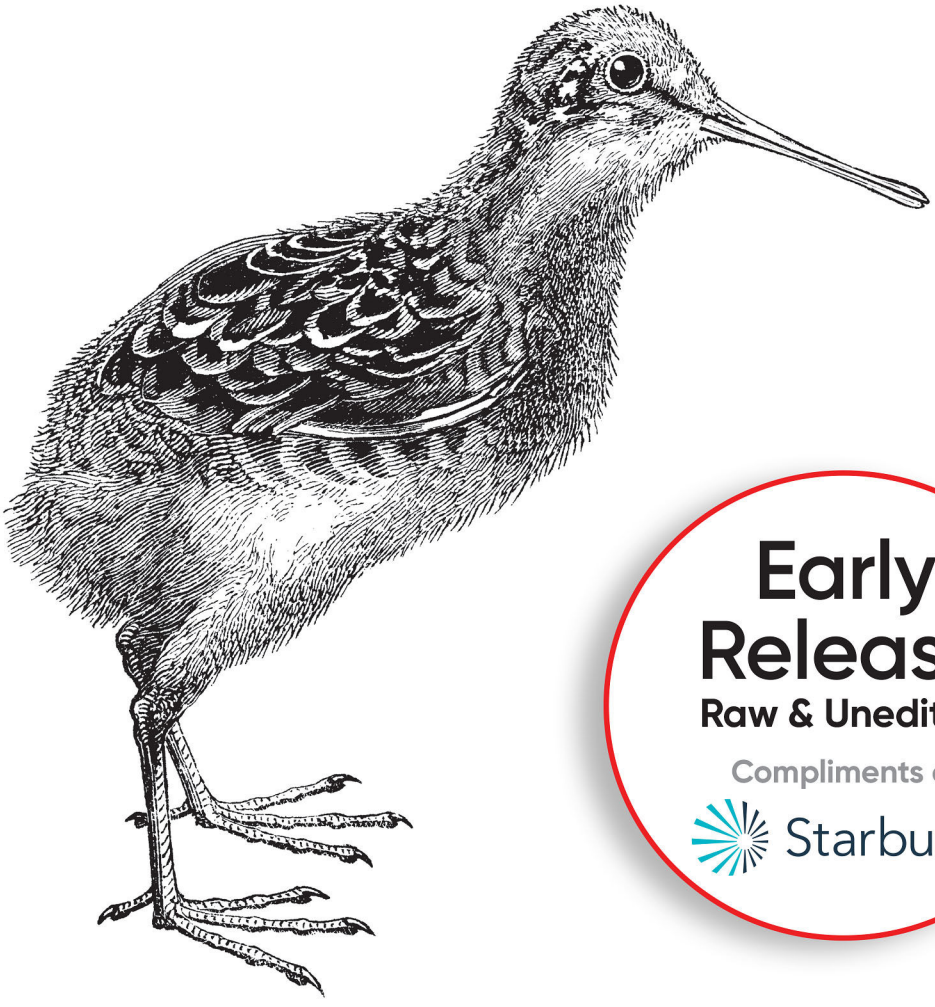
O'REILLY®

Data Mesh

Delivering Data-Driven Value at Scale



Early Release
Raw & Unedited

Compliments of

Zhamak Dehghani

Data Mesh

Delivering Data-Driven Value at Scale

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

Zhamak Dehghani

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

Data Mesh
by Zhamak Dehghani

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Melissa Duffield
Development Editor: Gary O'Brien
Production Editor: Beth Kelly
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Kate Dullea

December 2021: First Edition

Revision History for the Early Release
2021-06-18: First Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781492092391* for release details.

This work is part of a collaboration between O'Reilly and Starburst Data. See our statement of editorial independence.

978-1-492-09232-2

[FILL IN]

Table of Contents

PART I
Why Data Mesh?

By doubting we are led to question, by questioning we arrive at the truth.
—Peter Abelard

Data Mesh is a new approach in sourcing, managing, and accessing data for analytical use cases at *scale*. Let's call this class of data, analytical data. Analytical data is used for predictive or diagnostic use cases. It is the foundation for visualizations and reports that provide insights into the business. It is used to train machine learning models that augment the business with data-driven intelligence. It is the essential ingredient for organizations to move from intuition and gut-driven decision-making to taking actions based on observations and data-driven predictions. Analytical data is what powers the software and technology of the future. It enables a technology shift from human-designed rule-based algorithms to data-driven machine-learned models. Analytical data is becoming an increasingly critical component of the technology landscape.

The phrase data in this writeup, if not qualified, refers to analytical data. Analytical data serves reporting and machine learning training use cases.

Data Mesh calls for a fundamental shift in our assumptions, architecture, technical solutions, and social structure of our organizations, in how we manage, use, and own analytical data.

- *Organizationally*, it shifts from centralized ownership of the data by specialists who run the data platform technologies, to a *decentralized data ownership model* pushing ownership and accountability of the data back to the business domains where it originates from or is used.
- *Architecturally*, it shifts from collecting data into monolithic warehouses and lakes to connecting data through a *distributed mesh* of data accessed through standardized protocols.
- *Technologically*, it shifts from technology solutions that treat data as a by-product of running pipeline code, to solutions that treat data and code that maintains it as one lively autonomous unit.
- *Operationally*, it shifts data governance from a top-down centralized operational model with human interventions, to a *federated model* with computational policies embedded in the nodes on the mesh.
- *Principally*, it shifts our value system from data as an asset to be collected, to *data as a product* to serve and delight the users.

Figure I-1 summarizes the dimensions of shift that Data Mesh introduces.

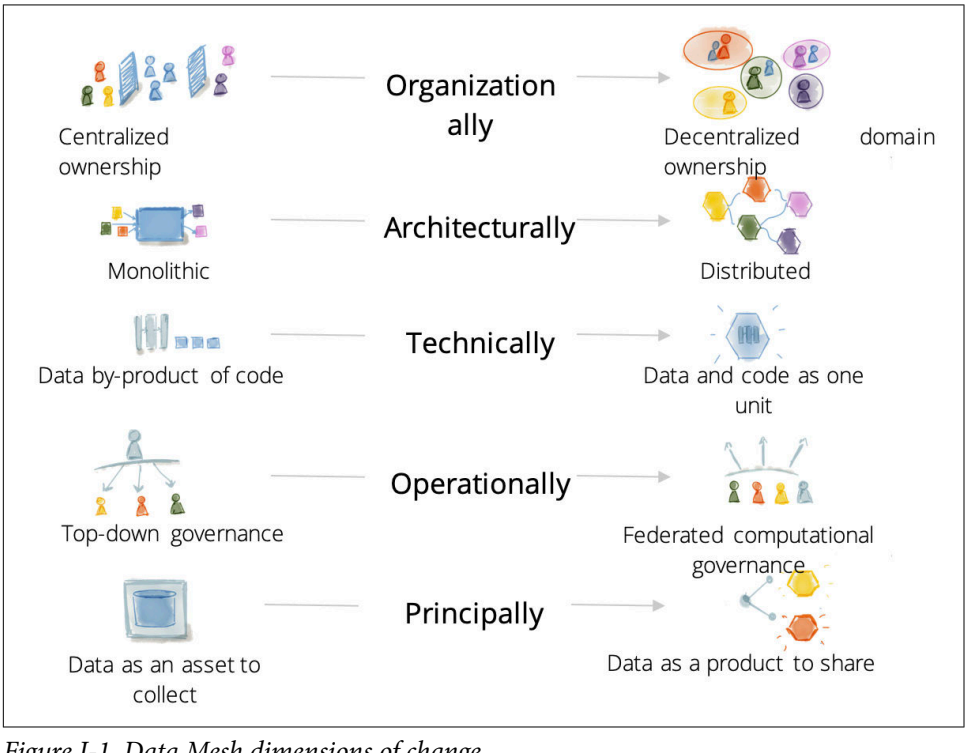


Figure I-1. Data Mesh dimensions of change

This is quite a shift, and an uncomfortable one. So why do we need it, and why now? I will answer this question in part I of the book.

In the first chapter we look at the macro drivers, the current realities that have pushed us to a tipping point, where our past evolutionary approaches no longer serve us. The second chapter introduces the core outcomes that Data Mesh achieves through its shifts in approach. And in the last chapter of part I, we briefly review the history of analytical data management architecture and why what got us here will no longer take us to the future.

Let's set the stage for Data Mesh.

CHAPTER 1
The Inflection Point

A note for Early Release readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the 1st chapter of the final book.

If you have comments about how we might improve the content and/or examples in this book, or if you notice missing material within this chapter, please reach out to the editor at gobrien@oreilly.com.

> *A strategic inflection point is a time in the life of business when its fundamentals are about to change. That change can mean an opportunity to rise to new heights. But it may just as likely signal the beginning of the end.*
>
> —Andrew S. Grove, CEO of Intel Corporation

Data Mesh is what comes after an inflection point, shifting our approach, attitude, and technology toward data. Mathematically, an inflection point is a magic moment at which a curve stops bending one way and starts curving in the other direction. It's a point that the old picture dissolves, giving way to a new one.

This won't be the first or the last inflection point in the evolution of data management. However, it is the one that is most relevant now. There are drivers and empirical signals that point us in a new direction. I personally found myself at this turning point in 2018. When many of our clients at ThoughtWorks, a global technology consultancy, simultaneously were seeking for a new data architecture that could respond to the scale, complexity, and aspirations of their business. After reading this chapter, I hope that you too arrive at this critical point, where you feel the urge for change, to

wash away some of the fundamental assumptions made about data, and imagine something new.

Figure 1-1 is a simplistic demonstration of the inflection point in question. The x-axis represents the macro drivers that have pushed us to this inflection point. They include an ever-increasing business complexity combined with uncertainty, proliferation of data expectations and use cases, and the availability of data from ubiquitous sources. On the y-axis we see the impact of these drivers on business agility, ability to get value from data and resilience to change. In the center is the inflection point, where we have a choice to make. To continue with our existing approach and, at best, reach a plateau of impact, or take the Data Mesh approach with the promise of reaching new heights in the agility of acting on data, immunity to rapid change, and being able to get value from data at a larger scale. Part II of this book will go through the details of what the Data Mesh approach entails.

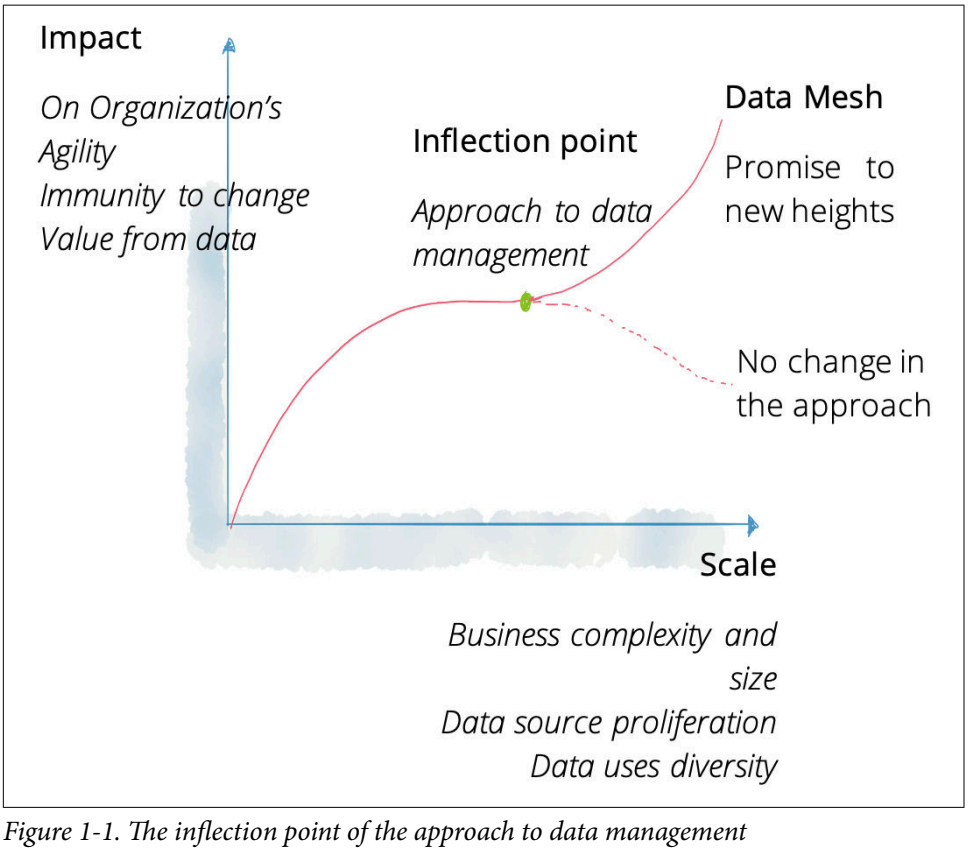


Figure 1-1. The inflection point of the approach to data management

In this chapter, I share today's data landscape realities that are the main drivers for Data Mesh.

