Data Mesh: Practices from a Single Domain Perspective

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Brian Mbogo

Fall 2022

Technical Project Team Members

Brian Mbogo

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Briana Morrison, Department of Computer Science

Data Mesh: Practices from a Single Domain Perspective

CS4991 Capstone Report, 2022
Brian Mbogo
Computer Science
The University of Virginia
School of Engineering and Applied Sciences
Charlottesville, Virginia USA
bpm4pkz@virginia.edu

Abstract
As modern businesses' data usage grows, so does their need for methods and technologies that preserve their agility for meeting business objectives in the face of the challenges posed by data. Pluralsight, a technology up-skilling and education platform uses a data mesh architecture to meet these modern data challenges. It ensures that each data endpoint team has the necessary data for its work serving independently defined internal and external customers and reduces duplication of effort when multiple teams are working with the same data. The analytics team in which I was embedded worked with two internal teams and the single idea of a large external customer. The team's preferred technology stack includes Apache Spark, cloud analytics infrastructure, a continuous integration/deployment platform and the necessary interface for company-wide data transfer. The result of the chosen technology stack is an efficient, largely formulaic development process for any given data product. Lastly, this report will discuss the ways in which team categorization can improve the ease with which data mesh is implemented and a way to approach systematic improvement of the data mesh architecture.

1. Introduction

Citing a survey from NewVantage Partners, Dehghani (2019) concludes: "[Companies] have been investing heavily in building enablers such as data and intelligence platforms. Despite increasing effort and investment in building such enabling platforms, the organizations find the results middling." As the nature of software-based companies evolves, they encounter increasing scalability challenges in their data architecture, including centralized or monolithic design, high coupling in the data pipeline, and siloed data ownership.

Monolithic data architecture poses a particularly difficult problem for companies with large amounts of data with varying sources and uses because monolithic data architecture is designed to be domain-agnostic. This design is sufficient and even good for companies with relatively simple data due to the lower overhead associated with identical sourcing and storage of the data. However, for companies with more varied sourcing and consumption, it interferes with other priorities and limits points of innovation.

Pipeline coupling is a result of the overall monolithic design, and its main effect is to limit the speed at which changes can be made to the data. Because the data architecture is decomposed into pipelines of data defined by ingestion, processing, and serving steps, new features involve changing the entire pipeline, as all data requires all of the functions performed by the pipeline.

The existence of siloed data teams in older data infrastructure leads to the isolation of data engineers from the consumers and producers, which can cause frustration in handling many different data while dealing with the uncertainty of their application to general business goals or specific domain use.

2. Related Works

Data architectures which preceded the data mesh are not particularly old because they arose from the still-nascent area of big data. Sawadogo and Darmont (2020) provide a detailed description of the pre-eminent data lake which explores difficulties in the standardization of the definition, principles and the implementation [1].

Priebe and Neumaier (2021) use a standard notation and framework which includes elements such as data integration & interoperability, data quality, and data security to compare the important elements of data warehouse, data fabric and data mesh [2]. Although their major focus is a systematic approach to the discussion of the variety in data architecture, Priebe and Neumaier offer a developmental link between data fabric and data mesh.

Data mesh architecture was first introduced by Dehghani in her 2019 article which reviewed problems presented by extant data architectures and the solutions that data mesh can offer [3]. Dehghani followed that seminal work with a 2020 article aimed at a high-level summary of the data mesh, including its underlying principles and a move towards a prescription for its logical architecture [4].

3. Proposed Design

The implementation of data mesh can be broken down along three axes: organization, team, and technology stack. The axis of the organization determines overall structure and team mission. The team determines how projects are partitioned and completed. The technology stack provides capabilities and constraints around which projects are completed.

3.1 Organizational Considerations

Because the data architecture inevitably deals with all of a company's data, it is, by necessity, instituted top-down. However, the implementation is decentralized, as its focus is on the distribution of responsibility across multiple domains [4]. At Pluralsight, the company, and especially the engineering department value team autonomy. Leaders reinforce this aspect of team operation by tying it to two of the company's core values, "champion the customer" and "own our outcomes."

The connection between team autonomy and customer-centrism is that teams are encouraged to voice concerns or opposition to wider initiatives in the company if they think those initiatives will prove directly detrimental to customers' interests or to the ability to meet customer needs based on trends they see. Of course, team autonomy comes with accountability, which leaders tout as the reason teams should be dedicated to putting forth the best work in their domains. The emphasis on accountability encourages teams to celebrate achievements, but also to quickly admit to shortcomings so they can be fixed quickly to maintain overall engineering momentum.

I suggest that all enterprises that would like to implement data mesh architecture begin with a cultural emphasis on team autonomy with at least the components of accountability and conflict resolution focused on company or mission outcomes.

