems are. Like you said, you love solving problems and I guess you saw that there are actually a lot of problems in the Ops part, right? The reason you became interested is (maybe I misunderstood you, but) you felt like “Okay, I'm more useful solving these problems and I actually like doing this.” That's why you started digging deeper into this.**

# What is DataOps?

18:59

Alexey

**We've been talking about DataOps for quite some time, or your transition into this, but we didn't actually discuss what DataOps is. So what is DataOps?**

19:10

Tomasz

I love the explanation that was given at one of the talks at DataTalks.Club, with Chris, if I remember correctly – the grandfather of DataOps. Essentially, a DataOps engineer or whoever takes a look at how people work, not doing the reports himself or herself, or modeling, or putting the models on prod, etc. He or she looks at how people work, where there might be inefficiencies, how to overcome them, and basically help people produce meaningful results faster, in a more pleasant, less scary way, and stuff like that.

For me, that might be the shortest description possible, which is essentially the same stuff that DevOps are doing. If you think about that, it's not a new concept. Programmers knew that for some time, that's how DevOps came about. But even before – Lean, Kaizen, Six Sigma, stuff like that in companies that are producing something physically – that was there and the concepts are super similar. We are producing software, okay, but the philosophy is exactly the same, IMO.

# How is DataOps related to infrastructure?

20:56

Alexey

**Okay. But how is it related to infrastructure and all these things that we talked about? You said DataOps is about solving inefficiencies, helping people overcome problems, produce results faster – how is this related to infrastructure?**

21:16

Tomasz

Excellent question. Because DataOps is not only infra, it's also about helping people write better SQL queries – as simple as that, or helping people keep… let's call them “secrets,” stored in the proper locations and accessing them the proper way and stuff like that. But it turns out that there’s actually a lot of confusion about infra like, “Okay, how should I create the S3 buckets through GitOps?” Someone might be like, “Okay, let's try some examples. Just switch the names and try to apply that and see what will happen.” But then, Atlantis returns some error, like, “Oh, crap, I just broke the production.” No, everything is okay.

But for the first time, if someone never used GitOps to make some changes in the infra, it will be scary, honestly. So that's why it's rather a good idea to sit with that person on the Zoom call and go step by step. “Do you know Terraform?” “No.” “Do you know Terragrunt?” “No.” And blah, blah, blah, and all of the sudden, all the errors, all the concepts become less ambiguous and like it’s for a specialist – it’s getting more familiar with technology. To give it a more human face.

23:04

Alexey

**Yeah. You mentioned GitOps, creating a bucket through Atlantis – can you maybe walk us through the process? What exactly does this process look like? Maybe high level without going too technical – just for those who don't know. I think I know a little bit, maybe. I will also check if I know how this thing actually works. [chuckles]**

23:26

Tomasz

I’m guessing you do. [chuckles] Essentially, you don’t need any stuff like infrastructure as code because that's Terraform, Terragrunt and all that jazz is for – or Cloud Formation, if you're working with AWS. It can… [cross-talk]

23:42

Alexey

**What is “infrastructure as code”? Before we even go there, for those that don’t know.**

23:49

Tomasz

Good question. So, if you want to create something, you can also go to your web browser, authenticate, and you can just click here and there, and create some resources – some roles, some buckets, some Kinesis streams, stuff like that.

24:06

Alexey

**With your mouse, in the Amazon web interface. Right?**

24:11

Tomasz

But then, imagine that you want to create one bucket on staging and one bucket on production. So you are doing essentially the same stuff, which changes just a little bit of the S3 bucket name or some tiny details. So that's how a lot of smart folks in operations came to the conclusion that *maybe* if we define all the configurations, all the infrastructure as code – not as clicking here and there – it will be more manageable. We can do some audits. We can do all that stuff via merge requests, which can be reviewed. Everybody in the company will be able to create a merge request as they say it, and then someone from more infra teams will go there and check your merge requests – but everybody can do, that's the enablement.

