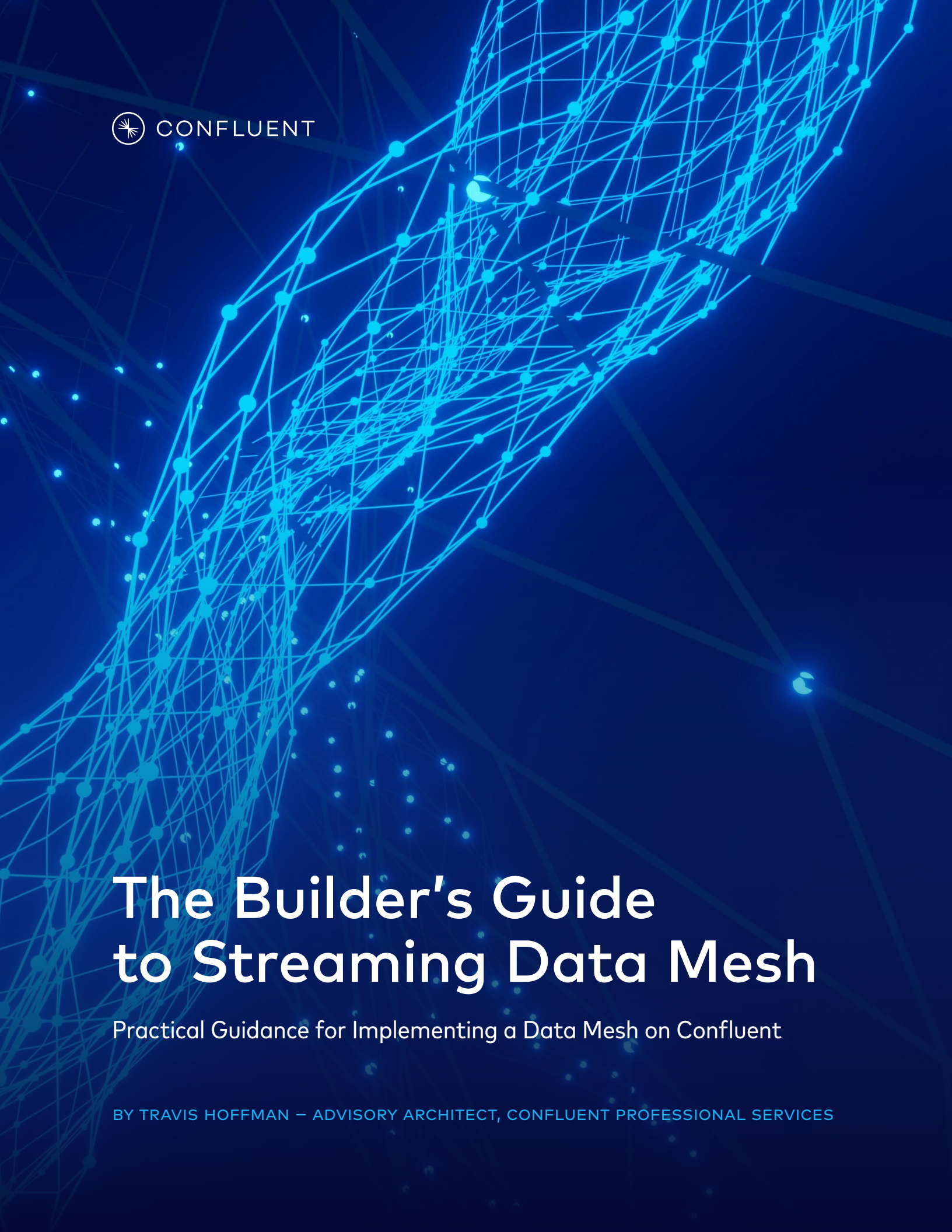


The Builder's Guide to Streaming Data Mesh

Practical Guidance for Implementing a Data Mesh on Confluent

BY TRAVIS HOFFMAN – ADVISORY ARCHITECT, CONFLUENT PROFESSIONAL SERVICES

Contents

Introduction

MUCH HAS BEEN written about data mesh—why it's valuable, why it's important, and what it is—but very little has been written for the practitioner on how to implement data mesh. This is partially because data mesh is very new and there are few mature example implementations of data mesh in the wild to draw upon. This document aims to be a practical guide for architects and technology thought leaders who seek to implement their own data mesh.

I have had the good fortune of working with several customers at different phases in their data mesh implementation journey. From their varied experiences, I've developed a composite perspective that yields practical trends and insights into the adoption of data mesh in practice. I've been able to see what works well, what doesn't work well, and most importantly how to get started. I've learned that some concepts—like Data as a Product—sell themselves, others are much more difficult to sell; to ensure success, I've included guidelines for how to make the organizational and technical changes you're going to need to make to implement a successful data mesh.

While this document provides a practical guide for those who seek to build their own data mesh with Confluent Data Streaming Platform, there are many aspects of implementing data mesh that are technology agnostic. This guide is designed so you can easily tell which parts are specifically about data streaming and which are general to data mesh.

What Is Data Mesh?

WHEN FIRST LEARNING about data mesh, one of the hardest things is to come up with a clear and concise definition. Here's how the creator of the term data mesh Zhamak Dehghani describes it:

> *"...Data mesh is a decentralized sociotechnical approach to share, access, and manage analytical data in complex large-scale environments—within or across organizations."*
>
> ZHAMAK DEHGHANI, DATA MESH, CHAPTER 1

One word usually needs special attention; sociotechnical means:

> *Having both sociological and technological aspects.*

The key takeaway is that any data mesh implementation requires a solution that considers the people, process, and technology aspects.

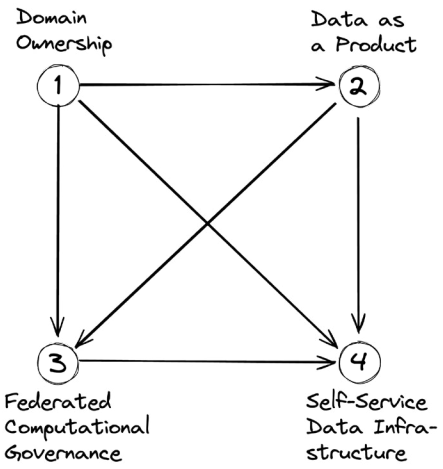
Is Data Mesh Only for Analytical Data?

Another important point about data mesh is that there is a debate surrounding whether data mesh can only be for analytical data. In the literature and online forums there is certainly a strong preference by purists that data mesh be only for analytical data for a variety of reasons. In my experience, every practical business use case of data mesh wants to combine their operational and analytical data in one system for the simple reason that businesses want to 1) gather analytical data from their operational systems, 2) use operational system actions and data to make analytical insights, and 3) make analytics-driven operational decisions from these insights.

The primary argument against combining operational and analytical data is that the two have very different shapes of data and that operational data is much more "messy." This is certainly a true observation, and having the two together in a common platform will make it tempting to blend them in inadvisable ways. This is one reason I advocate for what I call "Balanced Governance," a blended governance model which centralizes some things and federates as much as possible while still achieving the functional and nonfunctional objectives of the organization. Under Balanced Governance, such concerns could be centrally managed and provided with appropriate oversight. This approach to governance will ensure the right mix of oversight is used to maintain the correct separation of the two types of data in the logical layer.

In my experience, I have found there are no reasons that a data mesh cannot also encompass operational data in the same physical layer. There are good reasons to maintain a separation of operational and analytical data at the logical layer of your data mesh. It requires additional discipline to keep the two kinds of data cleanly delineated but, in my opinion, the benefits and conveniences of having all your data in a common physical layer substrate substantially outweigh the risks and concerns of having the two so readily available.

The Four Principles of Data Mesh



These four principles are widely accepted to indicate that you have indeed built a data mesh. There are many excellent articles that focus on the meaning of the principles of data mesh. I encourage you to read Zhamak Deghani's original article Data Mesh Principles and Logical Architecture and take the time you need to reflect on the four principles in depth.

