Data engineering | Platforms | Blog

By Arif Wider

Published: November 16, 2020

The **data mesh paradigm** is a strong candidate to supersede the data lake as the dominant architectural pattern in data and analytics. Importantly, the data mesh mainly introduces a new organizational perspective and is independent of specific technologies. Its key idea is to apply domain-driven design and product thinking to the

Data warehouse, data lake, and the issues with centralized data ownership

In traditional business intelligence (BI), a centrally maintained data warehouse is the basis for many business decisions, e.g. by providing up-to-date reports that support those decisions. As big data technology has matured and with the growing popularity of data science, many companies invest in building a central data lake — sometimes to replace the data warehouse but more often in addition to the existing data warehouse. The main difference between the two approaches is when curation and modeling happens: with the data warehouse, data is already transformed at ingestion time to fit a certain application; **with the data lake, this transformation happens only at the time when the data is prepared for consumption**. The common theme to both approaches, however, is centralization. And it is this centralization that leads to recurring patterns of problems.

One of those patterns that I've seen again and again, is that of an overwhelmed and stressed-out central "data team". This team maintains the central data infrastructure, be it the data warehouse or the data lake. More importantly, however, this team is solely responsible for delivering data sets or reports to stakeholders, product teams and data scientists in a reliable and timely fashion. I am consciously calling this a data team, and not more specifically a data engineering or a data insights team, because it reflects the unclear mix of responsibilities that this team is often dealing with.

Consequently the members of this data team often find themselves in a tight spot. They spend a lot of their time "firefighting" and fixing issues that have been introduced by data producing teams, while being the recipient of frustration from data consuming stakeholders. That is particularly sad, because those team members are often the most most data-savvy individuals in the company, and it's common to see this prolonged stress result in reduced productivity, lower workplace satisfaction, and even higher employee attrition rates.

Now why can't such capable engineers solve this situation? The reason is that the problem is not a technological but an organizational one. One of the main issues is an unfortunate distribution of responsibilities to the parties involved.

One party — the data producers — has the domain expertise, i.e. they understand the meaning of the data and they can directly change the way the data is shaped; Another party — the data consumers — has the vested interest in the data, understands its business potential and can therefore clearly describe requirements about, inter alia,

data quality. The members of the data team fall between those two parties: they have the responsibility to deliver data reliably and with high quality, but they have neither the domain expertise nor the ability to have direct influence on the data creation. Additionally, they're not the ones who ultimately use the data to make decisions. That means that the interest, the responsibility, and the ability are distributed over three different parties, which leads to friction, frustration and misunderstanding.

Figure 1: Traditional approaches to data can create a disconnect between data owners and users

Data mesh: Decentralized domain ownership, shared infrastructure

Instead, as a goal state, data producers and data consumers should be working together as closely as possible. From an organizational perspective, the ideal situation is when the same team is both producing and consuming the same data, so that

interest, responsibility, and ability are combined in the same team. In practice, this is often not feasible, as a data producing team already owns so many responsibilities in their particular domain that they cannot fully own a data consuming application too. Thus, splitting those roles into two teams that directly communicate without middlemen, is already a big step forward. The goal of a data producing team should be to provide their data in such a way that others can get value out of that data without requiring detailed domain knowledge, i.e. data producers should hide “implementation details.” Such a data producing team, of course, can also be in a data consuming position at the same time. There are consumer-oriented data domains that are complex enough to justify a whole team of domain experts but who themselves consume data from a source-aligned data domain.

From a purely organizational perspective, such a structure of bilateral relations between data producers and data consumers, where all the responsibility and expertise about one particular domain falls into one team, creates less friction, increases ownership and thereby scales better with higher quality. If we accept this premise, then why is it that the pattern of the overwhelmed central data team with centralized data ownership is so common? In my experience, there are three predominant concerns that drive mostly drive the ill-fated centralization of data ownership in enterprises:

1. The fear of not having enough data engineering and data science experts to equip several teams. Instead, there is the hope that a central team can make use of those scarce experts more efficiently and can support teams more equally.
2. The fear of losing control on data quality, e.g. because it seems hard to establish global standards with decentralized ownership.
3. The fear of duplicate infrastructure investments, because similar infrastructure pieces such as pipelines, services, and storage need to be created and maintained by every team again and again.

