A.8 Expert H

Interviewer
OK, so I identified 6 decisions. The first one is more conceptual, but it's also really important in the context of the other decisions. So if I choose for a specific data product type, this also affects all the other decisions. So that's why I identified this as an architectural design decision as well. The next one is in terms of migration. Do we want to migrate Umm. from a legacy architecture or do we want to do some Greenfield development? Athe same time, we can think about the infrastructure layer around the data products. So how does the data product interact with other data products, self-serve platform, the management layer and the consumers of course. Mm-hmm. After that we can deep dive more into the architectural design of the data product itself. So what's going on inside the data product? Umm. We can think about the data product interface / contract. So, uh, what kind of ports do we expect on the perimeter of our data product? Umm. And the last decision is on deployment. So how do we want to deploy all these things in the cloud for example. So if we start with the first one, it's a bit more extensive. But we can focus on these three concepts only perhaps. So these three options I've identified and these on the right, we see more some kind of variance and it's more detailed on the right side, but we should think about the three options that can answer our question over here so that the decision is what type of data product can be developed. And I identified 3 main options so you can expose your data product as raw data. You can expose your data product as derived data. So some small transformations are applied to the data set. Or you can expose your data product as an algorithm and with an algorithm I mean a power BI tool. For example, an optimization based decision support system or an AI / machine learning model. Umm. So if identified, these three decisions, I don't know if you agree with these, or if you think there should be more options or....

Expert H
At first, my perspective is very much a usage perspective coming from an analytics background. Umm. Uh, my old department's primary need was consuming data products. That said, we also identified that more and more we will be delivering data products ourselves as well because we're really trying to get analytics operationalized, which means you have to give back as well basically. Yep. In earlier discussions I've had with colleagues, we mainly identified top two data product varieties. Where we viewed this from the perspective of an end user. So as an analytics persona, I will want to explore if a domain has valuable data for me. The bottleneck that we've seen in the past in ingesting or consuming data is that it's in a non data mesh world. It takes very long to get your data properly curated via the property channels and then often the conclusion is even after the very short analysis you can already say this data is not useful for us for whatever reason and maybe a model just doesn't converge or whatever. Mm-hmm. So if in the data mesh world we would only accept these nicely productized data products. Then we would be defeating our own purpose because the whole idea for me is that you can go through quickly iterations. So we sort of defined a ladder system so the the bottom would be raw data and there should always be an acceptable first step. But explicitly as a first step. The data mesh thinking should not prohibit a source domain from giving me a one time dump for example. And if that one time dump is sufficient and it can be productized and then it sort of by definition it becomes the live data. Then you put a sticker on it. Then you say OK, this is now a proper data delivery. If the source domain can sustain that delivery. We've had some talks also about is an algorithm a data product? Mm-hmm. We're not there yet, I think. So. Well, for us, the thinking hasn't evolved far enough to say and does that fit? I don't think it does, because, uh, the way we are looking at, OK. but then maybe that's a regulatory thing. The way we are looking at algorithms is we have quite different requirements for those. So we

need to check for fairness and bias. And of course Hmm. you can say a data set, you also need to check for bias, but our customer base inherently is biased because Dutch society is biased, so we've got customers everywhere. So if I make a data set of all customers, it might end in overrepresenting certain demographics and there's nothing we can do about that. Yeah, we can try to get certain types of customers, but I doubt that would be a good idea. Yeah. Well, if we have an algorithm, let's say uh, an algorithm predicting whether the customer will default on their bills, we need to be very sure that there is no bias in there. So data can inherent bias and it's totally fine. But the algorithm generally shouldn't. So we've got a whole different set of demands actually for algorithms. Yeah. Which is why currently we don't feel an algorithm as a data product, but we see it as something. So the output of an algorithm can be a data product, but the algorithm itself not so much in our view.

Interviewer
So you mean there is some kind of reporting layer on top of a data product? That can be a dashboard.