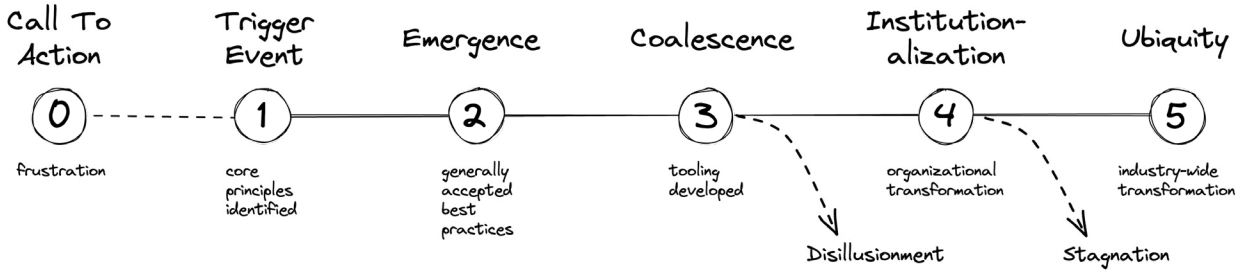
1. Local Domain-Driven Ownership of Data
2. Data as a First-Class Product
3. Federated Computational Governance
4. Self-Service Data Platform

What Is a Data Product?

Imagine all the entities in your business—customers, accounts, claims, inventory, shipments, etc.—each of these is a data product. They are continuously enriched, continuously governed, and continuously shared so that every team in the enterprise can have frictionless self-service access to trustworthy data assets and derive value from data the moment it's created. In short, data products are live, refined, and governed data assets that are discoverable, contextualized, trustworthy, and reusable for all operational and analytical use cases.

Data Mesh Is a Technology Movement

Now that I've worked with data mesh for a few years, I believe that data mesh is much more than just an architectural pattern or approach; I believe it is a technology movement with the same characteristics and qualities of the early days of the Agile movement (circa 2004). This perspective is important because we can make some valuable predictions and decisions about the future of data mesh based on this model.



For you, the thought leader and strategist, it is important to go into data mesh with a clear sense of what can be expected from the data mesh "marketplace of ideas" at its current state of evolution. At the time of this writing, data mesh has been around for barely three years, and is still early in the Emergence phase of the evolution of the movement. There were a flurry of "data mesh" vendors

attempting to jump on the new movement, but I would argue it is too early for these vendors to be successful. Without generally accepted best practices, it is impossible for enterprises to adopt a solution that can solve their data problems, as the requirements are still being defined.

In this emergence phase, we (as the data mesh community) need to focus on identifying best practices that have value for the practical application and implementation of the data mesh strategy. This is what will serve as the basis for the first generation of data mesh platforms and vendor offerings. This means that early adopters will have to do more investigational and prototyping work of tools and techniques until best practices are generally accepted and innovative tooling has been developed for those practices.

For the technology leader it is important to understand the return on investment (ROI) you can expect from data mesh. Data mesh is still in the emergence phase which means that tooling and experiences from most vendors are relatively immature. You are going to have to be a bit more hands-on and expend more DIY effort to implement your vision for your data mesh user experience. That said, you can get a lot of data mesh functionality with Confluent Data Streaming Platform in Stream Governance. All in all, even with the DIY effort, the benefits mentioned outweigh the startup costs and will set you up better for the long run.

Why Data Mesh?

THE ULTIMATE GOAL and value of any data mesh should be to find better, more effective, and more efficient ways to extract value from your organization's data. Data mesh is an option for every organization that has large amounts of data and a business case for realizing value from their data. The benefits of data mesh can be huge for organizations that are able to make the proper investment, but data mesh may not be appropriate for every organization at this stage in the development of data mesh as a movement (more on this later).

The Challenge That Data Mesh Addresses

There is more data than ever, and data growth is accelerating. Historically, data has been "small" and "simple" enough to be handled by a single team within a much larger organization. It was possible for a single team to manage both the technical and social complexity of understanding and managing nearly all domains of its parent organization's data. This is no longer the case; the scale and the complexity of managing all domains of data has grown to the point that it needs to be managed by multiple teams together in a coordinated fashion. How can we scale our data management practices to meet the scale and complexity of the challenge?

The need to make meaningful connections between systems is greater than ever before, leading to greater inter-domain complexity than ever. The requirements for interconnections between lines of business and the need for data integration from disparate sources make it fundamentally intractable for *any* single team to scale to keep up with the complexity of the broader organization. How do we reorient our teams to master the complexity?

Data storage and transportation costs are growing, and greater return on that investment has been promised by IT; "Data is the new oil," we have been told. And it is a good thing too, because we need to justify the expanding costs of storing all this data. How can we bring financial focus and business acumen to our data cost center?

The Benefits That Data Mesh Brings

A successful data mesh requires the consideration of people, process, and technology—and brings benefits to match.

1. Architectural Benefits

The astute student of data mesh will notice that its principles mirror the best ideas and patterns from the application space's monolith-to-microservices architecture transition. Your data architecture can reap many of the same benefits (e.g., efficiency, agility, and adaptability) of domain-driven design by reorganizing your data into a globally defined hierarchy of domains. The primary objective is to better define the boundaries between teams and business units to enable team parallelization and through clearer contracts for intercommunication between the domains and teams.

2. Organizational Benefits

Data mesh seeks to redistribute as much of the work of data engineering as possible back into the hands of average developers to better scale the amount of talent to be brought to bear on data. When combined with new tooling, this increases the pool of resources available to work on solving data-driven problems.

The goal is not to eliminate the data engineering team; rather it is to better leverage and expand data engineering as a skill among more developers in a way very similar to the approach taken by the DevOps movement.

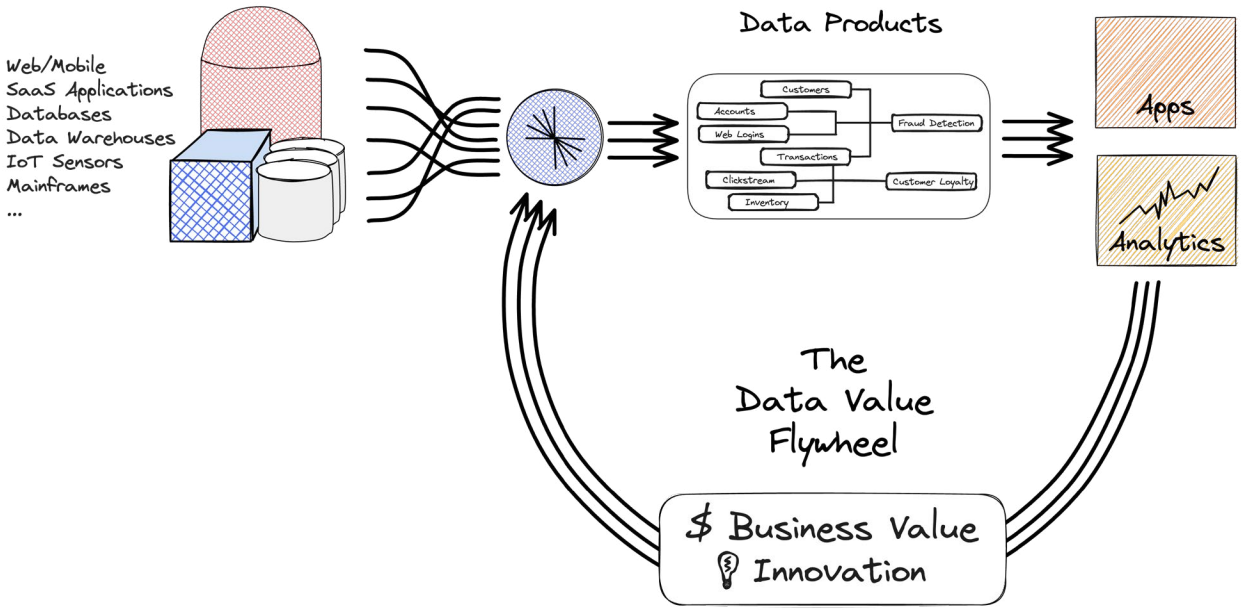
The goal is to refocus the specialist data engineer's effort toward building the tooling, frameworks, and infrastructure that enable self-service for the much larger pool of generalist software engineers in the organization. This expands the amount of talent and velocity available to tackle data challenges.