Often, the lack of a conceptual separation between centralized data ownership and centralized data infrastructure prevents realizing the advantages of decentralized data ownership. In fact, in all three cases described above the creation of a shared data infrastructure platform that focuses on self service tools can help to alleviate such concerns. However, it is crucial to keep this data infrastructure platform out of centralized domain data ownership by focusing on domain-agnostic self-service tools. Otherwise, there is a high risk that the data infrastructure platform quickly becomes the central data platform with centralized data ownership that we wanted to get rid of in the first place. Finally, this approach needs to be combined with establishing product thinking for data to ensure that the decentralization of ownership is sustainable.

Figure 2: A domain-agnostic data platform

Domain-agnostic data infrastructure and product thinking

How do you recognize that a data infrastructure platform is truly domain-agnostic and self-service focused though? One indicator is that a team that wants to share their domain expertise can start providing their domain data without having to contact members of the team that maintains the data infrastructure platform. That means, those data infrastructure platform developers must never need detailed domain knowledge to do their job.

On the flipside, the platform has to provide all the tools that enable domain data experts to manage the full lifecycle of their data offering without requiring them to have deep data engineering expertise. That means they must be enabled to create a data

domain product, to describe it, to evolve it, to observe its usage, and also at some point to destroy it again.

Creating a self-service platform that provides such a level of enablement is a huge technological and product development challenge. However, at its core it is a traditional internal software product development endeavor that can be tackled by first implementing the most common use cases and only afterwards extending the platform capabilities gradually.

A crucial feature of the platform is however, that it supports multi-tenancy and supports a logical isolation between domains on top of a shared infrastructure. This way, duplicate infrastructure efforts can be avoided without pulling the infrastructure platform team into centralized data ownership. Such a domain-agnostic platform team scales much better because its members do not need to keep up with the ever changing domain-specific intricacies and requirements of all the business domains. Instead, this detailed domain expertise should be actively cultivated and maintained in those domain data teams. Therefore, if done with the right focus, a medium-sized team can suffice to develop and to maintain a shared data infrastructure platform sustainably.

Another big advantage of a shared self-service data infrastructure platform, next to avoiding duplication of effort, is about data governance and standardization. If it is more convenient for domain data teams to provide their data using the tools of the platform than by building their own infrastructure, it's easy to enforce certain standards through those platform tools. This way, standardization and — to a certain extent — governance is simply driven by convenience.

Thus, out of the three concerns about decentralized data ownership outlined above, only the one about data quality is left. Now, responsibility about data quality cannot be taken over by a centralized team in a scalable and sustainable way anyway. No single team can build up enough domain expertise about all different business domains to ensure in-depth data quality. And this is what data quality is about: it is not about some general guarantees about the shape of the data but about the concrete contents, semantics, and evolution of the data.

However, this challenge also isn't solved just by decentralizing the responsibility. For this, product thinking comes into play. Domain data teams need to be incentivized to provide their data with high quality in a reliable fashion, e.g. by matching their budget to the number and happiness of their data consumers. This way, a domain data team will try to promote the value of their data and will try to cater for the needs of their data consuming customers.

In conclusion there are three approaches that need to be established in order to get to a scalable and sustainable data landscape with decentralized data ownership:

1. The use of domain-driven design as the primary means to structure data and the assignment of full end-to-end ownership of a domain (or a subdomain) to one cross-functional team that gets the necessary support to fulfil that responsibility.
2. The application of product thinking to data in order to provide the right capabilities and set the right incentives for data to be provided in such a way as it is needed by its consumers.
3. Leveraging platform thinking by investing in the creation of a shared domain-agnostic self-service data infrastructure platform that does not take centralized data ownership but focuses on supporting and enabling the direct collaboration between data producers and data consumers.

Disclaimer: The statements and opinions expressed in this article are those of the author(s) and do not necessarily reflect the positions of Thoughtworks.

Related Blogs

Data engineering

The curse of the data lake monster

Learn more ›

Data engineering

Coding habits for data scientists

Learn more ›

Data strategy

How much can you trust your data?

Learn more

Keep up to date with our latest insights

Explore our content

Company

Insights

Site info

Connect with us