Great Expectations of Data

One of the perks of being a technology consultant is traveling through many industries and companies, and getting to know their deepest desires and challenges. Through this journey, one thing is evident: being a data-driven organization remains one of the top strategic goals of executives.

Here are a few examples, all truly inspiring:

> *Our mission at Intuit is to power prosperity around the world as an AI-driven expert platform company, by addressing the most pressing financial challenges facing our consumer, small business and self-employed customers.*
>
> —Financial SaaS Company

> Our mission is to improve every single member's experience at every single touchpoint with our organization through data and AI.
>
> —Healthcare provider and payer company

> By People, For People: We incorporate human oversight into AI. With people at the core, AI can enhance the workforce, expand capability and benefit society as a whole.
>
> —Telco

No matter the industry or the company, it's loud and clear, we want to become intelligently empowered[1] to:

- provide the best customer experience based on data and hyper-personalization
- reduce operational costs and time through data-driven optimizations
- empower employees to make better decisions with trend analysis and business intelligence

All of these scenarios require data--a high volume of diverse, up-to-date, and truthful data that can, in turn, fuel the underlying analytics and machine learning models.

A decade ago, many companies' data aspirations were mainly limited to business intelligence (BI). They wanted the ability to generate reports and dashboards to manage operational risk, respond to compliance, and ultimately make business decisions based on the facts, on a slower cadence. In addition to BI, classical statistical learning has been used in pockets of business operations in the industries such as insurance, healthcare, and finance. These early use cases, delivered highly specialized teams, have been the most influential drivers for many past data management approaches.

1 Christoph Windheuser, What is Intelligent Empowerment?, (ThoughtWorks, 2018).

Today, data aspirations have evolved beyond business intelligence to every aspect of an organization, using machine learning in the design of the products, such as automated assistants, in the design of our services and experience of our customers, such as personalized healthcare, and streamlining operations such as optimized real-time logistics. Not only that, the expectation is to democratize data, so that the majority of the workforce can put data into action.

Meeting these expectations requires a new approach to data management. An approach that can seamlessly fulfill the *diversity of modes of access to data*. Access that ranges from a simple structured view of the data for reporting, to a continuously reshaping semi-structured data for machine learning training; from real-time fine-grained access to events to aggregations. We need to meet these expectations with an approach and architecture that natively supports diverse use cases and does not require copying data from one technology stack to another across the organization so that we can meet the needs of yet another use case.

More importantly, the widespread use of machine learning requires a new attitude toward application development and data. Need to move from deterministic and rule-based applications - where given a specific input data, the output can be determined - to nondeterministic and probabilistic data-driven applications - where given a specific input data, the output could be a range of possibilities which can change over time. This approach to application development requires continuous refining of the model over time, and continuous, frictionless *access to the latest data*.

The great and diverse expectations of data require us to step back, acknowledge the accidental technical complexities that we have created over time, and wonder if there is a simpler approach to data management that can universally address the diversity of needs today, and beyond.

The Great Divide of Data

Many of the technical complexities organizations face today stem from how we have divided the data -- *operational* and *analytical data*, siloed the teams that manage them, proliferated the technology stacks that support them and how we have integrated them.

Today, we have divided the data and its supporting technology stacks and architecture into two major categories: *operational data*: databases that support running the business and keeping the current state of the business - also known as transactional data; and *analytical data*: data warehouse or lake providing a historical, integrated and aggregate view of data created as a byproduct of running the business. Today, operational data is collected and transformed to form the analytical data. Analytical data trains the machine learning models that then make their way into the operational systems as intelligent services.

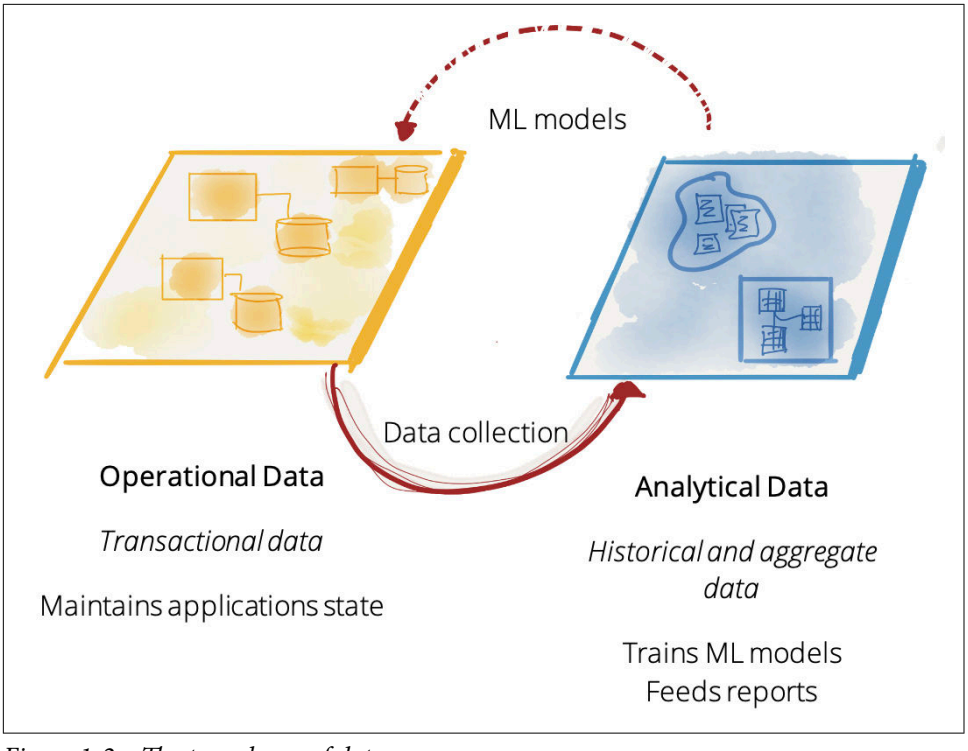


Figure 1-2. : The two planes of data

Operational Data

Operational data sits in databases of microservices, applications, or systems of records that support the business capabilities. Operational data keeps the current state of the business. It is optimized for application's or microservice's logic and access patterns. It often has a transactional nature. It's referred to as data on the inside, private data of an application or a microservice that performs CRUD (create, update, delete) operations on it. Operational data is constantly updated, so it's access requires reads and writes. The design has to account for multiple people updating the same data at the same time in unpredictable sequences (hence the need for transactions). The access is also about relatively in-the-moment activity. Operational data is recording what happens in the business, supporting decisions that are specific to the business transaction. In short, *operational data is used to run the business and serve the users.*

Imagine a digital media streaming business that streams music, podcast and other digital content to its subscribers and listeners. Registration service implements the business function of registering new users or unregistering them. The database that

supports the registration and deregistration process, keeping the list of users, is considered operational data.

INTRODUCING DAFF INC.

Daff Inc. is a global digital streaming company, that started its journey with streaming music and rapidly growing to provide other digital audio experiences such as sharing podcasts, radio shows and audio books. Daff Inc. serves both paid and free subscribers. Daff's mission is to get everyone's audio content heard; from independent artists, podcasters, to record labels and larger publishers. Daff hosts social events to connect the audience with the performers.

Daff Inc. is making big investments in a robust data and AI infrastructure to become data-driven; use their data to optimize every single aspect of their business. Serve the listeners with recommendations specialized to their taste, mood, time of day and location; empower the artists with information about their listeners such as locations, listeners profile, campaign results to help them refine their work; optimize the quality of their digital services using users and digital players captured events; and ultimately streamline their business operations such as artist onboarding, payments and advertisements using up to date and accurate data.

The word 'Daff' is the name of a Persian percussion instrument, dated more than 3000 years.

Analytical Data

Analytical data is the temporal, historic and often aggregated view of the facts of the business over time. It is modeled to provide retrospective or future-perspective insights. Analytical data is optimized for analytical logic - training machine learning models, creating reports and visualizations. Analytical data is called data on the outside, data directly accessed by analytical consumers. Analytical data is immutable and has a sense of history. Analytical use cases require looking for comparisons and trends over time, while a lot of operational uses don't require much history. The original definition of analytical data as *a nonvolatile, integrated, time variant collection of data*[2] still remains valid.

In short, analytical data is used to optimize the business and user experience. This is the data that fuels the AI and analytics aspirations that we talked about in the previous section.

For example, in the case of Daff Inc. it's important to optimize the listeners' experience with playlists recommended based on their music taste and favorite artists. The

2 Definition provided by William H. Inmon known as the father of data warehousing.

analytical data that helps train the playlist recommendation machine learning model, captures all the past behavior of the listener as well as all characteristics of the music the listener has favored. This aggregated and historical view is analytical data.

Over time, the analytical data plane itself has diverged into two generations of architectures and technology stacks: initially data warehouse and followed by data lake; with data lake supporting data science access patterns and preserving data in its original form, and data warehouse supporting analytical and business intelligence reporting access patterns with data conforming to a centrally unified ontology. For this conversation, I put aside the dance between the two technology stacks: data warehouse attempting to onboard data science workflows and data lake attempting to serve data analysts and business intelligence.

Analytical and Operational Data Misintegration

The current state of technology, architecture and organization design is reflective of the divergence of the analytical and operational data planes - two levels of existence, integrated yet separate. Each plane operates under a different organizational vertical. Business intelligence, data analytics and data science teams, under the leadership of Chief Data and Analytics officer (CDAO), manage the analytical data plane, while business units and their corresponding technology domains manage the operational data. From the technology perspective, there are two independent technology stacks that have grown to serve each plane, while there are some convergence such as infinite event logs.

This divergence has led to the two-plane data topology and a fragile integration architecture between the two. The operational data plane feeds the analytical data plane through a set of scripts or automated processes often referred to as ETL jobs - Extract, Transform, and Load. Often operational databases have no explicitly defined contract with the ETL pipelines for sharing their data. This leads to fragile ETL jobs where unanticipated upstream changes to the operational system and their data leads to downstream pipeline failures. Over time the ETL pipelines grow in complexity trying to provide various transformations over the operational data, flowing data from the operational data plane to the analytical plane, and back to the operational plane.