3. Business Benefits

The first business benefit is shortened time to value realization from better global understanding of data, as it is now owned and managed by the people who are closest to the source. Additionally, product-led thinking works to ascribe value to each data product to enable clear understanding of ROI so prioritization decisions can be data- and value-driven. See the section "Data as a First-Class Product" for more information on data products.

In the process of reorganizing and restructuring the ownership of your data, organizations naturally go through a highly valuable exercise of inventorying, ordering, and documenting their computational lineage—an exercise that has almost always been previously overlooked or one that's sorely out of date. During this process, it is important to carefully consider and assess the value of each data product.

Here's what that could look like in practice:



Data is better captured and better coordinated for analytical use and decision-making. Not only does this make current analytics use cases better, but it unlocks the value of additional derivative use cases that were previously unavailable. Teams can be more agile and react to important business events as they occur. Deeper and broader understanding of your data sources also enables creative reuse of data products for entirely new possibilities.

When implemented on a data streaming platform, you have a solution which sits between and across what is traditionally the analytical-operational divide. With a data streaming platform facilitating the interchange of data, you have a unified "physical layer" that allows data to more easily flow both from the operational estate to analytical estate and back again enabling greater data reuse. Analytical teams benefit by having higher quality data and clearer, better data contracts; operational teams benefit by having more direct access to analytical data products which provide insights that can be used for operational decision-making. This creates what is called the data value flywheel.

This data value flywheel is the ultimate benefit of data mesh, as it truly unlocks the creative power of your entire organization's—not just your analytics estate—ability to leverage your data, often for the first time. Learn how you can implement this.

Data Mesh Is Tech Agnostic

FIRST, DATA MESH is technology agnostic, and allows data to be shared in any format as negotiated for exchange between the producer and consumer. This is very flexible at what we might call the data mesh's logical layer, but at scale this introduces the practical challenge of data reusability and system interoperability at the physical layer. Should data products in the mesh align to just one common format for interchange that's easily transformed by consumers, or should each data product offer multiple data ports that are negotiated per consumer?

Data mesh only addresses the logical definition of implementation; there is a common practical problem that every data mesh must solve: how to move data between producer and consumer. While "data exchange" or "data movement" isn't a core principle, moving data between domains and from producer to consumer is clearly a requirement of any practical solution. How can we most efficiently and effectively maximize the value of data movement?

An additional challenge to solve is the data synchronization problem. When sharing any static data product, the timeliness (or freshness) of that data immediately becomes a concern. Many systems solve this with bitemporal data modeling, an approach that essentially begins to introduce meta information about the system's processing time of the data. What is the most effective way to model the timeliness of data in a large distributed system?

... But Data Streaming Has Key Advantages

Data streaming is the continuous flow of data as it's generated, allowing organizations to respond and adapt to events as they happen. It is the newly emergent category of computation by which data can be moved in more massive quantities (higher throughput) than ever before, efficiently (low latency) and from many producers to many consumers.

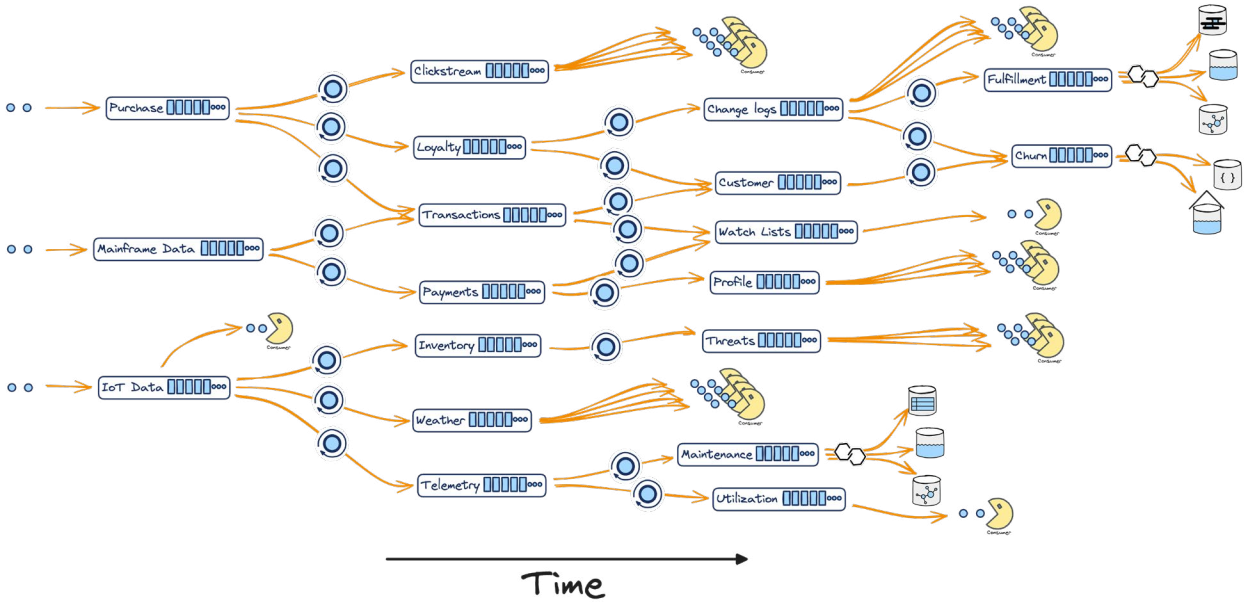
Data streaming systems do not directly replace databases (DBs) and message queues (MQs), but provide some of the best qualities of databases and message queues at scales never before achieved in either. Instead of tables (databases) or queues (message queues), data streaming systems use topics, a log of events, which retain an immutable historical record of what was sent by the data producer in sequential order. Every consumer may independently scan sequentially through as much or as little of this historical record as their own use case logic demands.

There are a few data streaming solutions, but the de facto standard is Apache Kafka®. Originally developed by the three founders of Confluent in 2011, Apache Kafka is an open source distributed event streaming platform known for its unmatched combination of scalability, stability, and features. It has grown to be the most popular Apache Foundation open source project and by far the most widely used streaming platform with estimates of more than 100K active installations globally.

Built on the Kafka foundation, Confluent provides an enterprise-ready, market-leading Data Streaming Platform and enables organizations to stream, connect, process, and govern data as it's set in motion across the enterprise. In particular, Confluent streaming data pipelines help organizations break down data silos and deliver governed real-time data flows that can be shaped on the fly, so different teams can create, discover, and reuse data, whenever and wherever it's needed. Data streaming using Confluent has key advantages when you're building a data mesh:

Data Freshness

Data streaming ensures that data propagates immediately. Near-real-time updates ensure that consumers always have the freshest data, and the contract-driven approach ensures that changes to the stream are compatible so that downstream consumers are not impacted.



Managing data propagation via a series of cascading API calls or data queries is brittle, hard to maintain, and will not be real time. You want your data products to be real time because consumers and the other data products built on them are only as good as their source ingredients—what's upstream. The vast majority of business functions are enhanced or are entirely dependent on fresh real-time data. This is why Apache Kafka, the pioneering tech of data streaming as a category, is so pervasive in industry. Data streaming can serve both real-time and non-real-time use cases; but you can't go the other way around.

Data Scalability

Data streaming is highly scalable; it is capable of supporting the ingress and egress volumes required for any workload and data product. If your goal is to have a large fanout (more consumers than producers), data streaming with Kafka eliminates point-to-point connections and communication bottlenecks. It also allows for in-stream processing, making it easy for the most up-to-date enriched version of the data to be multicast to multiple consumers, allowing data to scale as a product and unlock endless use cases.

Features like infinite storage and tiered storage make it possible to efficiently save vast amounts of data in streams at a bare minimum of cost, while still making it possible to consume those streams as many times as needed with little to no performance degradation.