Essentially, you're writing some Terraform, which is a huge config file, let's put it that way. Terragrunt is putting some variables to the Terraform code and what Atlantis does is it displays all the changes that are about to be made in the pull request on the merge request, where you can review what will be changed if the given procedure would be applied. It's kind of a “dry run”. Then after you get the approval from SREs or whoever, you can just click on “merge” (or “apply” in Atlantis) and the changes are made to the infrastructure. That’s probably still a little bit technical, but that’s the high level overview.

26:21

Alexey

**I'll try to summarize. We have “infrastructure as code” tools, and Terraform is one of them. With Terraform, we can create a config, and with this config, we create a bucket, we create this Kinesis thing that you mentioned – as a config, as code. Typically, without Git, what we would do is something like Terraform Apply on our computer. But with GitOps, the way we do it, is instead of getting this code and running this locally, we create a branch and in this branch, we put this piece of code and then we create a pull request or a merge request. Then what Atlantis does is apply Terraform or tries to see what would happen if we apply this to our cloud account. Then somebody – some SRE or some DevOps person or DataOps, if you will – comes, sees that your code is not breaking anything, they accept the merge request, and then you merge. At the end of this process, you have a bucket and the Kinesis stream in your account. Right? That’s the process?**

27:34

Tomasz

Exactly like that. You mentioned something that I haven't. [chuckles] Essentially, without GitOps, you will be as Alexey said, you will be doing all that stuff from your laptop. So you will have to have the proper data Terraform version and all other tools. Now imagine some poor data analyst trying to install Terraform config dat. That will be painful.

28:06

Alexey

**I think it was the biggest problem. At OLX, we thought that it was a good idea to ask data scientists to work on infrastructure and for that, they needed to clone this repository with Terraform and then do Terraform Apply, and then apply these changes to the cloud. The biggest problem was actually installing this and making sure you *can* apply. [chuckles] Many people couldn't do this, because it's just too difficult and this is not what data scientists are trained to do, typically. This is not what we learned at university. [chuckles] But I guess for you, you liked this part, right? You enjoyed doing this thing?**

28:48

Tomasz

Yes… To be a little bit more specific I haven't enjoyed…

28:59

Alexey

**[laughs] You didn't enjoy it?**

29:03

Tomasz

No, it’s just… it wasn't that much about creating the infra. I haven't fallen in love with creating Terraform code. It was rather about helping people do it – making them comfortable with that stuff. That was the main part of doing DataOps more for me.

# How Tomasz learned the skills necessary to become DataOps

29:34

Alexey

**Okay. One of the questions I wanted to ask you is – how did you actually learn this thing? How did you become a DataOps? But I think from what I understood is, you just simply had to do this and then you had a Zoom call with some sort of SRE or platform engineer who would guide you through this process – who would explain to you what Terraform is, what the other things are, how exactly you need to create this merge request to get your S3 bucket, and this is how you learned. Right?**

30:04

Tomasz

More or less, yes. But also to maybe make the process easier. If I would start again learning the same stuff, I would definitely narrow the scope. Because if you're asking some DevOps engineer, “Okay, I want to be more into operational stuff. What should I learn?” “Linux, then some cloud provider, Docker, Kubernetes and yada yada yada.” I would reply “Okay. So after five years, I will maybe become useful, finally.” [Alexey chuckles]

This is kind of a misunderstanding, because if you are working in the database, AWS has what –probably like 200+ different services? Scan the list and then answer yourself if you will be spawning some fleet of IoT robots? I have doubts. If you will be working on quantum computing? I have doubts. If you will be working on ground stations? Probably not. After that pre-filtering, you will come to the conclusion that “Okay, I actually need the IAM roles, EC2 machines, S3 buckets, Kinesis maybe, EMR.” So out of 200 services, you will end up with “Okay, I actually need 20 of them.” That narrows the scope a bit. A good example of the possible roadmap could be the roadmap.sh/DevOps, probably. It's also pretty accurate for the data domain. But I would say “Good enough is quite okay.”