Expert H
It could be, I mean we could do a once-a-day run. For example, a model predicting which customers by default. I add the output of that model as a new data set and that could be a data product but the algorithm itself not so much. I don't see the same governing patterns emerge there basically.

Interviewer
OK, alright, interesting. And if we quickly go on to the next one, because this is really focused on migrating or doing Greenfield development and in terms of migration I identified four different patterns. I don't know if you have an have an idea about Strangler fig

Expert H
No, I have to say I'm not too deeply involved in the specific frameworks so.

Interviewer
Uh, no worries. I can easily explain that one strangler fig is like decomposing your monolithic architecture and at the same time building up your data mesh architecture. Yeah, you choose a service from a monolithic architecture. You extract that and you develop a data mesh architecture on top of it. So you are doing two things at the same time while you're monolithic architecture shrinks and the data mesh architecture here is building itself up.

Expert H
Yeah, we we call this slicing the elephant. So it's just different name for the same,.

Interviewer
Yeah. And also data management and the others I think you probably know them. CQRS is about segregating your read and your write function to make sure that if something is being read that nothing changes in the data product itself. Yeah. So we are very much at the start of this journey, if you will.

Expert H
OK. I'm also not too involved in like how domains are being on boarded. The general thinking is OK, we have got central reporting or multiple sensible reporting databases and the idea is that we've got a data office going over 200 people, some of them are building platform capa-

bilities for the future. Some of them are maintaining what's currently there and some of them are onboarding new data flows every day. This last capability that will need to be distributed throughout the organization so IT teams will be doing the data productization. For now is that fundamentally nothing much should change. I mean that what's already there is good. It's just not in a in a sustainable framework. So what we would ideally want is to sort of distribute the ETL jobs that are currently there. Bring them back to the IT maybe have them tweak it a bit. But in principle, this is the Strangler Fig methodology. Yeah. Where we say, OK, the one open question that we do have is how identifiers will be handled. Because in the current ETL jobs multiple transformations might happen and so let's say a call center record might be attributed to the customer contact number well in reality or a call record is attributed to a phone number because you're calling from a phone and then have multiple transformations are needed. But then this new data mesh architecture, the call center will not necessarily know if I call, they will not know my contract number. They will only see my phone number or maybe some other identifying I give them.

Interviewer
So it's more like starting from scratch. And not really migrating or am I misinterpreting now?

Expert H
No in my view, multiple transformations are happening on the raw or raw-ish data. That's the source domain which is currently delivering. The question then is how many steps do you need to remove from the transformation to make a data mesh? Because the way we are looking at data mesh is every domain that we need has a curated list somewhere that says these are all the possible identifiers within our company. And if you as a domain have one of these identifiers you should always have one of these identifiers in your toolbox because otherwise your domain list is incomplete. And so in the example of the call center, maybe in the current flow there are three sequential transformations being done in order to get from a phone number to a contract number. Those three will need to be removed, but in my mind it's just that you've got this whole ETL pipeline doing and maybe you're subtracting the start and the end time to determine how long the call last. That's still perfectly valid. So that's Yeah. something you need to bring back to the domain. But these last three steps, transformations should not no longer need. So you need to cut those off the pipeline basically. Umm So it's during this migration. I think every pipeline will need to be evaluated if it's still valid basically.

Interviewer
Yeah, it also sounds a bit like master data management because we have a central customer and we can think about subscriptions, call duration and yeah, those are completely different things. But the same customer can perform these and if we have a master data customer, then there's some kind of central authority that's keeping track of all these different data products. So it's really like that master data management where we have a central entity the customer. This one has a global unique identify identifier. So that everything can be related can be back to this customer. Yeah. So for us, many attempts have been made towards master data management and equally many have failed basically.