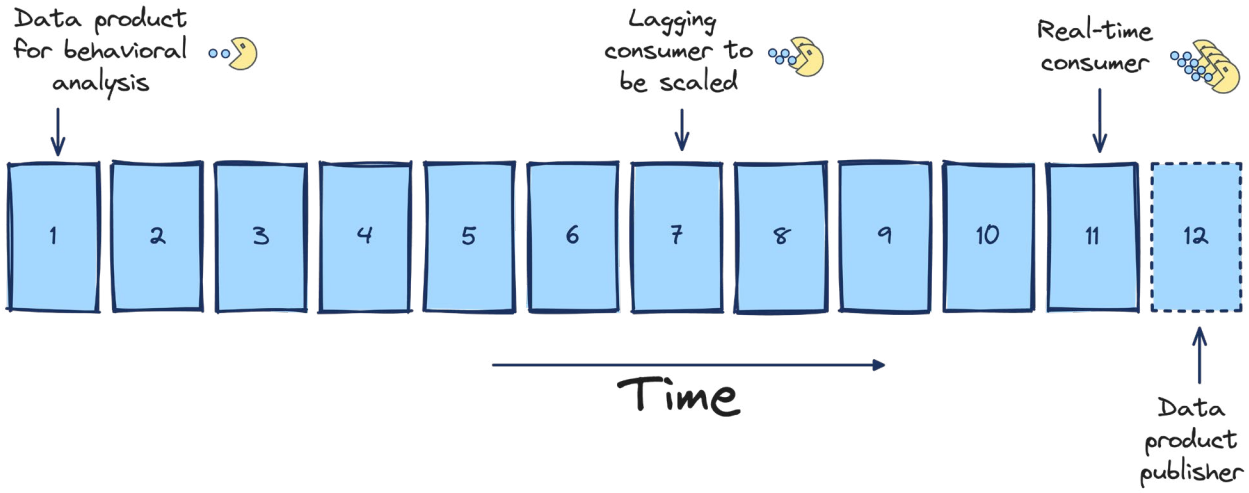
In fact, one can argue that the value of a data product is proportional to its fanout. As Zhamak Deghani says, "We need to change our mindset from data as an asset to data as a product — instead of collecting data to connecting the data together.[1]"

[1] Source

Data Replayability

Data streaming solutions are able to capture real-time and historical data equally well, and make it possible to capture both real-time and historical data with one common tooling.

To unlock the data value flywheel, one challenge with data mesh is that you can't know who your data product consumers will be tomorrow, much less who all of them are today. Furthermore, many use cases require an understanding of how the data product has changed over time. Data streaming is built on this premise; it is a historical log of data as it occurred.



APIs and databases are usually designed for the data model as it exists now, not also as it was, and certainly not for how it will be. Do you have the discipline and desire to build the temporal aspect into all your API and database storage, schema, and access models? Or, does it make more sense to leverage tooling (data streaming) that was built specifically for this purpose?

Data Immutability

Data streaming provides an immutable event store, which means the data in the stream is auditable and may be used as a source of record.

Data Discoverability

Discoverability is made possible through Schema Registry and Data Governance tools including Data Portal, key features of Confluent Data Streaming Platform for coordinating schema management across producers, consumers, tools, connectors, applications, topics, and libraries.

Data Integration

One challenge in data mesh is what is the best—least expensive, simplest, and most manageable—approach for actually propagating data? It has been proposed that each data product may provide multiple "data ports," which provide the product in the form desired by each data consumer.

This approach is perilous and presumes a few conditions are true: that there will be only a few data ports (and the number won't grow), and that consumers can align and agree on the same few formats. Our experience has shown that the more teams and business units are involved, the more federated and distributed the organization and the greater the diversity of tooling, approaches and formats. A team will likely choose a diverging standard unless there are controls to encourage convergence to a standard.

"The nice thing about standards is that there are so many to choose from."

ANDREW S. TANENBAUM

In our experience, this approach leads to excessive complexity for consumers in any practical system. Confluent simplifies data integration with pre-built source and sink connectors that bridge your entire data architecture, delivering the closest plug-and-play experience for deploying streaming data pipelines across an organization.

Data Movement

Systems built on Apache Kafka are world class at the logistics of moving data from *N* sources to *M* destinations.

Data Processing

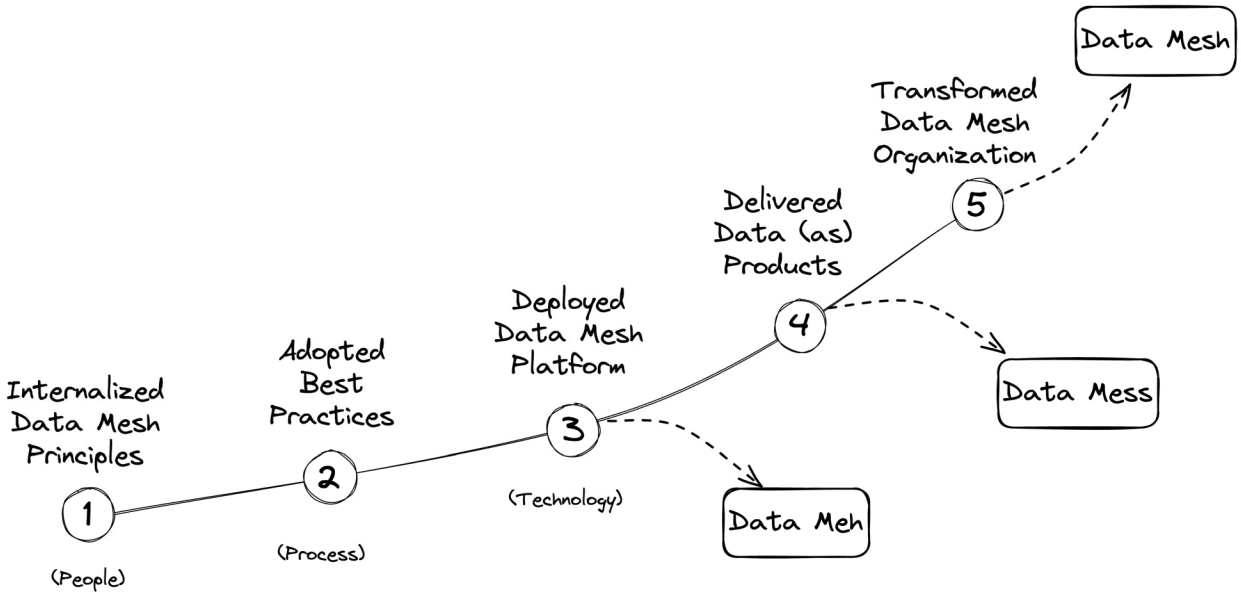
Apache Flink and KStreams provide powerful mechanisms for data transformations. Data lineage provides insights into the flow of data.

Data Governance

Apache Kafka provides the basics of access controls and governance tooling, but this is where the Confluent Data Streaming Platform really shines when it comes to data mesh. Confluent's Stream Governance suite—the only fully managed governance suite for Apache Kafka and data in motion—establishes trust in the data streams moving throughout different environments and delivers an easy, self-service experience for more teams to discover, understand, and put streaming data to work.

Charting the Data Mesh Adoption Journey

I've started to see some general patterns emerge around data mesh through my work with customers. From that, I have developed a five-stage maturity model:



There are three potential outcomes for a data mesh project: "Data Meh," "Data Mess," and "Data Mesh." Here's a look at each stage in more detail.

Stage 1 Internalized Data Mesh Principles

The first milestone every organization passes is when a small group of key individuals take it upon themselves to learn and internalize the data mesh principles. This group typically includes enterprise architects if the initiative is top-down, data architects if the initiative is bottom-up, or some combination of the two. In business terms, this is the **Forming Phase**.

Stage 2 Adopted Best Practices

Once the principles have been internalized in the individuals, it spreads around the organization and begins to seed change, answering the question "how can we make changes today, for free?" In short, this comes in the form of making process and procedural changes.

In our customers, we have seen this happen with the formation of a Center of Excellence (CoE), identification of data product owners, creation of an organization-wide domain-driven map, and generally getting alignment for the practice of "Shifting Left." In business terms, this is the **Storming Phase**.

Stage 3 Deployed Data Mesh Platform

As best practices roll out, a performance gap begins to grow. Individuals begin to feel the pain of inefficiency of using spreadsheets, ad hoc tools, and manual processes to implement their data mesh. These tools may be effective, but eventually the users yearn for more efficient and effective processes.

At this stage, custom tooling is developed, which enables best practices to be implemented and streamlined. This milestone is achieved when the data mesh platform has been delivered, but is still a significant focus of the development of the data mesh. In business terms, this is the **Norming Phase**.