You don't have to spend like five years in some kind of basement constantly training and learning, and then finally you will become the “useful” guy. Not necessarily. Every single team like security, SREs, platform team, and whoever, has a list of their least favorite tasks. For SREs, it might be something like, “Okay, every single resource needs to be tagged with like, name=something, owner=something, the on-call guy=something. Every single resource needs to be tagged. This is the sort of task that nobody likes to do.” So if you are a junior in the operations domain, you're basically going there and asking for that kind of rookie task, and they will be super helpful to give you that. Alongside doing so, you will learn a ton and everybody will basically love you. Because you are taking the crappiest work possible from them and at the same time, you're actually learning. So it's a win-win situation.

Take the security team, for example. They might have problems with “Folks are using privileged mode in the Kubernetes runners, that is like Docker. It's kind of not okay.” So you have to identify all such cases, go team by team and explain to them how Kaniko works. It's also a completely rookie task. You will learn a ton while doing so and you will know the people you will be working with better. Again, it's a win-win situation. So establish the connections. Make people from the technical teams know about you. Plus, teach others, obviously. Also – start simple. You don't have to start from administrating the Kubernetes cluster. You can just do the Docker image on your laptop, then push it to some registry, then push it to a different registry.

The first time, maybe to the GitLab registry, then to ECR. Then try to apply some security scanning. Then create that on the CI pipeline instead of your laptop. So those steps and blah, blah, blah. Out of making some little steps, you will finally go to the more or less end of the path into that particular task. One last piece of advice on the learning process – accept that it will be uncomfortable.

If you are from the data domain, you are probably closer to the PhD in stats than to the Linux admin. So now, out of being a senior data scientist or a super-powerful machine learning engineer or whoever, you are going to some very different domain. So it will be like forcing a weightlifter to do cardio training. It's a different speciality, so it will be uncomfortable and it doesn't mean that you are unqualified, or the worst case, stupid or whatever. It’s just a different domain. It will be tough. You've got some folks out there to support you.

# Becoming comfortable with terminal

35:55

Alexey

**Speaking of that – right now we have a machine learning engineering course and we are currently covering the deployment module. So far, for many students (we're on week five right now) it was fine, because it was a Jupyter Notebook. But now, all of a sudden, from this convenient, comfortable environment of a Jupyter Notebook, we end up doing things in the terminal. And I have a question for you – I think you mentioned that you studied econometrics, right? Then you worked as a data scientist. I guess this Linux admin stuff wasn't something life prepared you to do. [chuckles] So how did you learn this thing? How did it become comfortable for you to work in this environment?**

36:43

Tomasz

Honestly, by making every possible mistake that could be made. I know this might sound stupid, but that's how it was. But what may help you feel more comfortable with the command line is to do the proper setup. If you look at the terminals of some service or something, they will have plenty of [audio cuts out] and that sort of jazz and that's for a good reason. It probably doesn't have auto completion, syntax highlighting…

37:32

Alexey

**I don't know if it's just me or everyone, but I lost a part of your answer. Just to make sure we would get it – you said talk to SRE who already has proper setup in the terminal and then I lost it.**

37:47

Tomasz

And ask them basically what they have installed and what for. Because, honestly, terminal without auto-completion, without syntax highlighting, without proper bash RC – it's not a comfy place. With the proper bash RC setup, it will be much more friendly – you will immediately see the possible mistakes. commands will autocomplete. It will just be better.

38:20

Alexey

**Okay. What if I don't have an SRE who already configured bash RC and can just share this information with me? What is the best place to look for this kind of information?**

38:29

Tomasz

I'm super glad you asked this because my answer for that kind of question is always the same. There is a place called YouTube, which is awesome. [cross-talk]

38:41

Alexey

**[laughs] I thought you would say Google.**

38:44

Tomasz

Google is well… It's also awesome because you have articles on Hacker News, Medium, whatever. But it may happen that someone on YouTube already created a full-blown video about how to configure let's say Data Sage or anything about that. So basically type “command line tutorial course setup” whatever, and I bet you 50 bucks that there will be something in the first 10 results.