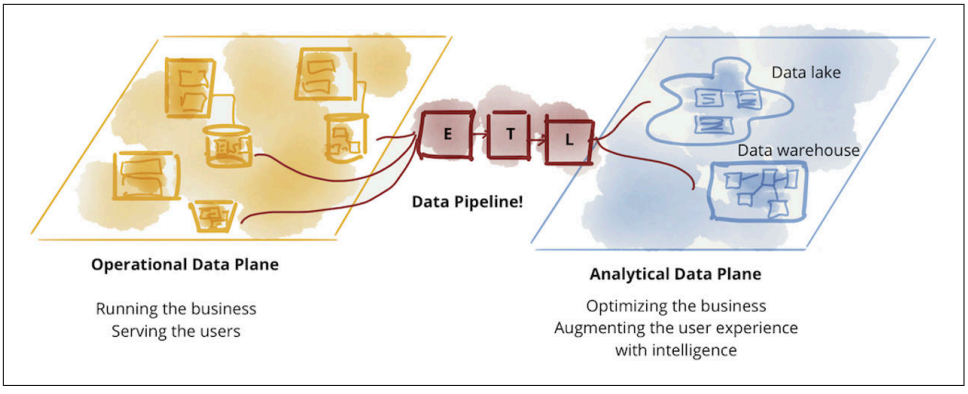


Figure 1-3. Pipeline-based integration of the data planes

The challenges of the two-plane data management approach with a brittle integration through pipelines, and a centralized data warehouse or lake for access to data is a major driver to reimagine the future solutions.

Scale, Encounter of a New Kind

Since the mid 2000s, we have evolved our technologies to deal with the scale of the data in terms of its *volume, velocity and variety*. We built the first generation batch data processing to manage the large volume of data that our applications and touchpoints generated, we built stream processing architectures to handle the speed of data that started flowing from our mobile devices, and built different types of storage systems to manage the diversity of data, text, imaging, voice, graphs, files, etc. Then we got carried away and kept tagging more Vs to data to encourage access to clean data - *veracity* - and aim to get *value*[3] from data.

Today, we are encountering a new kind of scale, the *origins and location of the data*. The data-driven solutions often require access to data beyond a business domain, organizational or technical boundary. The data can be originated from every system that runs the business, from every touchpoint with customers, and from other organizations. The next approach to data management needs to recognize the proliferation of the origins of the data, and their ubiquitous nature.

The most interesting and unexpected patterns emerge when we connect data from a variety of sources, when we can have access to information that is beyond the transactional data that we generate running our business. The future of intelligent healthcare requires a longitudinal human record of a patient's diagnostics, pharmaceutical records, personal habits, etc. and in comparison with all other patients' history. These

3 https://www.bbva.com/en/five-vs-big-data

sources are beyond a single organization's control. The future of intelligent banking requires data beyond the financial transactions that customers perform with their banks. They'll need to know the customers' housing needs, the housing market, their shopping habits, their dreams, to offer them the services they need, when they need it.

This unprecedented scale of diversity of sources, requires a shift in data management. A shift away from collecting data from sources into one big centralized place, repeatedly across every single organization, to *connecting data*, wherever it is.

Beyond Order

I'm writing this book during the pandemic of 2020-2021. If there was any doubt that our organizations need to navigate *complexity, uncertainty and volatility*, the pandemic has made that abundantly clear. Even on a good day outside of the pandemic, the complexity of our organizations demand a new kind of immunity, *immunity to change*.

The complexity that has risen from the ever changing landscape of a business is also reflected in the data. Rapid delivery of new features to products, new and changed offerings and business functions, new touchpoints, new partnerships, new acquisitions, all result in a continuous reshaping of the data.

More than ever now, organizations need to have the pulse of their data and the ability to act quickly and respond to change with *agility*.

What does this mean for the approach to data management? It requires access to the quality and trustworthy facts of the business at the time they happen. The data platforms must *close the distance* - time and space - between when an event happens, and when it gets consumed and processed for analysis. The analytics solutions must guide *real time decision making*. Rapid response to change is no longer a premature optimization[4] of the business; it's a baseline functionality.

Data management of the future must build-in change, by default. Rigid data modeling and querying languages that expect to put the system in a straitjacket of a never-changing schema can only result in a fragile and unusable analytics system.

Data management of the future must embrace the complex nature of today's organizations and allow for *autonomy* of teams with *peer-to-peer* data collaborations.

Today, the complexity has stretched beyond the processes and products to the technology platforms themselves. In any organization, the solutions span across multiple cloud and on-prem platforms. The data management of the future must support

4 Donald Knuth made the statement, "(code) premature optimization is the root of all evil."

managing and accessing data *across multiple cloud providers*, and on-prem data centers, by default.

Approaching the Plateau of Return

In addition to the seismic shifts listed above, there are other telling tales about the mismatch between data and AI investment and the results. To get a glimpse of this, I suggest you browse the NewVantage Partners annual reports; an annual survey of senior corporate c-executives on the topics of data and AI business adoption. What you find is the recurring theme of an increasing effort and investment in building the enabling data and analytics platforms, and yet experiencing low success rates.

For example, in their 2021 report, only 26.8% of firms reported *having forged a data culture.* Only 37.8% of firms reported that they have *become data-driven,* and only *45.1%* of the firms reported that they are *competing using data and analytics.* It's too little result for the pace and amount of investment; 64.8% of surveyed companies reported greater than $50MM investment in their Big Data and AI strategies.

Despite continuous effort and investment in one generation of data and analytics platforms to the next, the organizations find the results middling.

I recognize that the organizations face a multi-faceted challenge in transforming to become data-driven; migrating from decades of legacy systems, resistance of a legacy culture to rely on data, and competing business priorities.

The future approach to data management must look carefully at this phenomena, why the solutions of the past are not producing a comparable result to the human and financial investment we are putting in today. Some of the root causes include lack of skill sets needed to build and run data and AI solutions, organizational, technology and governance bottlenecks, friction in discovering, trusting, accessing and using data.

Recap

As a decentralized approach to managing data, Data Mesh embraces the data realities of organizations today, and their trajectory, while acknowledging the limitations our solutions face today.

Data Mesh assumes a new default starting state: proliferation of data origins within and beyond organizations boundaries, on one or across multiple cloud platforms. It assumes a diverse range of use cases for analytical data from hypothesis-driven machine learning model development to reports and analytics. It works with a highly complex and volatile organizational environment and not against it.

In the next two chapters, I set the stage further. Next, we look at our expectations from Data Mesh as a post-inflection-point solution. What organizational impact we expect to see, and how Data Mesh achieves them.

CHAPTER 2
After The Inflection Point

A note for Early Release readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the 2nd chapter of the final book.

If you have comments about how we might improve the content and/or examples in this book, or if you notice missing material within this chapter, please reach out to the editor at gobrien@oreilly.com.

> *The only way to make sense out of change is to plunge into it, move with it, and join the dance.*
>
> —Alan Watts

Standing at an inflection point is a magical experience. It's where we learn and depart from the past and choose a new path. It's a point where we have a choice to turn to a new direction. The rest of this book provides instructions on how to move toward this new direction with Data Mesh. We will discuss what constitutes Data Mesh in Part II, how to architect it in Part III, and how to begin executing it in Part IV. However, before we dive into that, in this chapter, I'd like to introduce Data Mesh based on its *impact* on organizations, given the *environmental conditions* it operates in, and the *issues from the past solutions* it must address.

Data Mesh must accept the *environmental conditions* that we discussed in Chapter 1, as the default starting point. It must assume, by default, the *ubiquitous nature of the data*. Data can be of any origin, it can come from systems within an organization, or outside, beyond the boundary of organizational trust. It can be physically served by

any underlying platform on one cloud hosting or another. Data Mesh must embrace the *diversity of the use cases* and their unique modes of access to data. The use cases range from historical data analysis and reporting, training machine learning models and data-intensive applications. Each needs to read data in a different format, in the spectrum of graphs, files, tables and events. An ever increasing *complexity of the business* landscape--its diversity of functions, continuous change, and need for real-time decision making in volatile times--is the organizational reality within which Data Mesh must succeed.

Data Mesh must *learn from the past solutions* and address their shortcomings. It must *reduce points of centralization* that act as coordination bottlenecks. It must find a new way in *decomposing the data architecture* that, unlike technology-driven decomposition, does not slow the organization down with multi-point synchronizations. It must remove the gap between where the data originates and where it gets used in its analytical form, and *remove all the accidental complexities* - aka pipelines - that happen in between the two planes of data. Data Mesh must depart from data myths such as a single source of truth, or one tightly-controlled canonical model.

Ultimately, Data Mesh's *goal is to enable organizations to thrive* in the face of the *growth of data sources*, *growth of data users* and *use cases,* and the increasing change in cadence and complexity. Adopting Data Mesh, organizations must thrive in *agility, creating data-driven value* while *embracing change*.

Figure 2-1 lists the expected organizational outcomes applying Data Mesh, as the organization size and complexity grows, as the diversity of data and organization's data aspirations scale.

Figure 2-1. Data Mesh outcomes for organizations

In this chapter we look at the top-level outcomes that your organization achieves by adopting Data Mesh, the impact of Data Mesh and why you should care about it. For each of these outcomes, I discuss how Data Mesh accomplishes them, what shifts it creates. In discussing the shifts I give you a brief description of the foundational principles of Data Mesh -- *Domain Data Ownership, Data as a Product, Self-serve Data Platform, Computational Federated Governance*.You will see these in action and I point you to Part II and Part III of the book where you can get all the details.

Embrace Change in a Complex, Volatile and Uncertain Business Environment

Businesses are complex systems, composed of many domains that each have their own accountability structure, goals, and each changing at a different pace. The behavior of the business as a whole is the result of an intricate network of relationships between its domains and functions, their interactions and dependencies. The volatility and rapid change of the markets and regulations within which the businesses operate compounds the complexity.

How can businesses manage the impact of such complexity on their data? How can organizations keep going through change while continuing to get value from their data? How can businesses avoid increased cost of managing the change of their data landscape? How can they provide truthful and trustworthy data without disruption, in the face of continuous change? This comes down to *embracing change* in a complex organization.

In this section I discuss a few ways Data Mesh achieves embracing change despite increased complexity of the business.

Align Business, Tech and Now Analytical Data

One way to manage complexity is to break it down into independently managed parts . Businesses do this by creating domains. For example, Daff Inc. breaks down its business domains according to relatively independent outcomes and functions--including managing podcasts, managing artists, player applications, playlists, payments, marketing, etc.

This allows the domains to move fast without tight synchronization dependencies to other parts of the business.

Just as a business divides its work through business domains, technology can, and should, align itself to these business divisions. We see the best organizations orienting their technology staff around their business units, allowing each business unit to be supported by a dedicated technology capability for that unit's work. The recent movement towards Microservices is largely about performing this kind of decomposition. As part of this we see business units controlling and managing their operational applications and data.

The first principle of data mesh carries out the same decomposition for analytic data, resulting in the *Domain Ownership of Data*. Each business unit takes on the responsibility for analytic data ownership and management. This is because the people who are closest to the data are best able to understand what analytic data exists, and how it should best be interpreted.

Domain ownership distribution results in a distributed data architecture, where the data artifacts - datasets, code, metadata, and data policies - are maintained by their corresponding domains

Figure 2-2 is demonstrating the concept of organizing technology (services), analytical data aligned with business.