Stage 4 Delivered Data (as) Products

Data products should be delivered early in the maturation process as they should be the primary value measure of a data mesh. This milestone is achieved when the tooling "gets out of the way" and the primary focus is on delivery of data products, not on the development of the tooling that enables delivery of the data products. In business terms, this is the **Performing Phase**.

Stage 5 Transformed Data Mesh Organization

In this last phase, we see a socio shift in the organization, which has started to see the undeniable impact of the data mesh. The data product owner may become a professionalized position. Team structures and budgets may be altered to provide proper support for the platform and a Center of Excellence may be formed to spread best practices organization wide. In business terms, this is the **Transforming Phase**.

The ideal outcome from this journey is, naturally, a well-functioning data mesh. But it's possible to go astray.

Outcome 1 Data Meh

When an organization attempts the "build it and they will come" approach, they disregard demonstrating ROI early in the process and instead focus on developing a full-formed platform before having buy-in from a user base, they risk making a large investment over a longer period of time without having clearly demonstrated value "at the end."

In our experience, it is important to demonstrate the end-to-end value of data products and data mesh as early as possible. This provides clear benefits that non- or less-technical stakeholders can lean on while they wait for the more slowly developing sociotechnical benefits to pay dividends. An important factor for early success is not improvising existing approaches and tools, but to completely redesign from the ground up to truly achieve data mesh. However, many organizations resort to shortcutting the approach, which results in a "data mess."

Outcome 2 Data Mess

Once the organization has turned the corner from developing the data mesh platform and is primarily focused on delivering value in the form of data products, the next risk is that all of these data products being created by a distributed organization will end up in yet another dreaded "spaghetti mess" architecture.

To avoid creating a data mess, it is important that data products are always designed so that they may be easily evolved—this enables rework for the sake of complexity management. Teams must be guided by a central authority who lays out principles that lead to less complexity while enabling federated design autonomy.

Outcome 3 Data Mesh

For an organization to fully (and successfully) realize the data mesh outcome, they must make the organizational changes necessary to align their organizational structure—à la Conway's Law—to one which aligns to the architectural outcome desired with data mesh.

For many organizations, this means following the steps taken in the transition from "over-the-wall IT" to "DevOps," in which ops personnel were embedded in development teams earlier, and therefore better practices in operations that include:

- Coordination without centralization by empowering teams most familiar with the data to be data stewards.
- Frictionless discovery of high-quality data assets.
- Data contracts and on-the-fly enrichment for continually evolving and contextualizing reusable data assets.
- Understanding the data journey and ensuring observability, compliance, and confidentiality of data that's always on the move.

Additionally, I have seen some organizations have started to professionalize their data product ownership roles (sometimes called Data Stewards), parallelling the movement to have professional Product Owners and Scrum Masters.

Each of the four principles of a data mesh has to come to life in your organization to have a successful data mesh. Let's discuss how to implement each of them in detail.

Confluent Is Built for Data Mesh

DOMAIN-DRIVEN OWNERSHIP OF DATA

- Business metadata
- Flink
- ksqlDB
- Kafka Streams
- Connectors
- Run everywhere
- Data Portal
- Data contracts

SELF-SERVE DATA PLATFORM

- Self provision
- OAuth
- RBAC
- Confluent Cloud UI
- Terraform
- Serverless
- Data Portal
- Data contracts
- Expand and shrink
- Multi-tenancy

DATA **AS A PRODUCT**

- RBAC
- Schema Registry
- Data Portal
- Data contracts
- Data quality rules
- Business metadata
- Cluster Linking
- Confluent Cloud 99.99% SLA

FEDERATED GOVERNANCE

- Business metadata
- Tagging
- Schema Registry
- Data Portal
- Data contracts
- Data quality rules
- Stream Lineage
- Stream Catalog
- Data governance APIs

PRINCIPLE 1

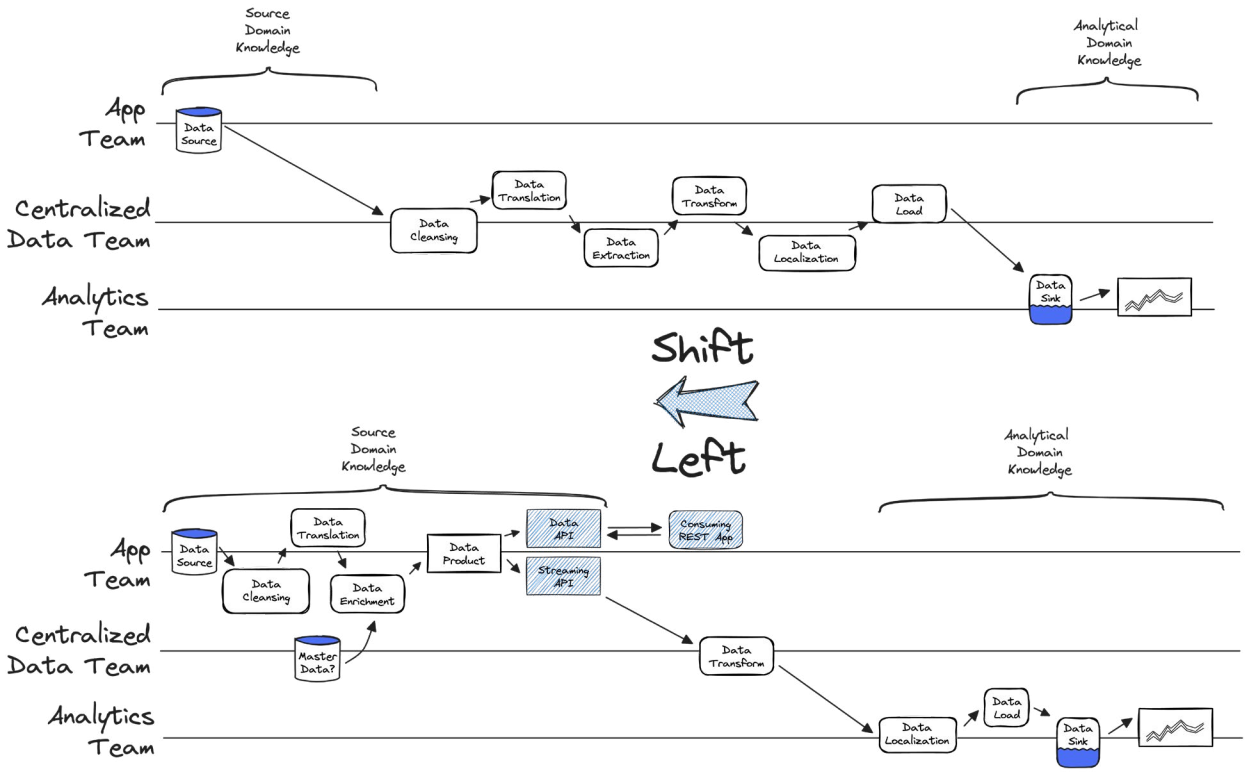
Domain-Driven Ownership of Data

THE FIRST PRINCIPLE is the most radical departure from the status quo of data engineering. Historically, data engineering has been a small, centralized function that supports all domains and lines of business at the organizational level. This was largely because data started "small" at most organizations and has only recently grown "big" and complex enough to outgrow the ability of a single team to manage effectively.

Shifting Left

The primary action customers take when taking on domain-driven ownership of their data is to put more control—and responsibility—of data in the hands of the teams that own and operate the data. If we consider the chain of ownership of the typical analytics pipeline in an organization with a centralized data engineering/science team, historically that centralized team has done the majority (if not all) of the work of creating the pipeline.

In a data mesh approach and decentralized model, "shifting left" guides us to rebalance the work so that more developers from the domain teams (at the source) are able to implement more functional components of the data pipeline whether it's pre-cleaning or transforming data or building in data quality rules while ingesting data into the pipeline, even if with different tools.



Organizational Considerations

When making this shift, it is unlikely (and unreasonable) to assume that every developer is suddenly going to become an expert in the specialized technologies, tools, and techniques that the data engineering team has been using for years. During this time, you will need a strategy to embed data engineers in your application teams to provide mentorship and best practice guidance to minimize missteps. This approach can (and should) be similar to the way DevOps engineers are embedded into the application development process.