Expert H
OK. Yeah, it's quite hard, yeah. I was working customer profile capability basically in the ideal world where you have nice master data management, you can uniquely identify your customer. You can uniquely reason about them. We've come to believe that that's not possible because there are too many ways to look at a customer that are more or less mutually exclusive, and it also concludes us from giving a single master identifier. To give an example there. I live

together with my girlfriend. Umm. So we've got two people in one household. I've got a mobile phone subscription and the internet subscription is mine, but she's got a mobile phone subscription of her own. So who is the customer here? Is it our households comprising? No. Yeah. That'd be fixed internet connection and to shop customers. Is it me? As the owner of the account, if though my girlfriend isn't under there, but she is linked to me because you get a discount if there are multiple people living in the same household. So that's one aspect there. How do you identify personas in Mm-hmm. a household and becomes even more complex in the case where the children of elderly people will be maintaining the contracts of their parents. So who is then the customer? Is it the person living in the house where the internet is being delivered or is it the person administering the bill? Yeah. So we we are now in the process of decentralizing that and that comes with its own difficulties because it becomes more difficult to reason about relationships, but it also makes it more explicit because we're actually saying for every request you make, you do need to think about what is it that you mean, because we know we don't have a customer. Yeah. Basically.

Interviewer
Yeah, this also relates to the infrastructure. So if we think about what's going on outside the data product. So in the entire data mesh, we can think about the schema registry and what the schema registry actually does is when we use Kafka, for example, as an event streaming backbone, a message is published into the Kafka topic, several other data products are subscribed to this product to this topic. And of course, a data product catalog, which contains all the metadata from the entire mesh And if a change event through the Kafka backbone is sent to the central data product catalog it's often not really user friendly. So we need a schema registry to make sure that such events are made user friendly to read. You know? So that's what this schema registry actually does. Some practitioners say that we should have a shared storage because otherwise there would be so many different individual storage accounts, which can be really costly. Umm. So perhaps it's interesting to talk about that because other practitioners really evangelize the concept of internal storages and that each data product should be as autonomous as possible. It. API invocation. So we have three kinds of APIs, REST API's, graphQL and gRPC and there are again some variant. So these can be rest API, graphQL et cetera. But these are related to the shared storage. We need a SQL access point because most of the data analysts nowadays are really familiar with SQL and they prefer this kind of access method. And on the bottom, we see more like no functional requirements. Umm. So of course we need security controls. We can implement an in-memory cache and what I mean met with a query catalog is that there can be some kind of manual for the data analyst with all the possible queries in it, so that the data analyst knows how to access the data. Yeah, this is actually all I have in the infrastructure layer around the data products.

Expert H
Yeah. So I am me not being sort of centrally responsible for the whole data mesh architecture. But can say something about what we're looking at from an analytics perspective and also explicitly from the customer project. Yeah, sure. Yeah. One of the things I've seen is that the creative catalog we have already in a very basic way, I mean, but it's about functionality, it's not about implementation for me, but this is something that glue naturally for us as well. A colleagues were just backing up queries that we're doing and putting them in a central place so that you can see and not necessarily how to create the table, but which tables are interesting in relation to each other. Umm. So I believe the query catalog is on the backlog of our data catalog so that you can sort of say, yeah, OK, we've got this data product and it's quite often used in conjunction with tables A,B and C. Umm. For our SQL access endpoints, the capability that when I'm working on the aim is that we have a components that we can connect to