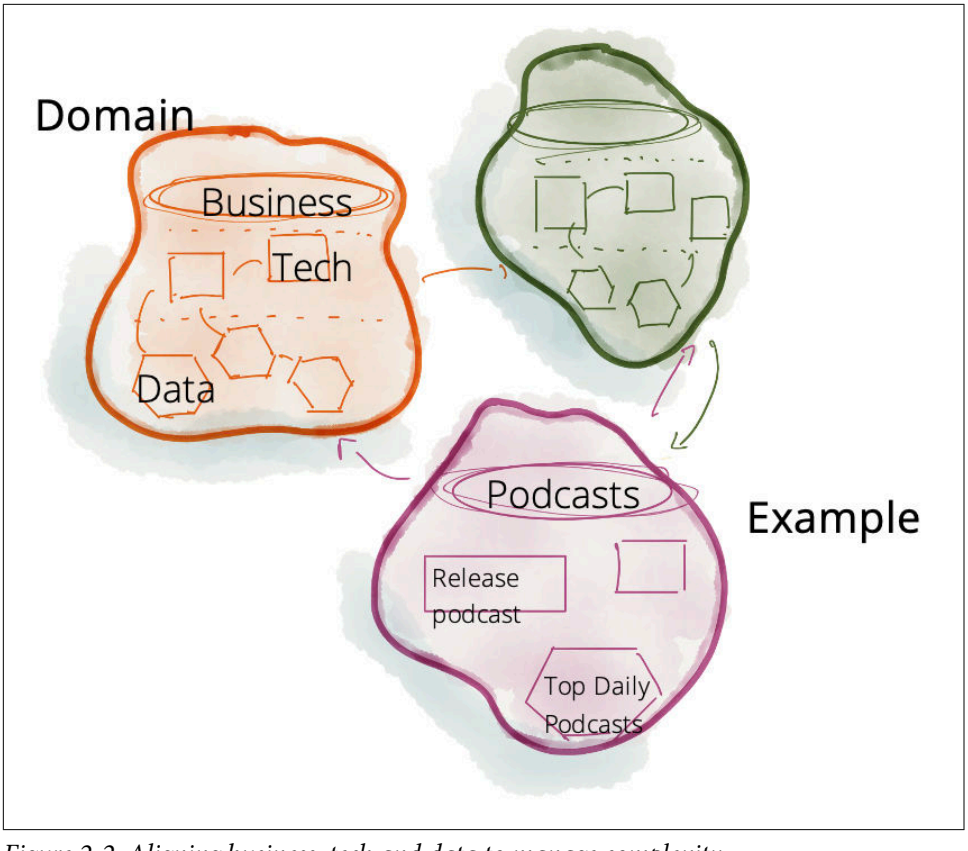


Figure 2-2. Aligning business, tech and data to manage complexity

See Chapter 4, The Principle of Domain Ownership for further details.

Close The Gap Between Analytical and Operational Data

To make good decisions in the moment, analytical data must reflect *business truthfulness.* They must be as close as possible to the facts and reality of the business at the moment the decision is made. As we saw in Chapter 1, this can't be achieved with two separate data planes - analytical and operational data planes - that are far from each other and connected through fragile data pipelines and intermediary data teams. Data

pipelines must dissolve and give way to a new way of providing the analytical data and capabilities as close to the source as possible.

Changes in the business, such as adding a new feature to a product, introducing a new service, or optimizing a workflow, must be reflected near real time in both the state of the business captured by operational data as well as its temporal view captured by the analytical data.

Data Mesh suggests that we continue to recognize and respect the differences between these two planes: the nature and topology of the data, the differing use cases, their unique personas of consumers, and ultimately their diverse access patterns. However Data mesh connects these two planes under a different structure - *an inverted model and topology based on domains and not technology stack* - where each domain extends its responsibilities to not only provide operational capabilities but also serve and share *analytical data as a product.*

Data Mesh principles of *Data as a Product* introduces an accountability for the domains to serve their analytical data as a product and delight the experience of data consumers; streamlining their experience discovering, understanding, trusting, and ultimately using quality data. Data as a product principle is designed to address the data quality and the age-old siloed data problem, and their unhappy data consumers. See Chapter 6, The Principle of Data as a Product for more on this.

Implementing this approach introduces a new architectural unit, called data product quantum that will embed all the structural components needed to maintain and serve data as a product. The structural components include the code that maintains the data, additional information, metadata, to make data discoverable and usable, and a contract to access the data in a variety of access modes native to the data consumers.

Figure 2-3 demonstrates a different integration model between operational and analytical planes. You have seen these planes in chapter 1, integrated through clever and complex data pipelines. Here, the planes are divided by business domains. The integration between data product quantums, the analytical data plane, and their corresponding domain's operational plane services are rather simple and unintelligent. A simple movement of data. Data product quantums will embed and abstract the intelligence and code required to transform the operational data into its analytical form.

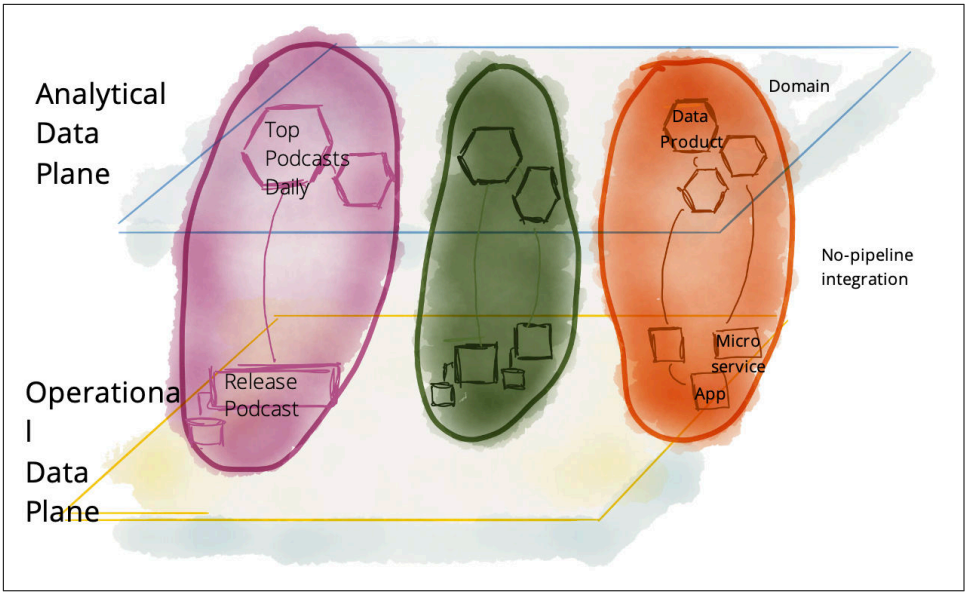


Figure 2-3. Closing the gap between operational and analytical data

Differences in today's available technology to manage the two archetypes of data should not lead to the separation of organizations, teams, and people who work on them. I believe that our technologies will evolve at some point in the future to bring these two planes even closer together, but for now, I suggest we keep their concerns separate. The primary focus of this book, and Data Mesh itself, is on the analytical plane and its integration with the operational plane.

Localize Data Change to Business Domains

Data Mesh must allow for data models to change continuously without fatal impact to downstream data consumers, or slowing down access to data as a result of synchronizing change of a shared global canonical model. Data Mesh achieves this by localizing change to domains by providing autonomy to domains to model their data based on their most intimate understanding of the business without the need for central coordinations of change to a single shared canonical model.

Data Mesh imposes contracts, well-defined and guaranteed interfaces, to share data. This liberates domains to change their data models, given that they still support the older revisions of their contracts, until they gracefully migrate their data users to the new revisions. Data Mesh introduces a set of discovery APIs that allow data product users to locate and consume data according to the guarantees of the data discovery APIs. See Chapter 4, The Principle of Domain Ownership and Chapter 5, The Principle of Data as a Product for more on this.

Reduce Accidental Complexity of Pipelines and Copying Data

As Fred Brooks laid out in his widely popular paper, "No Silver Bullet – Essence and Accident in Software Engineering", there are two types of complexity when building software systems. First, the essential complexity: the complexity that is essential and inherent to the problem space. This is the business and domain complexity we discussed earlier. And second, the accidental complexity: the complexity that we - engineers, architects and designers - create in our solutions and can be fixed.

The world of analytical solutions is full of opportunities to remove and fix accidental complexities. Let's talk about a few of those accidental complexities that Data Mesh must reduce.

Today, we keep copying data around because we need the data for yet another mode of access, or yet another model of computation. We copy data from operational systems to a data lake for data scientists. We copy the data again into lakeshore marts for data analyst access and then into the downstream dashboard or reporting databases for the last mile. We build complex and brittle pipelines to do the copying. The copying journey continues across one technology stack to another and across one cloud vendor to another. Today, to run analytical workloads you need to decide upfront which cloud provider copies all of your data in its lake or warehouse before you can get value from it.

Data Mesh addresses this problem by creating a new architectural unit that encapsulates a domain-oriented data semantic, but yet provides multiple modes of access to the data suitable for different use cases and users. This architectural unit is called the Data Product Quantum. It will have a clear contract and guarantees to its readers, and meet their native access mode, SQL, files, events, etc. Data product quantum can be accessed anywhere across the internet and it provides access control and policy enforcement necessary at the time of access, locally at its interface. Data product quantum encapsulates the code that transforms and maintains its data. With abstraction of transformation code inside a data product quantum, and accessing data through data product quantum interfaces, the need for pipelines will go away. Removing the brittle concept of pipeline reduces the opportunity for failure in case of an upstream data change. Data Mesh introduces standardized interfaces to discover and access every data product enabled by a self-serve infrastructure. See Chapter 8 on the logical architecture of Data Mesh, and Chapter 9 for more details on the data product quantum and Chapter 6, on self-serve data infrastructure.

Sustain Agility in the Face of Growth

Today, businesses' successes are predicated on multi-faceted growth--new acquisitions, new service lines, new products, geolocation expansions and so on. All this leads to new sources of data to manage and new data-driven use cases to build. Many

organizations slow down or plateau in the speed of delivering value from their data, onboarding new data, or serving the use cases as they grow.

Data Mesh's approach to sustain agility in the face of growth can be summarized in a few techniques that aim to reduce bottlenecks, coordination, and synchronization. Agility relies on business domains' ability to achieve outcomes autonomously and with minimal dependencies.

Remove Centralized and Monolithic Bottlenecks of the Lake or the Warehouse

A centralized data team, managing a monolithic data lake or warehouse limits agility, particularly as the number of sources to onboard or number of use cases grow. Data Mesh looks carefully for centralized bottlenecks, particularly where they are the focal point of multi-party synchronization, both from the human communication perspective and architecture. These bottlenecks include data lakes and data warehouses.

Data Mesh proposes an alternative, a peer-to-peer approach in data collaboration when serving and consuming data. The architecture enables consumers to directly discover and use the data right from the source. For example, an ML training function or a report, can directly access independent data products, without the intervention of a centralized architectural component such as a lake or a warehouse, and without the need for an intermediary data (pipeline) team. See chapter 9 for details of peer-to-peer data consumption through data product quantum's output data ports and input data ports.

Figure 2-4 demonstrates the conceptual shift. Each data product provides versioned interfaces that allow peer-to-peer consumption of domain-data to compose aggregates or higher-order data products.

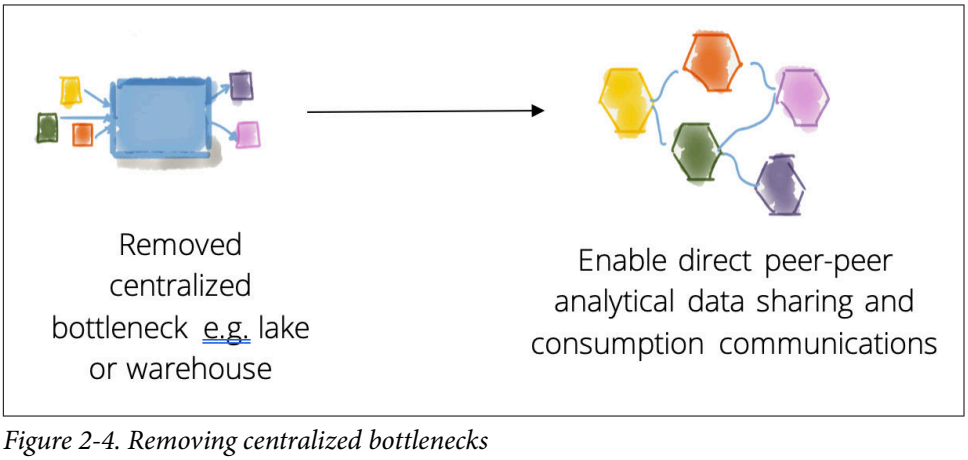


Figure 2-4. Removing centralized bottlenecks

Reduce Coordination of Data Pipelines

Over the last decades, the technologies that have exceeded in their operational scale have one thing in common, they have minimized the need for coordination and synchronization. Asynchronous IO has scaled the throughput of networked applications over blocking IO. Reactive applications, for example, have resulted in faster parallel processing of messages. Apache Hadoop scaled data processing by running Map-Reduce functional programming model across many servers distributedly. Using choreographed event-driven microservices over centrally orchestrated ones has allowed us to scale our business workflows.