3.2 Team Composition

In discussing modern technology products, Cagan (2017) emphasizes the importance of the product manager role [5]. Every engineering team at Pluralsight has a product manager, including the data team I worked with. This practice reflects the understanding of data as a product, an evolution from data fabric [2]. The product manager is the primary domain data product owner [3]. The domain data product developers are the software engineers. An additional feature of Pluralsight's engineering team is the rotation of the lead engineer position, which improves every engineer's feeling of ownership over the domain product.

Because of data mesh's focus on data as a product, enterprises implementing data mesh should do so within the product framework offered by Cagan and after the example offered by Pluralsight. For teams whose primary product is not data, the product manager should understand that the data is as much a product as the primary product. Data customers should therefore be treated with equal consideration.

I do not universally recommend the rotation of the lead engineer role. It is well-suited for organizations accustomed to organizational flux. It works well at Pluralsight because of the relative commonality of lateral mobility within and across departments. Because of the dynamic nature of startups, it would likely work in those settings, as well [5].

3.3 Technology Stack

My Pluralsight team's preferred technology stack includes Apache Spark, cloud analytics infrastructure, a platform for continuous integration/deployment and the necessary interface for company-wide data transfer. Cloud infrastructure and continuous integration/deployment are common parts of modern software development [7]. The need for company-wide data transfer in data mesh is articulated by Dehghani [2].

I recommend that enterprises implementing data mesh using a data stream or micro-batch processing tool that can handle data from different sources with cloud interfacing capabilities. Apache Spark is an open source tool with functionality separated into modules. Apache Spark offers the ability to process the variety and volume of data that can be produced in a data mesh environment [3, 6]. Companies also offer specialized tools such as Google Cloud DataFlow and Amazon Kinesis which are inherently cloud-based. There is a wide variety in continuous integration/deployment tools, but because the core functionality is the same, it should be chosen based on developer familiarity. I recommend Apache Kafka for company-wide data transfer because it is inherently stream-based and has a large community of users [8].

4. Anticipated Results

Company policy inherently affects the whole company. Based on this fact, the effect of implementing my proposed autonomy-focused cultural policies will be consistency in teams' ability to innovate across the organization [5]. Teams will be able to clearly identify their products and how they help other teams and customers, which I observed at Pluralsight.

Organizing teams according to my proposal will ensure efficiency in the development of data products, as all members will have clearly defined, but not siloed roles. These teams will be able to scale and deliver on the promises of a data-driven enterprise [3, 4]. Additionally, the limitation on direct access to data improves security [7].

The technology stack I am proposing centers around well-known, flexible technologies,

which offer an onboarding advantage relative to in-house solutions [6, 8]. Developers will be able to take advantage of existing testing and troubleshooting resources. A data mesh so implemented will allow companies to use data as an enabler.

5. Conclusion

This proposal gives data-oriented enterprises exact guidance and reasoning in their data mesh implementation. They will be able to use it as a starting point and adjust it to fit their specific uses. Implementation starts are the organization level and incorporates autonomous teams. Companies and teams will benefit from the increased opportunities for innovation and more user-friendly data.

6. Future Work

Future development in this area fits within either the implementation or development of the data mesh. The implementation could be further developed by the categorization of team compositions. Along with the study of different technology stacks, this would offer guidance on team composition that is based less on intuition. Further development of data mesh architecture itself could leverage the ArchiMate notation based on DAMA used by Priebe and Neumaier (2021). This approach would render weaknesses in data mesh, allowing them and the characteristics to which they are related to be improved systematically.

References

[1] Pegdwendé Sawadogo and Jérôme Darmont. 2020. On data lake architectures and metadata management. J Intell Inf Syst 56, (June 2020), 97–120. DOI: https://doi.org/10.1007/s10844-020-00608-7

[2] Torsten Priebe, Sebastian Neumaier, and Stefan Markus. 2021. Finding Your Way Through the Jungle of Big Data Architectures. In *Proceedings of 2021 IEEE International Conference on Big Data (Big Data)*, *5994-5996*. https://doi.org/10.1109/BigData52589.2021.9671862

[3] Zhamak Dehghani. 2019. How to move beyond a Monolithic Data Lake to a distributed data mesh. (May 2019). Retrieved September 18, 2022 from https://martinfowler.com/articles/data-monolith-to-mesh.html

[4] Zhamak Dehghani. 2020. Data mesh principles and logical architecture. (December 2020). Retrieved September 18, 2022 from https://martinfowler.com/articles/data-mesh-principles.html

[5] Marty Cagan. 2017. *INSPIRED: How to Create Tech Products Customers Love*. John Wiley & Sons, Hoboken, NJ.

[6] Apache Software Foundation. 2022. Spark Overview. Retrieved October 16, 2022 from https://spark.apache.org/docs/latest/

[7] Habib Ullah Khan, Farhad Ali and Shah Nazir. 2022. Systematic analysis of software development in cloud computing perceptions. *J Softw Evol Proc*, e2485, (June 2022). DOI: https://doi.org/10.1002/smr.2485

[8] Confluent. 2014. What is Kafka. Retrieved October 16, 2022 from https://www.confluent.io/what-is-apache-kafka/