various streaming topics. And in which we can combine data across domains. So what we have there is we've got a capability, it's squeezable, as you might say. So we need something that can both be queried and can publish data. That can connect to a key ring functionality. So we do have a key ring in development and more and more keys are being added to that. The idea of the key ring is that it will be decentrally fat, so an authoritative domain. Let's say I'm responsible for the relationship between a customers contract number and it's mobile phone number. Then I'm the authority and I'm the only one supposed to push that relationship, so it's just keep there basically. Mm-hmm. And of course I need to then list that one as well in the data product catalog, because otherwise people don't know the specifics of that relation. Yeah. And the idea is that combination will allow us to both observe on the events team in backbone, but also the sequel on the event streaming backbone, and that's the ambition there. OK. And that does come with built in caching because they've got more complex creators on Kafka. You would generally not want to do so you need to pre cache them. In terms of shared storage, are thinking is that? In some cases, you might want to Daisy chain domains, let's say. Uh, we don't think that customer profile is a domain, but let's say we do think that, that we do believe in that pipe dream of our unified customer profile which I just explained. We don't believe there is, but let's say we do and then we would have a ton of source domains, we would have. this custom source domain and that would have its internal storage. So that's actually where you're caching happens because they will then expose the single unified view customer profile to the rest of the organization. So in that sense. Uh, I would argue with shared storage doesn't exist. It's always in a domain and so it's private storage and that domain. OK. Umm. Uh determines what data they will expose again.

Interviewer
Yeah. All right. And some practitioners also mentioned that I was perhaps too focused on event streaming and that data products can also do batch processing. So if the data is not coming in like real time that we can also for example think about blob storage and a query table, things like that. What perspective from that or do you only do event streaming in your data mesh?

Expert H
No, I think we will all keep on doing batch processing and that's a largely a cost and performance question as well. If you in, in discussion solution that we envision, it's quite likely that we'll be memory intensive and that's of course Yeah. way more expensive than just having a table with all your customers in there. So I don't think data mesh will be the end of of that storage. I think it will be very hard to get rid of batch storage. I've worked on projects where there's like terabytes of data per day being generated, you're just not going to process that in a streaming pipeline. At least if you if you want to train a model, you don't want to pull that all through a streaming pipeline.

Interviewer
That will be that it wouldn't be cost effective. Do you have a few minutes extra bad way? Because we're almost at the end of our session.

Expert H
I'm I'm afraid I don't know if I've got

Interviewer
Several meetings, OK, no worries. I will quickly go. I think those two are less important. So I really want to deep dive into this one for the last three minutes, 2 minutes. This is what's going

on inside the data product itself. So in Umm. the case of event streaming, of course we need a change data capture. We can have an immutable change audit log. So if something changes in the internal storage of a data product. All the immutable logs of the other data products are being notified and it's like an append only mechanism. So the change event is appended in this log and it's immutable so it doesn't change. Umm. Uh, the internal storage we just discussed this already. Umm. Then each data product on its own should have a data catalog. Uh, there should be some kind of observation plane, a control plane to maintain governance and data onboarding. So this is like Apache Spark doing all the transformations, things like that.

Expert H
I would relate this to data platforms. In our vision. Basically, we see maybe two or three modalities arise and we've got a central streaming solution. So if as an IT team you are able to push data on that Kafka bus then all good you're done. Just make sure you comply to the schema registry and So what you do in your own solution that's not up to us. Yeah. OK. Then, but we will be providing data platforms. So if you want you can use staging area as well on that streaming bus and the same will apply to the batch processing. It just general ETL Mm-hmm. tooling. Basically if you' view Spark ETL tooling as well because it can do that. Umm. So. I would argue that, for example, a change audit log that might be something that you built sort of on top of your bus. You monitor the messages and if you see something change you flag it. Yeah. Yeah. So I don't tie these two data products then it becomes the responsibility of the underlying IT teams, at least that's the way we view that. And so we would rather that to data platforms and in that sense. Centrally developed and decentrally fed. So the data catalog is centrally developed and then it should be very easy via maybe two or three standard ways to put your data products in the data catalog. Those are genetic capabilities that we want to build on data platforms and in principle as an IT team, you are free to use them if you got a better solution that comes from your domain and by all means use that. I mean we're not here to tell people how to do their in domain job. Yeah, to an extent, of course, because it if everyone buys their own solutions then the licensing cost goes out of hand. So, but that's synchronous thing.

Interviewer
Thank you very much. I won't take any more of your time, but thank you so much for this interview and yeah, I will