Despite our relentless effort to remove coordination and synchronization from our core technologies in order to achieve scale and speed, we have, for the most part, neglected organizational and architectural coordination. As a result, no matter how fast our computer systems run, achieving outcomes have fallen behind coordinating activities of teams and humans.

Data Mesh addresses two of the main coordination issues in the data management processes and architecture. A technically-divided architecture of a pipeline - ingestion, processing, serving, etc. - results in the coordination of these functions to deliver a new data source, or serve a new use case. Data Mesh moves away from technical-partitioning of data management, to domain-oriented partitioning. Domain-oriented data products must be able to develop and evolve independently of other data products. The domain-oriented decomposition reduces the need for coordination to achieve an outcome. For the most part, a new data source or a new use case can be served by a domain-oriented data product team. In cases where a new use case requires access to a new data product outside of the domain, the consumer can make progress by utilizing the standard contracts of the new data product, mocks, stubs, or synthetic data interfaces until the data product becomes available. This is the beauty of contracts, easing the coordination between consumer and provider during development. See chapter 4, The Principle of Domain Ownership for more on this.

Reduce Coordination of Data Governance

Today, another major coordination bottleneck is the central function of data governance. Data governance coordination is necessary to provide access to data users, approve the quality of datasets, and validate the conformance of data changes with the organization's policies.

Data Mesh introduces a *federated and computational data governance model*, where the governance team is composed of the individual domain data product owners, the main owners of the data products. The governance function aims to embed policy execution into every data product in a computational and automated fashion. This vastly improves the function of governance today, which is one of the main synchro-

nization points for discovering data, approving data, and making sure it follows the necessary policies.

As you can imagine, the autonomy of the domains can have undesirable consequences if not checked; isolation of domains, incompatibility and disconnection of one domain's data product from others, and a fragmented experience when consuming multiple domains' data. Data Mesh governance heavily relies on the automation of governance concerns for a consistent, connected and trustworthy experience using the domains' data products.

See Chapter 7, The Principle of Federated Computational Governance.

Enable Autonomy

The correlation between team autonomy and team performance has been the subject of team management studies. Empirical studies show that teams' freedom in decision making to fulfill their mission can lead to better team performance. On the other hand, too much autonomy can result in inconsistencies, duplicated efforts and team isolation.

Data Mesh attempts to strike a balance between team autonomy and inter-term interoperability and collaboration, with a few complementary techniques. It gives domain teams autonomy to have control of their local decision making, such as choosing the best data model for their data products. While it uses the computational governance policies to impose a consistent experience across all data products; for example, standardizing on the data modeling language that all domains utilize.

Domain teams are given autonomy to build and maintain the lifecycle of their data products. While Data Mesh places a domain-agnostic data platform team who empowers the domain teams with self-serve capabilities to manage the lifecycle of data products declaratively and consistently, to prevent team isolation and decrease cost of autonomy.

The *principles of self-serve data platform*, essentially makes it feasible for domain teams to manage the lifecycle of their data products with *autonomy*, and utilize the skillsets of their generalist developer to doso.

These self-serve data infrastructure APIs allow data product developers to build, deploy, monitor and maintain their data products. The APIs allow data consumers to discover, learn, access, and use the data products. The self-serve infrastructure makes it possible for the mesh of data products to be joined, correlated and used as a whole, while maintained independently by the domain teams.

See Chapter 6, The Principle of Self-serve Data Platform, for more.

Increase the Ratio of Value from Data to Investment

Industry reports, such as the NewVantage Partner I shared in chapter 1, and my personal experience, point to the fact that we are getting little value from data compared to the investments we are making in data management. If we compare the value we get from our data teams and data solutions, compared to other technical developments such as infrastructure, mobile and application development, it's evident that we are facing headwinds when it comes to data.

Data Mesh looks at ways to improve the ratio of value over effort in analytical data management: creation of a new archetype of data platform that abstracts today's technical complexity, through open data interfaces that enable sharing data across organizational trust boundary or physical location, and through applying product thinking to remove friction across all stages of the value stream that gets analytical data from the points of origin to the hands of data scientists and analysts.

Abstract Technical Complexity with a Data Platform

Today's landscape of data management technology is undoubtedly too complex. The litmus test for technical complexity is the ever growing need for *data engineers* and *data technology experts*. We don't seem to ever have enough of them. Another litmus test is the low value to effort ratio of data pipeline projects. Much effort is spent with little value returned - getting access to baseline datasets with quality.

Data Mesh looks critically at the existing technology landscape, and reimagines the technology solutions as a *data-product-developer*(or consumer)-*centric* platform. In chapter 6, The Principle of Self-Serve Data Platform, we will see how Data Mesh arrives at a set of self-serve data platform capabilities to remove friction and complexity from the workflows of *data product developers* and *data product users*. Data Mesh intends to remove the need for data specialists and enable generalist experts to develop data products.

Additionally, Data Mesh defines a set of open interfaces for different affordances of data products - discovering, querying, serving data, etc. - to enable a more collaborative ecosystem of tools. This is in contrast to a heavily proprietary data technology landscape with a high cost of integration across vendors. See chapter 9 on data products' open interfaces.

Embed Product Thinking Everywhere

Data Mesh introduces a few shifts to get us laser focused on the value, as perceived by the data users. It shifts our thinking from data as an *asset* to data as a *product*. It shifts how we measure success from the volume of the data to measure to the happiness and satisfaction of the data users. See chapter 5, The Principle of Data as a Product, for more details on achieving this shift.

Data is not the only component of a Data Mesh ecosystem that is treated as a product. The self-serve data platform itself is also a product. In this case, it serves the data product developers and data product consumers. Data Mesh shifts the measure of success of the platform from the number of its capabilities, to the impact of its capabilities on improving the experience of data product development, the reduced lead time to deliver, or discover and use of a data product. See chapter 5, The Principle of Self-Serve Data Platform for more on this.

Go Beyond The Boundaries

The value that a business unit can generate almost always requires data beyond the unit's boundary, requiring data that comes from many different business domains. Similarly, the value that an organization can generate serving its customers, employees, and partners often requires access to data beyond the data that the organization generates and controls.

Consider Daff Inc. In order to provide a better experience to the listeners with autoplay music, it not only requires data from listeners' playlist, but also their network, as well as their social and environmental influences and behaviors. It requires data from many corners of Daff Inc, and beyond including news, weather, social platforms, etc.

Multi-domain and multi-org access to data is an assumption built into Data Mesh. Data Mesh's data product concept can provide access to data no matter where the data physically resides. Data product quantum provides a set of interfaces that essentially allow anyone with the proper access control, discover, and use the data product independent of its physical location. The identification schema, access control and other policy enforcement assumes using open protocols that can be enabled over the internet. See chapter 8, Data Mesh Logical Architecture for more on this.

Recap

After reading this chapter you might assume that Data Mesh is a silver bullet. Quite the contrary. Data Mesh is an important piece of the puzzle. It enables us to truly democratize access to data. However to close the loop of deriving value from data, there is much more that needs to be done beyond just getting access to data. We must be able to continuously deliver analytical and ML-based solutions. However bootstrapping this closed loop, requires access to data at scale, which is a focus of Data Mesh.

The Data Mesh goals listed in this chapter, invites us to *reimagine* data, how to architect solutions to manage it, how to govern it, and how to structure our teams . The expectation to *become resilient to business complexity, to sustain agility in the face of growth,* and *accelerate getting value from data* pushes us to work with the reality of increasing complexity, growth and volatility instead of fighting to control it.

In the next chapter, I will give an overview of what has happened before the inflection point. Why the data management approach that got us here, won't take us to the future.

CHAPTER 3
Before The Inflection Point

A note for Early Release readers

With Early Release ebooks, you get books in their earliest form—the author's raw and unedited content as they write—so you can take advantage of these technologies long before the official release of these titles.

This will be the 3rd chapter of the final book.

If you have comments about how we might improve the content and/or examples in this book, or if you notice missing material within this chapter, please reach out to the editor at gobrien@oreilly.com.

Today's problems come from yesterday's "solutions."
—Senge, Peter M. The Fifth Discipline

Organizational complexity, growth of data sources, proliferation of data expectations. These are the forces that have put stress on our existing approaches to analytical data management. Our existing methods have made remarkable progress scaling the machines: manage large volumes of a variety of data types with planet scale distributed data storage, reliably transmit high velocity data through streams, and process data intensive workloads, concurrently and fast. However, our methods have limitations with regard to the organizational complexity and scale, the human scale.

In this chapter, I give a short introduction to the current landscape of data architectures, their underlying characteristics and the reasons why, moving into the future, they limit us.

Evolution of Analytical Data Architectures

How we manage analytical data has gone through evolutionary changes; changes driven by new consumption models, ranging from traditional analytics in support of business decisions to intelligent business functions augmented with ML. While we have seen an accelerated growth in the number of analytical data technologies, the *high level architecture* has seen very few changes. Let's have a quick browse of the high level analytical data architectures, followed by a review of their unchanged characteristics.

The underlying technologies supporting each of the following architectural paradigms have gone through many iterations and improvements. The focus here is on the architectural pattern, and not the technology and implementation evolutions.

First Generation: Data Warehouse Architecture

Data warehousing architecture today is influenced by early concepts such as facts and dimensions formulated in the 1960s. The architecture intends to flow data from operational systems to business intelligence systems that traditionally have served the management with operations and planning of an organization. While data warehousing solutions have greatly evolved, many of the original characteristics and assumptions of their architectural model remain the same:

- Data is extracted from many operational databases and sources
- Data is transformed into a universal schema - represented as a multi-dimensional and time-variant tabular format
- Data is loaded into the warehouse tables
- Data is accessed through SQL-like querying operations
- Data is mainly serving data analysts for their reporting and analytical visualizations use cases

The data warehouse approach is also referred to as data marts with the usual distinction that a data mart serves a single department in an organization, while a data warehouse serves the larger organization integrating across multiple departments. Regardless of their scope, from the architectural modeling perspective they both have similar characteristics.

In my experience, the majority of enterprise data warehouse solutions are proprietary, expensive and require specialization for use. Over time, they grow to thousands of ETL jobs, tables and reports that only a specialized group can understand and maintain. They don't let themselves to modern engineering practices such as CI/CD and

incur technical debt over time and an increased cost of maintenance. Organizations attempting to escape this debt, find themselves in an inescapable cycle of migrating from data warehouse solution to another.