39:26

Alexey

**Yeah, I think I saw a couple of videos like that – that are one hour long and some of them are even longer – that show you how to set up your environment from nothing, from a clean Windows/Mac OS/Linux, whatever you use. I saw a video with Windows. One year ago I switched from Ubuntu to Windows and I needed to prepare the environment for that. Of course, for me, everything was alien. I found the tutorial that just walked me through the entire thing of how I should set up a terminal. At the end it was like a usual Linux setup with just that single video. I unfortunately don't remember that video. But it wasn't difficult to find. I think it was one of the top results.**

40:11

Tomasz

People just *love* to create videos. There is a high chance that someone already created something for exactly your case, like “How to set up a Kubernetes cluster from scratch,” or “full course about how to prepare for AWS Solutions Architect certification.” There you go. For free. Basically, YouTube – the most common place to search for cat pictures (or videos, actually).

# The overlap between DataOps and Data Engineering

40:44

Alexey

**[chuckles] Yeah. We have a few questions and two of these questions ask about the overlap between DataOps and data engineering. In your opinion, what is the overlap? Is there any overlap? And if there is, what is it?**

40:59

Tomasz

Between DataOps and data engineering? ML data engineer will be more operational, meaning he or she will be actually doing some pipelines, preparing some quality checks or whatever. A DataOps will honestly, gee… If I remember correctly my statistics from Google calendar when I was working as a DataOps in OLX, it was like 25 hours average per week on Zoom calls. So a data engineer will probably spend more time in PyCharm and DataOps will spend more time on Slack, Zoom, and email.

41:45

Alexey

**And these Zoom calls, what were you doing exactly? Helping others with problems, I guess?**

41:52

Tomasz

Exactly. Most often, honestly, live coding, designing some solutions. If you think about which domains DataOps touches, it's essentially past – meaning absorbing the technical debt. The present – meaning handling the users’ requests, like daily problems. And you're also thinking about the future –maybe you just prepare the summary of how the past month went, like “Okay. Most people have had problems with some service roles for the GitHub runners, because they have to go to 12 different repositories.” So now you're talking with the SREs “Guys, we might want to simplify that. Because there is, yet again, a problem with this process being too complicated. So maybe we should do something about this.”

Also, you're educating people. Imagine you've got newcomers to the company – someone has been onboarded by HR, but HR can only onboard you on what the teams are, what the structures are, who your boss will be, where you can ask for this or that. But the technical onboarding will probably be on the shoulders of DataOps. It's also your job to catch the newcomers to the company and make them comfortable. I don't know if that answers the question.

43:36

Alexey

**Maybe I'll try to summarize. Data engineers actually work in the PyCharm, VS Code, and so on. But DataOps mostly use Zoom and Slack and other things. [chuckles] That was the summary from my point of view. But I guess there is more to that. There is mostly support, also, at least with SREs that I see – my colleagues always have something open, like some sort of dashboard or something like Grafana or New Relic. So not only do they help people who come with ad hoc requests, but also they see “Okay, yeah, something is off here. Let me take a look at what’s inside.”**

44:19

Tomasz

Trying to be proactive, exactly.

# Suitable/useful skills for DataOps

44:23

Alexey

**You said when you were in this position that you spent a lot of time doing live coding, supporting, and education. From what I understood, this means that you already need to be a quite experienced person to work as a DataOps. Do you need to be a data engineer in the past to be successful in this role? What kind of background is actually useful for this role? Or not “useful” but maybe suitable?**

44:55

Tomasz

Any background in any data position.

44:58

Alexey

**Any?**

44:59

Tomasz

Any. Whether you were an analyst, data engineer, data scientist, whoever it will be useful. Why? Because you don't have to be an expert. You will be serving as a middleman between, let’s say, the platform team, the security team, the SRE team, *and* the users, meaning data analysts, engineers, scientists. The majority of the cases are literally not that hard. It’s more than enough to be able to read the log and try to figure out what is actually happening. You were working previously as a Java developer, if I remember correctly, so you definitely know how verbose the error messages and the logs are.