One important factor to consider is potential apprehension or opposition from data engineering teams who have decades of practice with existing familiar tools. However, in reality, current approaches are manually intensive and time-consuming. When data is repeatedly duplicated, cleansed, and remodeled amid lack of domain expertise, this leads to higher licensing costs, data quality issues, and a bottleneck around data engineering. The shift left approach for data streaming alleviates many of these challenges and allows organizations to design a system that bridges the operational-analytical data divide and increases data reusability.

As a reminder, the goal is not to replace data engineering, but rather to refocus data engineering talent on enabling the rest of the organization to be able to self-service data work for themselves.

Technology Considerations

Rather than trying to align all technologists to a common toolchain or skill set, our goal is to enable developers to use their familiar tools in the programming languages of their choice along with new, adjacent tools. With help and guidance from data engineering, the goal is for most developers to become enabled to tackle more aspects of data engineering challenges, without the need for specialized training or a need to retool.

Data mesh proponents generally emphasize the need for vendor neutrality when it comes to interchange and interoperability—where interoperable data doesn't rely on centralized authority or controls and there's a simple, flexible, and consistent interface for data interchange across domains. In our experience, this is workable at a small scale because there are a small number of permutations. As the number of connections grows, however, it becomes dramatically more expensive to integrate data sources that have each chosen their own format and mechanism for data exchange.

In the vast majority of our customers' experience, they have found it is best to align to data streaming as the sole common technical standard for pushing state changes across domains. They have found that using a single technical solution creates a virtuous cycle that enables each new data product and connection to have network effects analogous to the value of friendships and friends-of-friends in social networks.

Tens of thousands of organizations have found Kafka to be the perfect foundation for implementing the shift left approach and streaming data between applications for the following reasons:

Scalability – Kafka can scale to handle any workload.

Idempotence – Once written, the data stream can act as an auditable log of the exchange, which is extremely valuable for security and compliance use cases. This also guarantees that any downstream consumer may reread data with confidence.

Replayability – Data streams may act as a system of record without having to consume the data from overloaded source systems, or which may simply be unable to rewind to a specific point in time.

Reactivity – Systems can immediately react to changes in source data, enabling new patterns of behavior, faster decision-making, and more immediate responses to changing circumstances.

Confluent adds many features to open source Kafka that specifically support the needs of domain-driven data ownership:

Connectors, including change data capture, allows data to be easily sourced and sinked directly to or from domains at the edges of the data mesh.

Stream processing (Flink, ksqlDB, and KStreams) enables domain owners to easily build high-quality, reusable data products.

Stream Governance enables teams to build data contracts for better, more refined collaboration and understanding.

Business metadata helps domain owners track their data in ways specific to their use cases.

Data Portal provides self-serve access control to critical data assets where teams can discover, explore, and access relevant data products.

Stream Catalog, available through Confluent Cloud UI and APIs, allows users across teams to collaborate within a centralized, organized library built through automatic metadata collection—including topics, schemas, fields in schemas, clusters, connectors, and more.

Role-based access control (RBAC) to define the users within the organization that have access to sensitive data and bring your own key (BYOK) to protect data at rest using self-managed encryption keys.

Hybrid/multicloud data interoperability—increasingly critical in regulated industries—enables data streaming across any architecture, connecting all your systems, applications, and datastores, whether they're running on premises, in the cloud, or a combination of both.

Leveraging these components within Confluent enables you to reduce complexity within your organization and simplify the data movement process.

Business Benefits

The key business benefits of the shift left approach include:

1. Analytics teams can maximize the ROI of the data stored in cloud data warehouses and data lakes by eliminating compute and storage costs for redundant data that needs to be duplicated, recleaned, and remodeled over and over again. Significantly reduce spend by moving processing and schema management upstream to Confluent.
2. Greater agility and productivity of your data-dependent teams:
    a. Developers across the organization can effortlessly consume data assets without dependency on data engineering teams, enabling greater velocity in developer feature work. Operational efficiency is also improved since data is published by stewards of the data, making it far more trustworthy and reliable.
    b. Instead of building and maintaining bespoke data pipelines for every use case, data engineers can now focus their efforts on higher value initiatives that drive top-line impact.
3. Most organizations are riddled with data debt—data inconsistencies, redundancies, inaccuracies, and immature governance. With a shift left approach, enterprises can draw down this data debt.
4. It drives greater data reuse across the organization with data products. With stream processing and governance moved upstream and the ability to multicast data to numerous destinations—where data is continuously and instantly shaped on the fly—data becomes trustworthy, discoverable, and instantly consumable for every downstream system and application that needs it.

Use Case Example

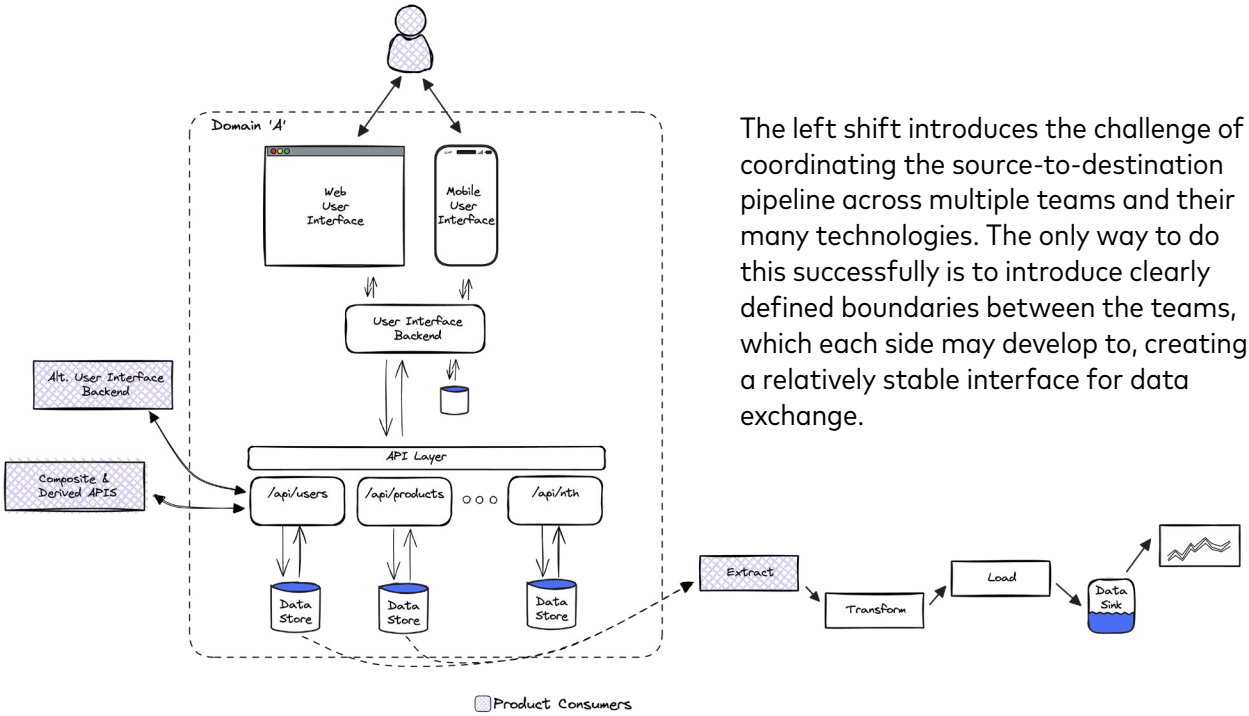
Today, data sources are spread across an organization with vastly different levels of data quality. Extract, transform, load (ETL) jobs send data in batches to a data warehouse. Centralized data ownership can create a bottleneck and a data warehouse team may be required to clean and fix data, without fully understanding the datasets.

In a data mesh, Confluent's decoupled architecture and Stream Governance support decentralized data ownership, which allows data assets to be owned by the local team most familiar with them (e.g., product team, logistics team, billing team, marketing team). Each team is responsible for providing good quality data that can be efficiently used by others. Product team owns catalog and pricing data while the logistics team owns inventory and shipment data, for example. If any issues arise, stakeholders can directly contact the data owner to fix them at the source. Data can be directly streamed from their respective domains and operational data systems into Confluent, processed in flight and ready to be used as soon as it lands in a data warehouse.