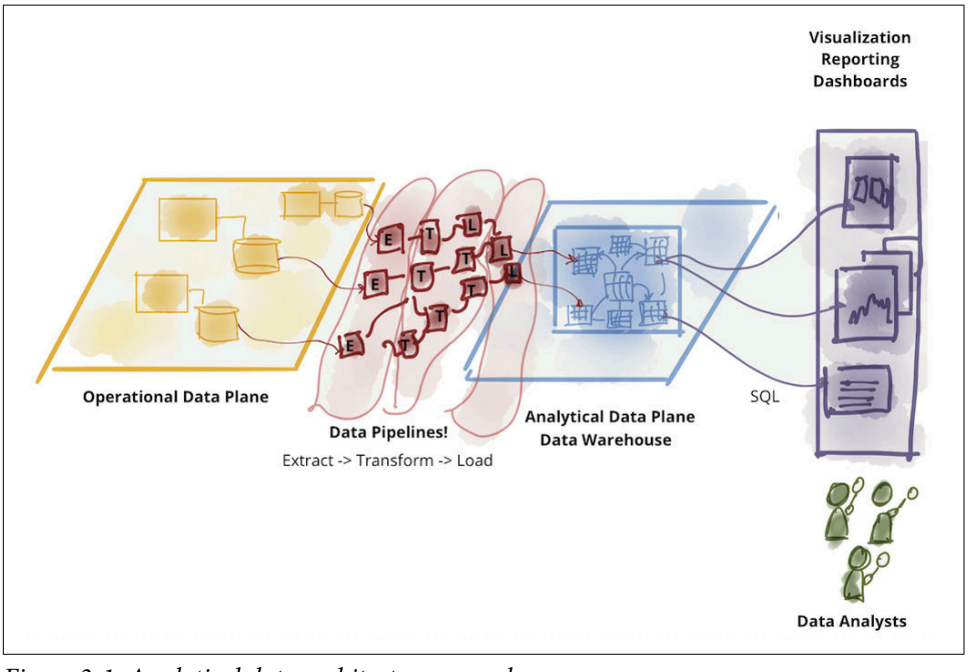


Figure 3-1. Analytical data architecture - warehouse

Second Generation: Data Lake Architecture

Data lake architecture was introduced in 2010 in response to challenges of data warehousing architecture in satisfying the new uses of data; access to data based on data science and machine learning model training workflows, and supporting massively parallelized access to data. Data lake architecture, similarly to data warehouse, assumes that data gets extracted from the operational systems and is loaded into a central repository often in the format of an object store, storage of any type of data. However unlike data warehousing, data lake assumes no or very little transformation and modeling of the data upfront; it attempts to retain the data close to its original form. Once the data becomes available in the lake, the architecture gets extended with elaborate transformation pipelines to model the higher value data and store it in lakeshore marts.

This evolution to data architecture aims to improve ineffectiveness and friction introduced by extensive upfront modeling that data warehousing demands. The upfront transformation is a blocker and leads to slower iterations of model training. Additionally, it alters the nature of the operational system's data and mutates the data in a

way that models trained with transformed data fail to perform against the real production queries.

In our example, a music recommender when trained against a transformed and modeled data in a warehouse, fails to perform when evoked in an operational context - e.g. evoked by the recommender service with the logged-in user's session information. The heavily transformed data used to train the model, either misses some of the user's signals or has created a different representation of users attributes. Data lake comes to rescue in this scenario.

Notable characteristics of a data lake architecture include:

- Data is extracted from many operational databases and sources
- Data is minimally transformed to fit the storage format e.g. Parquet, Avro, etc.
- Data - as close as the source syntax - is loaded to scalable object storage
- Data is accessed through the object storage interface - read as files or data frames - a two-dimensional array-like structure.
- Lake storage is accessed mainly for analytical and machine learning model training use cases and used by data scientists
- Downstream from the lake, lake shore marts, are fit-for-purpose data marts or data services serve the modeled data
- Lakeshore marts are used by applications and analytics use cases

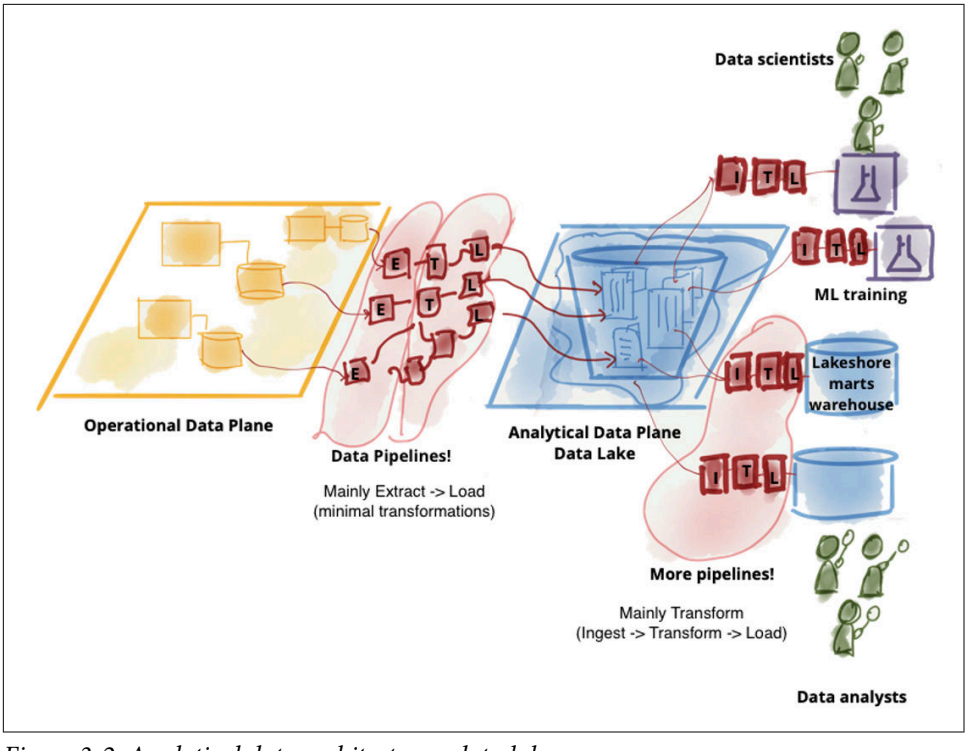


Figure 3-2. Analytical data architecture - data lake

Data lake architecture suffers from complexity and deterioration; complex and unwieldy pipelines of batch or streaming jobs operated by a central team of hyper-specialized data engineers; deteriorated and unmanaged datasets, untrusted, inaccessible, provide little value.

Third Generation: Multimodal Cloud Architecture

The *third and current generation data architectures* are more or less similar to the previous generations, with a few modern twists:

- Streaming for *real-time data availability* with architectures such as Kappa
- Attempting to unify the batch and stream processing for data transformation with frameworks such as Apache Beam
- Fully embracing cloud based managed services with modern cloud-native implementations with isolated compute and storage
- Convergence of warehouse and lake, either extending data warehouse to include embedded ML training, e.g. Google BigQuery ML, or alternatively build data

warehouse integrity, transactionality and querying systems into data lake solutions, e.g., Databricks Lakehouse

The third generation data platform is addressing some of the gaps of the previous generations such as *real-time data analytics*, as well as *reducing the cost of managing big data infrastructure*. However it suffers from many of the underlying characteristics that have led to the limitations of the previous generations.

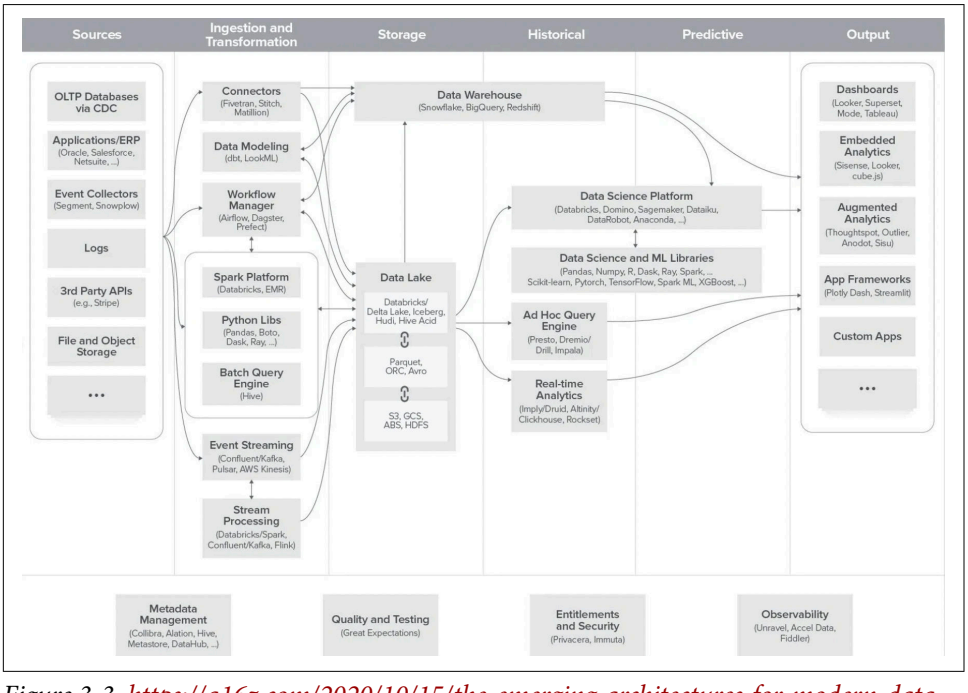


Figure 3-3. https://a16z.com/2020/10/15/the-emerging-architectures-for-modern-data-infrastructure Multimodal data architecture

Characteristics of Analytical Data Architecture

From the quick glance at the history of analytical data management architecture, it is apparent that the architecture has gone through evolutionary improvements. The technology and products landscape in support of the data management have gone through a cambrian explosion and continuous growth. The dizzying view of First-Mark's[1] annual landscape and "state of the union" in big data and AI, is an indication of the sheer number of innovative solutions developed in this space.

1 An early-stage venture capital firm in New York City

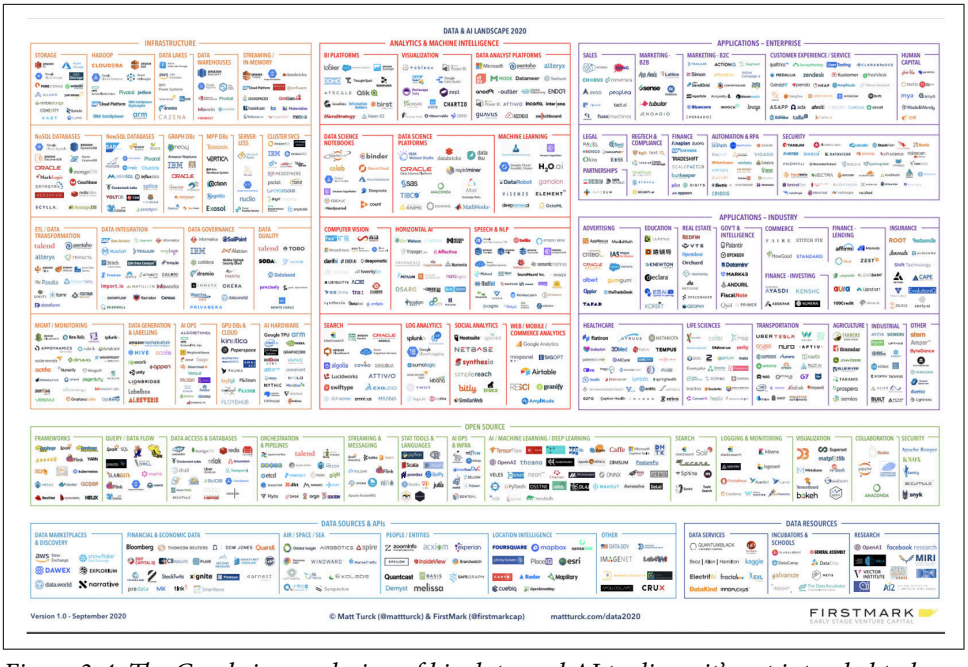


Figure 3-4. The Cambrian explosion of big data and AI tooling - it's not intended to be read, just glanced and feel dizzy Courtesy of FirstMark

So the question is, what hasn't changed? What are the underlying characteristics that all generations of analytical data architecture carry? Despite the undeniable innovation, there are fundamental assumptions that have remained unchallenged for the last few decades and must be closely evaluated:

- Data must be centralized to be useful - managed by a centralized organization, with an intention to have an enterprise-wide taxonomy.
- Data management architecture, technology and organization are monolithic.
- The enabling technologies dictate the paradigm - architecture and organization.