45:55

Alexey

**Very. But python is not different. It's sometimes even worse.**

46:00

Tomasz

Which basically means that if you're able to help people understand the logs, if you can help them understand how the cross-account roles will work in AWS – it's more than enough. You don't have to be a super expert, meaning, if you all of a sudden need to set up DNS records here and there, there *will be* some SRE who will be super glad that you're asking him or her about the technical questions, because SREs *like* technical questions. They’re typically not super thrilled to explain to someone over and over again how AssumeRole works. They are more thrilled about some really complex issue in the Kubernetes cluster that keeps them up and running. And you're taking that, let's say “unpleasant” or less favorite part from them, leaving them with the more technical side, which is, again, a win-win for both sides.

47:15

Alexey

**So how does AssumeRole work? [laughs] Okay. You don't have to explain this. I guess you did a fair amount of explanations already, right? [chuckles]**

47:30

Tomasz

A little bit, yes.

47:36

Alexey

**So you wanted to actually answer that? [chuckles]**

47:39

Tomasz

We can, but maybe people will find a better explanation on some YouTube channel than I would do now, probably. So I guess it's a good place to start.

# Minimal operational skills for DataOps

47:55

Alexey

**There are a few things I still want to ask you. We talked a bit about skills and there was actually a comment. You said there are 200 (or even more) services in AWS and you don't need to use all of them. Somebody commented that this is the Pareto principle applied to AWS services. But still, apart from the services that you mentioned, like IAM role, EC2, Ss3, EMR, we also have Docker, we also have Kubernetes. We also have CI/CD tools, we also have Prometheus, Grafana.**

**I haven't even started mentioning data-specific tools. These are all like general software engineering tools – general SRE/DevOps tools. So how do you actually start learning that? Do you have any suggestions? What are the minimal operational skills that I need to have in order to be able to work in this role?**

48:58

Tomasz

They have to be minimal. It has to be a really narrow set. A little bit of context. Some time ago, I read an excellent article. I believe it was called Good Enough Practices in Scientific Computing, or something like that. Someone went through all the best practices, they advised the best practices, or the best set of tools to someone, and then after some weeks checked if, in fact, that list and that training actually changed anything. The answer was – only partially. Because if you introduce someone to all the best-in-class, it might be complicated. If someone has never worked with any version control system, let's start with Dropbox. Honestly.

If you are still keeping passwords in some passwords.txt – password manager, please. YubiKey maybe. But start with a password manager honestly. Command line – if you set up the command line properly, then it will make your work and your life, basically, so much easier. And then if you will be working some DataOps, DevOps, SRE, whoever, he or she will also be super happy because they will not spend time trying to figure out what you have done in your command line. They will recognize the common stuff that they already know, meaning that you are somewhat experienced already, which will make it easier for them to diagnose the real problems here and there.

So I would say the minimal operation skills for everybody, whether this will be DataOps, or just a data analyst or whoever – version control system (probably Git), command line – to some extent, it will be enough to just know how to move between directories, how to grab something, how to cut something, how to assume the role in the Adobe CLI. Pretty much it. Plus password manager. Plus, as I said, IAM roles, which essentially means IAM. Why I'm stressing the IAM part over and over again, because honestly 90% of the errors are about “access denied”. Being able to run AWS STS get-caller-identity to know which role I'm currently in is super powerful – and super simple at the same time. Just drawing the different like, “Okay, *this* role can be assumed from *that* role. And *that* role can be from *that* role.” Just writing that down on some piece of paper, creating some dots [audio cuts out]

52:11

Alexey

**I think we've lost you again. Or maybe it's something with me? I don't know. I hope it's not me. Yeah, I think Tomasz also froze on YouTube. Hope he recovers the connection soon. We actually lost Tomasz, so I hope he will be able to rejoin us now. Yeah, he's gone, indeed. So I'm wondering how to keep you entertained while he's joining.**

53:11

Alexey

**Adonis, you mentioned that DataOps sounds kind of like a manager for data and this is exactly what I was thinking about. It sounds very similar to a management role – all this education, all these calls in Zoom, figuring out what the process should look like, all this support – it does look like management.**

53:39

Alexey

**Okay, now I am getting... I'm starting to worry a little bit. I hope the connection will be back with Tomasz. I think it's the first time that actually happened in the stream. Okay, you're back. Are you back?**

54:04

Tomasz

Okay, phew. I just created a hotspot from my telephone. Sorry for that.