Albertsons is one of the largest retailers in the country, with 2,276 stores and 34 million customers and growing. They leverage Confluent for real-time analytics at scale—streaming data across domains into their data lakehouse for supply chain order forecasting, warehouse order management and delivery, labor forecasting and management, demand planning, and inventory management. Without dependency on a centralized data team and high-latency batch ETL, the result is accelerated time to insight for faster, more informed decision-making.

PRINCIPLE 2

Data as a First-Class Product



The left shift introduces the challenge of coordinating the source-to-destination pipeline across multiple teams and their many technologies. The only way to do this successfully is to introduce clearly defined boundaries between the teams, which each side may develop to, creating a relatively stable interface for data exchange.

IN THE TRADITIONAL approach, the centralized Data Engineering team typically has to reach into each domain, learn and master each domain's semantics and models, and then extract data from the operational data store of the applications of that domain, hoping that nothing changes in the operational tables.

This ad hoc approach creates a second-class experience across multiple different dimensions. A change in an application or its schema causes unpredictable cascading failures downstream. Data engineers spend huge amounts of effort building and maintaining hundreds of duplication pipelines for each use case, which is not scalable, not real time, not reliable, ungoverned, and insecure. And while this kind of centralized data engineer approach worked for a period when datasets were small, data needs from lines of businesses were more analytical and data latency was not an issue, operational workloads, that rely on the availability of real-time trustworthy data, cannot wait for the typical amount of time it takes for data to progress through each stage of this waterfall architecture.

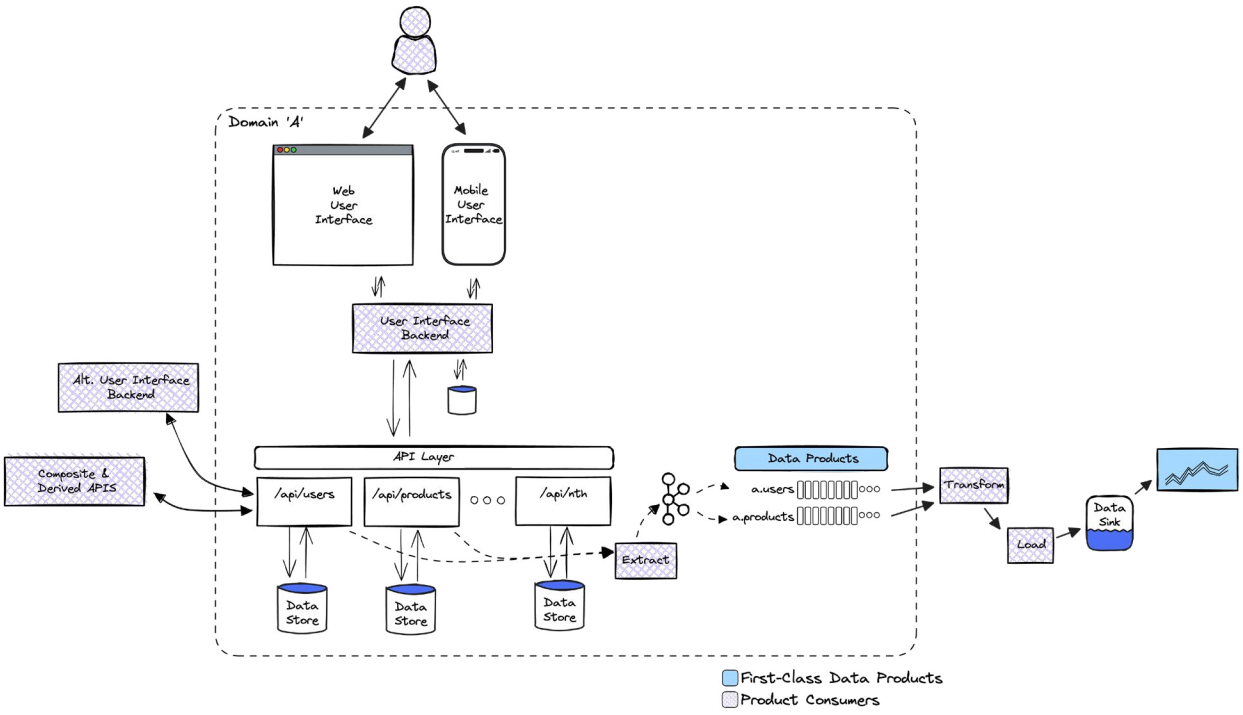
Data Product Thinking

In the data mesh approach, we elevate data to be a first-class citizen, processing it on the fly and ensuring that it is high quality, secure, and reusable, so organizations can accelerate use case delivery. All the entities in your business can be considered a data product—this can include customers, purchases, inventory, shipments, etc. Only instead of querying data from static rows in a database, data products are live assets. They are continuously enriched, continuously governed, and continuously shared for teams to build with data assets faster and unlock the value of data

the moment it's created. Multiple data products can also be mixed and matched, or fused to create derivative data products for endless use cases.

Creating a shared understanding of data involves:

1. Scalable, decoupled architecture as a single source of record.
2. Coordination of data consumers, data producers, and data platform teams without centralization so that each can be autonomous, self-sufficient, and move with agility.
3. Stream processing of data in flight, immediately and continuously combining any stream from anywhere for an enriched view of your data.
4. Data contracts that formalize the expected structure, semantics, and enforcement policies of the data. Data contracts specify the agreed-upon structure, integrity constraints, metadata, rules or policies, and change or evolution. A data contract is implemented using a schema, tags, metadata, and rules.
5. Schema Registry to centrally store data contracts and schemas and manage their evolution.
6. Data Portal to discover and access high-quality data products in a self-service way.



Consider user interactivity that provides value through the functionality and behaviors of the users themselves. In some applications, there is indirect value, for example, by capturing eyeballs so that advertising may be presented to the user. In many others, the value is in directly capturing the user behavior or information entered by the user. The more active users of the system, the more value is derived from their interactions.

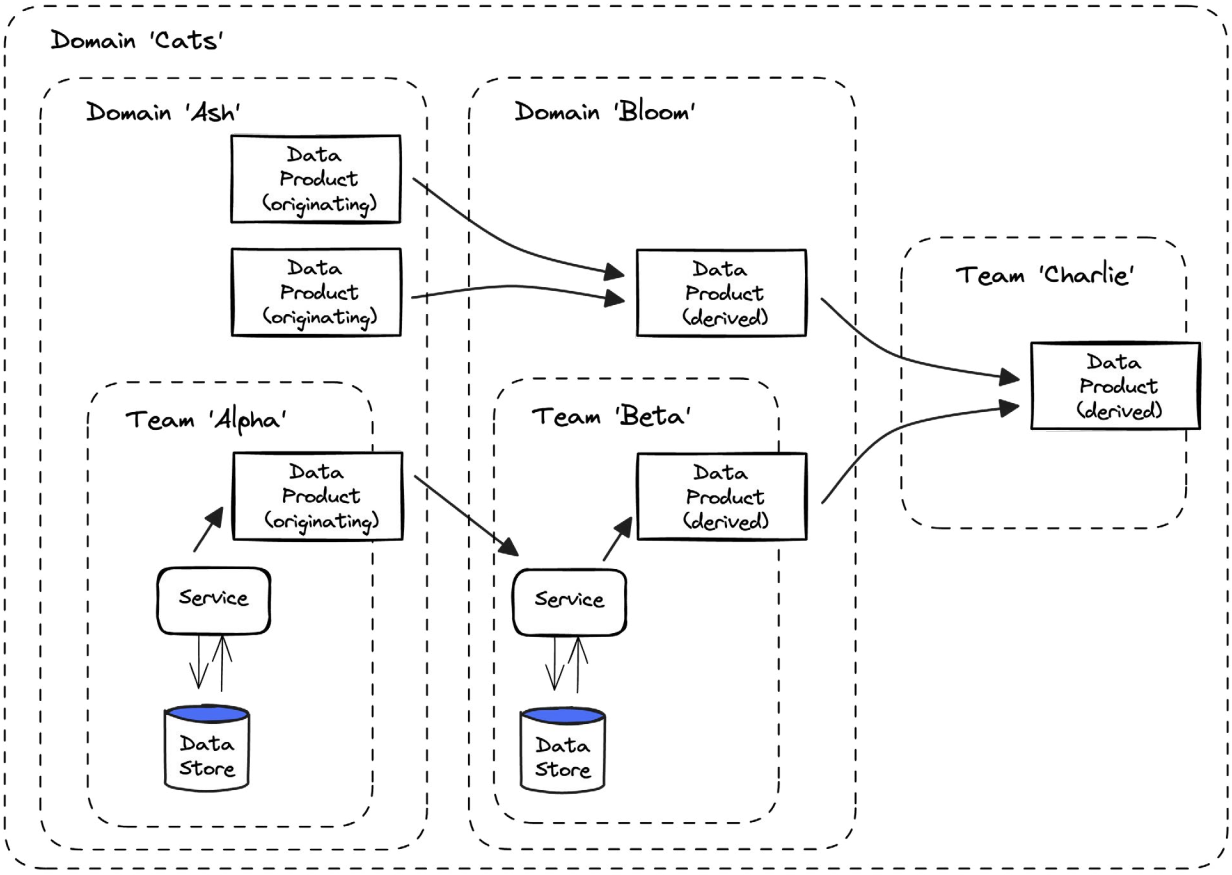
System interactivity provides value through the interactions of the components of the systems themselves. Another system could provide value by responding to requests about the current state of the system; for example, an API that responds to requests for the list of currently active users near a geographic location. The more requesters of information, the more value is derived from the system.