The architectural characteristics discussed in this chapter, including centralization, are only applied to the logical architecture. Physical architecture concerns such as where the data is physically stored - whether it is physically collocated or not - is out of scope for our conversation, and it's independent of the logical architecture concerns. The logical architecture focuses on the experience layer of the data developers and consumers. Such as whether data is being managed by a single team or not - data ownership - whether data has a single schema or not - data modeling - and whether a change on one data model has tight coupling and impact on downstream users - dependencies.

Let's look a bit more closely at each of these underlying assumptions and the limitations each impose.

Monolithic

> Architecture styles can be classified into two main types: monolithic (single deployment unit of all code) and distributed (multiple deployment units connected through remote access protocols)
>
> —Fundamentals of Software Architecture

Monolithic Architecture

At 30,000 feet the data platform architecture looks like Figure 2-7 below; a monolithic architecture whose goal is to:

- Ingest data from all corners of the enterprise and beyond, ranging from operational and transactional systems and domains that run the business, to external data providers that augment the knowledge of the enterprise. For example in the case of Daff Inc., the data platform is responsible for ingesting a large variety of data: the 'media players performance', how their 'users interact with the players', 'songs they play', 'artists they follow', 'labels' and 'artists' that the business has onboarded, the 'financial transactions' with the artists, and external market research data such as 'customer demographic' information.
- Cleanse, enrich, and transform the source data into trustworthy data that can address the needs of a diverse set of consumers. In our example, one of the transformations turns the 'user clicks stream" to 'meaningful user journeys' enriched with details of the user. This attempts to reconstruct the journey and behavior of the user into an aggregate longitudinal view.
- Serve the datasets to a variety of consumers with a diverse set of needs. This ranges from data exploration, machine learning training, to business intelligence reports. In the case of Daff Inc., the platform must serve 'media player's near real-

time errors' through a distributed log interface and at the same time serve the batched aggregate view of a particular 'artist played record' to calculate the monthly financial payments.

Figure 3-5. The 30,000 ft view of the monolithic data platform

While a monolithic architecture can be a good and a simpler starting point for building a solution - e.g. managing one code base, one team - it falls short as the solution scales. The drivers we discussed in Chapter 1, organizational complexity, proliferation of sources and use cases, create tension and friction on the architecture and organizational structure:

- Ubiquitous data and source proliferation: As more data becomes ubiquitously available, the ability to consume it all and harmonize it in one place, logically, under the control of a centralized platform and team diminishes. Imagine the domain of 'customer information'. There are an increasing number of sources inside and outside of the boundaries of the organization that provide information about the existing and potential customers. The assumption that we need to ingest and harmonize the data under a central customer master data management to get value, creates a bottleneck and slows down our ability to take advantage of diverse data sources. The organization's response to making data available from new sources slows down as the number of sources increases.
-
- Organizations' innovation agenda and consumer proliferation: Organizations' need for rapid experimentation introduces a larger number of use cases that consume the data from the platform. This implies an ever growing number of transformations to create data - aggregates, projections and slices that can satisfy the test and learn cycle of innovation. The long response time to satisfy the data consumer needs has historically been a point of organizational friction and remains to be so in the modern data platform architecture. The disconnect between peo-

ple and systems who are in need of the data and understand the use case, from the actual sources, teams and systems, who originated the data and are most knowledgeable about the data, impedes the company's data-driven innovations. It lengthens the time needed to access the right data, and becomes a blocker for hypothesis-driven development.

-
- Organizational complexity: Adding a volatile and continuously shifting and changing data landscape - data sources and consumers - to the mix, is when a monolithic approach to data management becomes a synchronization and prioritization hell. Aligning the priorities and activities of the continuously changing data sources and consumers, with the capabilities and priorities of the monolithic solution - isolated from the sources and consumers - is a no-win situation.

Monolithic Technology

From the technology perspective, the monolithic architecture has been in a harmonious accordance with its enabling technology; technologies supporting data lake or data warehouse architecture, by default, assume a monolithic architecture. For example, data warehousing technologies such as Snowflake, Google BigQuery, or Synapse, all have a monolithic logical architecture - architecture from the perspective of the developers and users. While at the physical and implementation layer there has been immense progress in resource isolation and decomposition - for example Snowflake separates compute resource scaling from storage resources and BigQuery uses the latest generation distributed file system - the user experience of the technology remains monolithic.

Data Lake technologies such as object storage and pipeline orchestration tools, can be deployed in a distributed fashion. However by default, they do lead to creation of monolithic lake architectures. For example, data processing pipeline DAG definition and orchestrations' lack of constructs such as interfaces and contracts abstracting pipeline jobs dependencies and complexity, leads to a big ball of mod monolithic architecture with tightly coupled labyrinthic pipelines, where it is difficult to isolate change or failure to one step in the process. Some cloud providers have limitations on the number of lake storage accounts, having assumed that there will only be a small number of monolithic lake setups.

Monolithic Organization

From the organizational perspective, Conway's Law has been at work and in full swing with monolithic organizational structures - business intelligence team, data analytics group, or data platform team - responsible for the monolithic platform, its data and infrastructure.

> Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure.
>
> —Melvin Conway, 1968

When we zoom close enough to observe the life of the people who build and operate a data platform, what we find is a group of hyper-specialized data engineers siloed from the operational units of the organization; where the data originates or where it is used. The data engineers are not only siloed organizationally but also separated and grouped into a team based on their technical expertise of data tooling, often absent of business and domain knowledge.

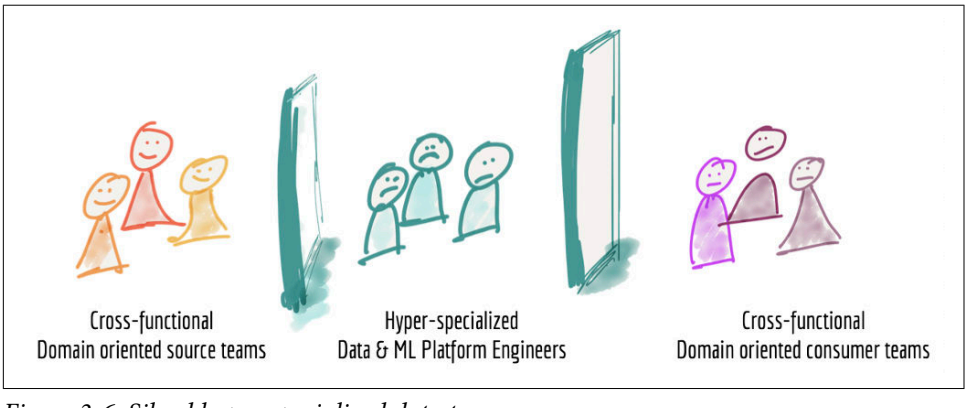


Figure 3-6. Siloed hyper-specialized data team

I personally don't envy the life of a data engineer. They need to consume data from operational teams who have no incentive in providing meaningful, truthful and correct data, based on an agreed-upon contract. Given the data team's organizational silo, data engineers have very little understanding of the source domains that generate the data and lack the domain expertise in their teams. They need to provide data for a diverse set of needs, operational or analytical, without a clear understanding of the application of the data and access to the consuming domain's experts.

For example at Daff Inc., on the source side we have a cross-functional 'media player' team that provide signals of how users interact with media player features e.g. 'play song events', 'purchase events', and 'play audio quality'; and on the other end sit a cross-functional consumer team such as 'song recommendation' team, 'sales team' reporting sales KPIs, 'artists payment team' who calculate and pay artists based on play events, and so on. Sadly, in the middle sits the data team that through sheer effort provides analytical data on behalf of all sources and to all consumers.

In reality what we find are disconnected source teams, frustrated consumers fighting for a spot on top of the data team's backlog and an over stretched data team.

The complicated monolith

Monolithic architectures when they meet scale - here, scale in diversity of sources, consumers, and transformations - all face a similar destiny, becoming a complex and difficult to manage system.

The complexity debt of the sprawling data pipelines, duct-taped scripts implementing the ingestion and transformation logics, the large number of datasets - tables or files - with no clear architectural and organizational modularity, and thousands of reports built on top of those datasets, keeps the team busy paying the interest of the debt instead of creating value.

In short, a monolithic architecture, technology and organizational structure is not suitable for analytical data management of large scale and complex organizations.

Centralized

It's an accepted convention that the monolithic data platform hosts and owns the data that belongs to different domains, e.g. 'play events', 'sales KPIs', 'artists', 'albums', 'labels', 'audio', 'podcasts', 'music events', etc.; collected from a large number of disparate domains.

While over the last decade we have successfully applied domain driven design and bounded context to the design of our operational systems to manage complexity at scale, we have largely disregarded the domain driven design paradigm in a data platform. DDD's strategic design introduces a set of principles to manage modeling at scale, in a large and complex organization. It encourages moving away from a single canonical model to many bounded contexts' models. It defines separate models each owned and managed by a unit of organization. It explicitly articulates the relationships between the models.

While operational systems have applied DDD's strategic design techniques toward *domain-oriented data ownership*, aligning the services and their data with existing business domains, analytical data systems have maintained a *centralized data ownership outside of the domains*.

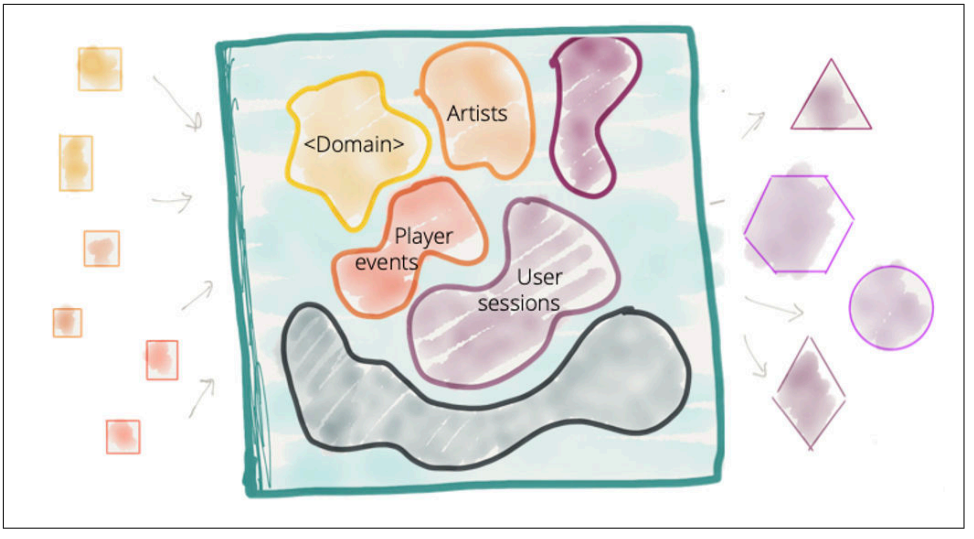


Figure 3-7. Centralization of data with no clear data domain boundaries and domain-oriented ownership of data

While this centralized model can work for organizations that have a simpler domain with a smaller number of consumption cases, it fails for enterprises with rich and complex domains.

In addition to limitations to scale, other challenges of data centralization include providing quality data that is resilient to change; data that is as closely as possible is reflective of the facts of the business with integrity. The reason for this is that business domains and teams who are most familiar with the data, who are best positioned to provide quality data right at the source, are not responsible for data quality. The central data team, far from the source of the data and isolated from the domains of the data, is tasked with building quality back into the data through data cleansing and enriching pipelines. Often, the data that pops out of the other end of the pipelines into the central system loses its original form and meaning.