54:10

Alexey

**[chuckles] Yeah. Internet. Live. Happens. Okay. Yeah.**

54:14

Alexey

**You were talking about AssumeRole – how to assume roles and then how you can just draw different roles and how one can assume another and how it immensely helps to figure out what the problem is.**

54:29

Tomasz

Okay, so actually, all the contents stayed. [chuckles]

# Similarities between DataOps and Data Science Managers

54:37

Alexey

**While you were away, I was trying to keep people on the call entertained. One Adonis mentioned is that what we talked about largely sounded like a data management role – all these Zoom calls, all this support in Slack, all this trying to live code with somebody – from what I see managers also often do it, especially with somebody who's, let's say, less senior. They often do it with juniors, with maybe middle-level people, all this kind of work. So what is the difference between a DataOps person and a data manager person – somebody who is mentoring people?**

55:22

Tomasz

That will be quite simple to explain, because usually data managers have their own team – a fixed set of like five or six or ten or whatever – of some folks. And as DataOps, you are working across different teams, across different business units. You are observing some Slack channel, like “data support\_something” and that will be pretty much for everybody. You will definitely have some splits if you are not the only one doing DataOps in the company. Someone will take the requests from one business unit and someone will take them from other business units. But essentially you will be working with a larger number of people, not only the fixed set of five or six. A data manager will also do one-on-ones, promotion plans, plan some sprints, and stuff like that. Whereas DataOps works with multiple different teams – but also with data managers, definitely yes.

56:44

Alexey

**Also maybe data managers – at least a typical data manager – might not have kubectl installed, they might not have Kubernetes access configured. They might not be able to actually log into the Kubernetes cluster and check what logs are there and what could be happening there. But a DataOps person *will* probably have these things, right?**

57:12

Tomasz

Probably, yes. But also, fun fact, Kubernetes is not that present in the data domain. If you are in the software engineering side, and you are a DevOps, not DataOps, definitely Kubernetes will be your bread and butter – every single thing will be on that platform, let's say, because what are you doing, essentially? Some frontend APIs, which is what Kubernetes is suited for. If you're in the data domain, you'll get a ton of batch jobs, which is not necessarily the first use case for Kubernetes. It will be most probably something like ECS, AWS batch, or GitLab, Jenkins – that sort of stuff.

58:05

Alexey

**I think you can still run Kubernetes jobs, but it's not the first choice, right? Sometimes there are some services that we can use from AWS or other cloud providers that [cross-talk]**

58:16

Tomasz

With Kubeflow, obviously, you can. But, as you said, it won't be the first choice. I guess.

# Tomasz’s interesting projects

58:26

Alexey

**Okay, we have a few questions. One of the questions is, “What was your most interesting project and why?”**

58:38

Tomasz

Well, the one I spent the most time on – was migrating a lot of workloads (almost 600) from all Jenkins servers to GitLab CI. It wasn't the most interesting because of the migration – because migrations are, honestly, pretty boring. What was interesting was working with pretty much the whole company at the same time on one project. Everybody and their mothers had some ETL job somewhere on some Jenkins instance. So that was the fun part, working with pretty much everybody. Maybe not the most interesting, but funny, was with debugging, actually. SREs plus contractors plus whoever were trying to move some Kinesis readers from an old EC2 machine to Kubernetes. Stream processing is a perfect use case for Kubernetes. Okay, fine.