Centralization of the analytical data has been our industry's response to the siloed and fragmented data, commonly known as Dark Data. Coined by Gartner, Dark Data refers to the information assets organizations collect, process and store during regular business activities, but generally fail to use for analytical or other purposes.

Technology driven

Looking back at different generations of analytical data management architectures, from warehouse to lake and all on the cloud, we have heavily leaned on a technology-driven architecture. A typical solution architecture of a data management system merely wires various technologies, each performing a technical function, a piece of an

end to end flow. This is evident from a glance at any cloud provider's modern solution architecture diagram, like the one below. The core technologies listed below are powerful and helpful in enabling a data platform. However, the proposed solution architecture decomposes and then integrates the components of the architecture based on their technical function and the technology supporting the function. For example, first we encounter the *ingestion* function supported by Cloud Pub/Sub, then *publishing data* to Cloud Storage which then *serves data* through BigQuery. This approach leads to a *technically-partitioned architecture* and consequently *an activity-oriented team decomposition.*

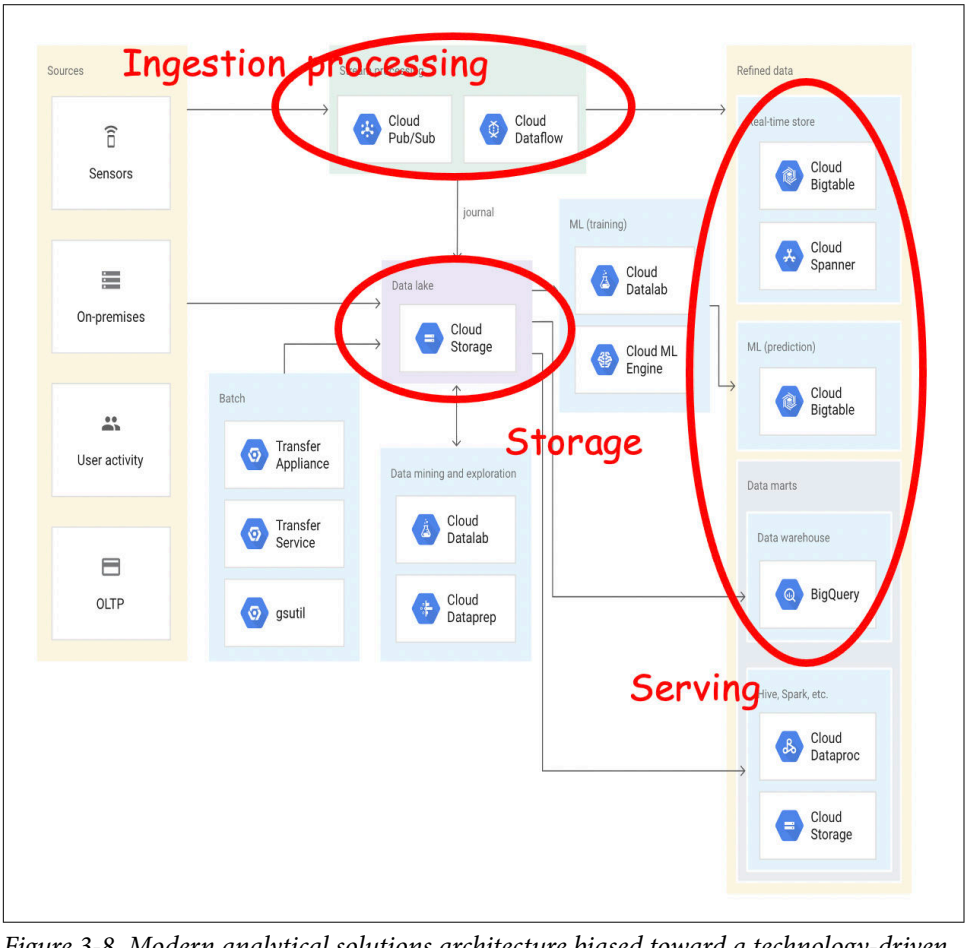


Figure 3-8. Modern analytical solutions architecture biased toward a technology-driven decomposition - example from GCP https://cloud.google.com/solutions/build-a-data-lake-on-gcp

Technically-Partitioned Architecture

One of the limitations of data management solutions today, comes down to how we have attempted to manage its unwieldy complexity; how we have decomposed an ever-growing monolithic data platform and team to smaller partitions. We have chosen the path of least resistance, a *technical partitioning for the high level architecture.*

Architects and technical leaders in organizations decompose an architecture in response to its growth. The need for on-boarding numerous new sources, or responding to proliferation of new consumers requires the platform to grow. Architects need to find a way to scale the system by breaking it into its top-level components.

Top-level technical partitioning, as defined by Fundamentals of Software Architecture, decomposes the system into its components based on their technical capabilities and concerns; it's a decomposition that is closer to the implementation concerns than business domain concerns. Architects and leaders of monolithic data platforms have decomposed the monolithic solutions based on a pipeline architecture, into its technical functions such as *ingestion*, *cleansing*, *aggregation*, *enrichment, and serving.* The top-level functional decomposition leads to synchronization overhead and slow response to data changes, updating and creating new sources or use cases. An alternative approach is a *top-level domain-oriented top-level partitioning,* where these technical functions are embedded to the domain, where the change to the domain can be managed locally without top-level synchronization.

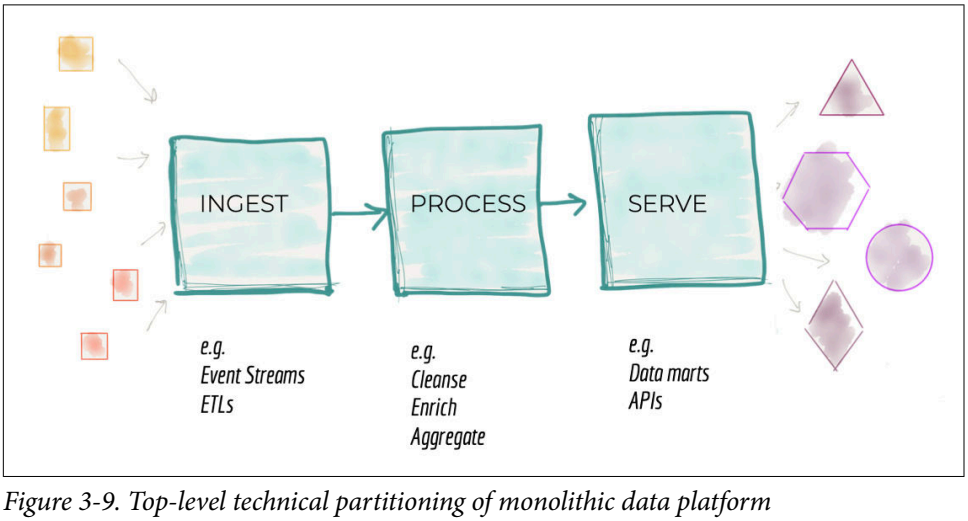


Figure 3-9. Top-level technical partitioning of monolithic data platform

Activity-oriented Team Decomposition

The motivation behind breaking a system down into its architectural components is to create independent teams who can each build and operate an architectural component. These teams in turn can parallelize work to reach higher operational scalability and velocity. The consequence of top-level technical decomposition is decomposing teams into *activity-oriented groups*, each focused on a particular activity required by a stage of the pipeline. For example, a team focusing on *ingestion* of data from various sources or a team responsible for *serving* the lakeshore marts. Each team attempts to optimize their activity, for example find patterns of *ingestion*.

Though this model provides some level of scale, by assigning teams to different activities of the flow, it has an inherent limitation that does not scale what matters: delivery of outcome - in this case, delivery of new quality and trust-worthy data. Delivering an outcome demands synchronization between teams and aligning changes to the activities. Such decomposition is *orthogonal to the axis of change or outcome*, and slows down the delivery of value and introduces organizational friction.

Conversely, an outcome-oriented team decomposition, optimized for achieving an end to end outcome fast with low synchronization overhead.

Let's look at an example. Daff Inc. started its services with 'songs' and 'albums', and then extended to 'music events', 'podcasts', and 'radio shows'. Enabling a single new feature, such as visibility to the 'podcasts play rate', requires a change in all components of the pipeline. Teams must introduce new ingestion services, new cleansing and preparation as well as served aggregates for viewing podcast play rates. This requires synchronization across implementation of different components and release management across teams. Many data platforms provide generic and configuration-based ingestion services that can cope with extensions such as adding new sources easily or modifying the existing sources to minimize the overhead of introducing new sources. However this does not remove an end to end dependency management of introducing new datasets from the consumer point of view. The smallest unit that must change to cater for a new functionality, unlocking a new dataset and making it available for new or existing consumption, remains to be the whole pipeline - the monolith. This limits our ability to achieve higher velocity and scale in response to new consumers or sources of the data.

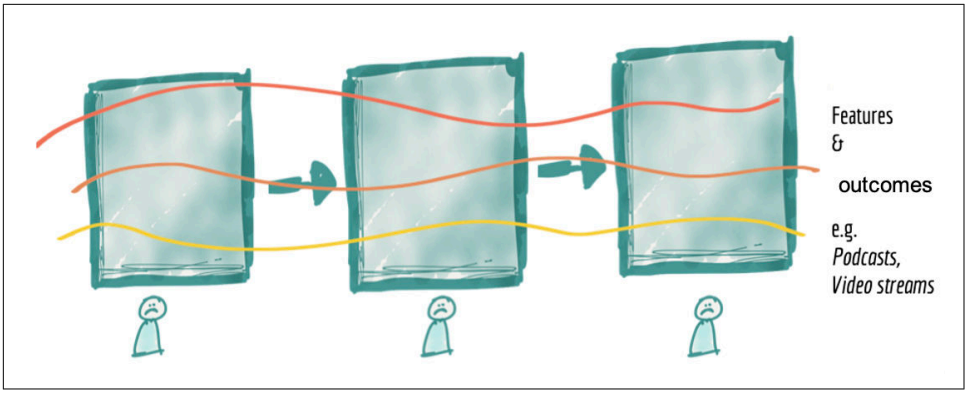


*Figure 3-10. Architecture decomposition is **orthogonal to the axis of change (outcome)** when introducing or enhancing features, leading to coupling and slower delivery*

We have created an architecture and organization structure that does not scale and does not deliver the promised value of creating a data-driven organization.

Recap

> The definition of insanity is doing the same thing over and over again, but expecting different results.
>
> —Albert Einstein

You made it, walking with me through the evolution of analytical data management architecture. We looked at the current state of the two-plane division between operational data and analytical data, and their fragile ETL-based integration model. We dug deeper into the limitations of analytical data management; limitations to scale - organizational scale in expansion of ubiquitous data, scale in diversity of usage patterns, scale in dynamic topology of data and need for rapid response to change. We looked critically into the root causes of their limitations.

The angle we explored was architecture and its impact on the organization. We explored the evolution of analytical data architectures from data warehousing, data lake to multi-modal warehouse and lake on the cloud. While acknowledging the evolutionary improvement of each architecture, we challenged some of the fundamental characteristics that all these architectures share: monolithic, centralized and technology driven. These characteristics are driven from an age-old assumption that to satisfy the analytical use cases, data must be extracted from domains, and consolidated and integrated under central repositories of a warehouse or a lake. This assumption was valid when the use cases of data were limited to low-frequency reports; it was valid when data was being sourced from a handful of systems. It is no longer valid when data gets sources from hundreds of microservices, millions of devices, from

within and outside of enterprises. It is no longer valid that use cases for data tomorrow are beyond our imagination today.

We made it to the end of Part I. With an understanding of the current landscape and expectations of the future, let's move to Part II and unpack what Data Mesh is based on its core principles.