I guess the count of people who were into that debugging process finally reached like nine or ten – super experienced guys, honestly. They just said “Okay, it's Dockerized. We got the Helm chart. Everything is set. Service accounts are properly done and deployed and yada, yada, yada.” Yet, the application just starts and immediately dies – out of nowhere. Then I joined that effort and one of the first questions I asked was like, “Hey, guys. Do we actually know that the library versions on that EC2 machine inside that container are actually the same as the ones we have there?” Someone was like, “Okay, you are a junior in the operation space. You probably don't know how Docker actually works, yada, yada, yada.” I was like, “Okay, maybe I don't.” But then we scanned how the Docker file was actually created. It was fetching the requirements.txt and the versions weren't specified there. So all of the sudden, when we just packed the library version, the problematic one was Psychopg – the PostgreS driver. The whole fix was like four characters. It took a quarter.

61:27

Alexey

**Psychopg relies on a binary… There is some binary code in Python, where you don't see any stack traces, it just dies and kills the entire container. Right? Sounds like fun.**

61:43

Tomasz

The code was prepared to work with version 2.7-something – without the version being specified, when folks try to run that stuff on Kubernetes, it fetched the latest version. And the API changed. This is the perfect use case. I tell everybody when they ask questions like “Why do we have to use fixed versions? Why can’t we just use any version, like the latest?” That's exactly why.

62:23

Alexey

**That's a good story. Do you have a couple of more minutes?**

62:27

Tomasz

Sure.

# Confidence in results and avoiding going too deep with edge cases

62:28

Alexey

**Okay. Last question for today. At the beginning, you told us a story when you worked in analytics and somebody from management asked you how confident you were in the results. So how do you usually answer this question?**

62:50

Tomasz

That “I'm as comfy as we get money to check all the edge cases.” Because the same manager who asked me that question was the same, one that healed me totally from checking every single possible edge case and error. If you are working in basic research, (basic is the name, but it implies that it might be simple – it's not. It's fundamental research in academia) then you basically are receiving the CSV file and that's it. If you are working in a company, data is flowing constantly. So tracking changes, policy changes, whatever changes, pipeline changes, schema changes, like everything changes constantly and failure is the only constant. As Werner Vogels said “Everything fails all the time.” Vogels is currently the CTO of AWS, so he probably knows what he was talking about.

I remember when I was checking some clickstream data and I was looking for errors, like “Before presenting the results, maybe let's check if all the things are set properly.” While I was doing that, my manager came in and asked, “Okay, are you done?” “Almost. I still get some edge cases in 10,000 records.” And he asked “Out of how many?” I was Like “Eh…400 million?” [chuckles] So he was like, “Okay, so you were checking 10,000 records out of 400 million and you're spending time on that. Congratulations.” Okay, that's just… it won't be perfect – ever.

But it’s definitely a wise idea to, for instance, if you are working with Airflow and you see that your pipelines are all green and so on. What does it actually mean? That the records were inserted into the given table *or* that the network didn't fail as it did for me today. The answer might be sometimes surprising. Okay, zero records inserted. Jobs are still green. So that kind of stuff might be checked before presenting some extraordinary results to leadership. Extraordinary claims require extraordinary proof.

# Conclusion

65:41

Alexey

**That actually happened to me. Green jobs in Airflow with zero records inserted. [chuckles] I guess that everyone had to experience this. [laughs] Okay. Now it's time to wrap up. Before we finish, maybe you forgot to mention something and you want to bring it up?**

66:07

Tomasz

Paraphrasing what you said at the very beginning of our talk – only one half of the people watching are also subscribed. So if you are in that particular group, don't forget to smash that like button and subscribe. [chuckles] I always wanted to say that, honestly. [laughs] That was a super, super, super pleasure to be here. Thank you for the invitation. And thank you all for being here with us this Friday evening.

66:49

Alexey

**Yeah, plus one to everything you said. [chuckles] So we should be finishing. Thanks, Tomasz, for joining us today. Thanks, everyone, for joining us today as well, for asking questions. I wish everyone to have a great weekend.**