Stream reactivity provides value through the pushing of information and reactive response of consumers. For example, a stock ticker system may provide value by pushing updates of stock prices as the changes happen on a continuous basis to downstream consumers which need to have the most timely information for the best decision-making. The more consumers of this stream of data, the more value is derived from the stream.

Organizational Considerations

The primary challenge when introducing data products for the first time is deciding which product(s) to implement first. We have found that customers who lean into product design thinking have the most success prioritizing their data products. Product design thinking introduces the idea that each (data) product feature should have a value ascribed to it in some way, even if only as a relative measure. The most successful data mesh implementations we've seen have had the mandate to demonstrate value back to the business as quickly as possible. Emphasis on demonstrable value can and should remain a core driver of the data mesh.

It takes a collaborative effort between data product owners (aka data stewards), business, data, and technology partners to engage in meaningful conversation about the business value of individual data products, the cost to implement them, and the relative strategic importance between data products. Introducing this as a discipline to planning helps ensure there is agreement that effort is focused on the data products with the greatest importance and ROI. Maintaining this discipline will ensure that data remains seen as a tangible, highly valuable asset of the organization.



The next class of challenge lies in how to manage the ownership of data products. With original sources of data, this choice is made clear by the Domain-Driven Ownership Principle—the team that owns and operates the source system should also own the data product delivering the data from the originating source system.

What is less clear is the ownership of derived data products—those data products that are built from other data products. There are many philosophical angles on the question of ownership, but from a practical perspective the problem boils down to "Who is responsible for fixing the data product when it's broken?" There is no one-size-fits-all answer to this question; it will need to fit the budgeting, cultural, and political characteristics of your organization. Furthermore, the answer needs to consider the practicalities of the data product's contract with its consumers: its SLAs, data quality guarantees, data restatement policies/approach, etc.

The best answer to this will seek to strike the appropriate balance between the need for velocity and the need for safety in the changes. In some organizations, there is a "just go fix it" attitude and an openness about developers reaching across teams to fix a bug in another team's work. In other organizations, there are proper channels that one must go through for any fix. There may also need to be a range of answers for your organization to find the right balance—some data products will need to be treated with extreme care and others with a more free-wheeling approach. Whatever your solution, be thoughtful about this as it will have ramifications for the ways your teams work together during challenging times.

One customer I worked with developed a solution based on three key aspects:

1. The team that wrote it owns it.
2. The data product contract includes an SLA for bug fixes.
3. Use Confluent's Stream Lineage to provide root cause analysis of who is responsible for what.

Technology Considerations

As architects, when evaluating a technology solution, we must consider not only the direct functionality of the solution, but also the maturity of indirect functionality like tooling support for debugging, monitoring, and analysis which enables development teams to rapidly respond to and resolve problems in their data products. This is part of the TCO calculation of any technology, but must be considered even more carefully with an emerging socio-technical approach like data mesh.

Fortunately, this is one area where the Confluent Data Streaming Platform supports the data mesh journey (even in this nascent form) very well. The Stream Catalog and Data Portal together provide you user-friendly discoverability, sharing, and maximum reuse of your data products. Data contracts are, in my opinion, a game changer for ensuring higher data quality, compatibility between producers and consumers, and better developer experiences.

To ensure proper access to data, role-based access control (RBAC) provides fine-grained control to limit access to the topic level.

In larger organizations, a common challenge is distributing data across multiple clusters or even multiple accounts. Confluent has features for this as well. Cluster Linking vastly simplifies sharing data across clusters.

Taken together, the Confluent Data Streaming Platform provides nearly all the technology features needed to support your streaming data mesh vision.

Business Benefits

The benefits of elevating data to be a first-class citizen include:

1. Your newfound ability to more easily manage your high-quality data products as a ready-to-use reusable asset, across the enterprise. The productization process is like refining data ore; it cleaves away that which hides the valuable information that lies within and makes it more easily consumable in a self-service, frictionless way.
2. Being able to grade the quality of each data product and better assess which data products are worthy of greater investigation and deeper mining. This enables you to have greater, more detailed insight into the value of your data products which allows you to be more efficient in your data investment decisions.
3. Time and cost savings from eliminating duplicative or cumulative efforts that may be more error-prone.

Use Case Example

Trust Bank is Singapore's first digital-native bank with a mission to bring real tangible value to customers through the accessibility and convenience of digital banking. They needed a secure, composable banking system that could expand into the future to support rapid growth.

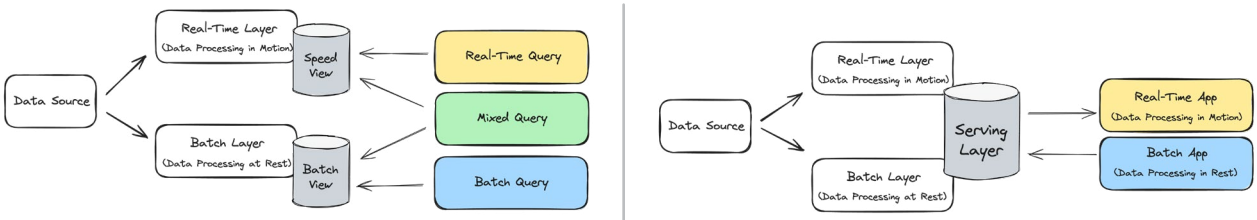
Using Confluent, Trust Bank teams (e.g., marketing, fraud, cards, lending) produce, share, consume, and trust their data products in the form of real-time streams. Teams have explicit ownership around their own data, and downstream services can use and readily harness that data to add and build what they need. For example, their marketing team builds customizable notifications by subscribing to existing customer data available in Confluent generated by other teams. Schema Registry eliminated interservice bottlenecks, allowing Trust Bank developers to seamlessly connect to any data system while maintaining schema compatibility, version control, and quality assurance. The result is greater agility with built-in governance to meet regulations as well as self-service data products to break down data silos and accelerate delivery of innovative new customer experiences.

PRINCIPLE 3

Self-Service Data Platform

THE THIRD PRINCIPLE empowers teams to move as rapidly as possible and infers they must have the tools and processes that have been designed to scale with their needs. Following the inspiration of the DevOps movement, we are called upon to provide data mesh users with an experience that's empowering, that gets out of their way, and that provides enough feedback to yield high confidence in the infrastructure of the system.

Lambda vs. Kappa



The Lambda architecture has one ingestion pipeline for batched data and one for streaming data which basically requires that you maintain two pipelines—one batch and one streaming—while the Kappa architecture ingests all data through a streaming layer and generates batched "samples" of the stream on an as-needed basis. The primary benefit of the Kappa approach is that it unifies all your data processing through a single processing and transformation pipeline.

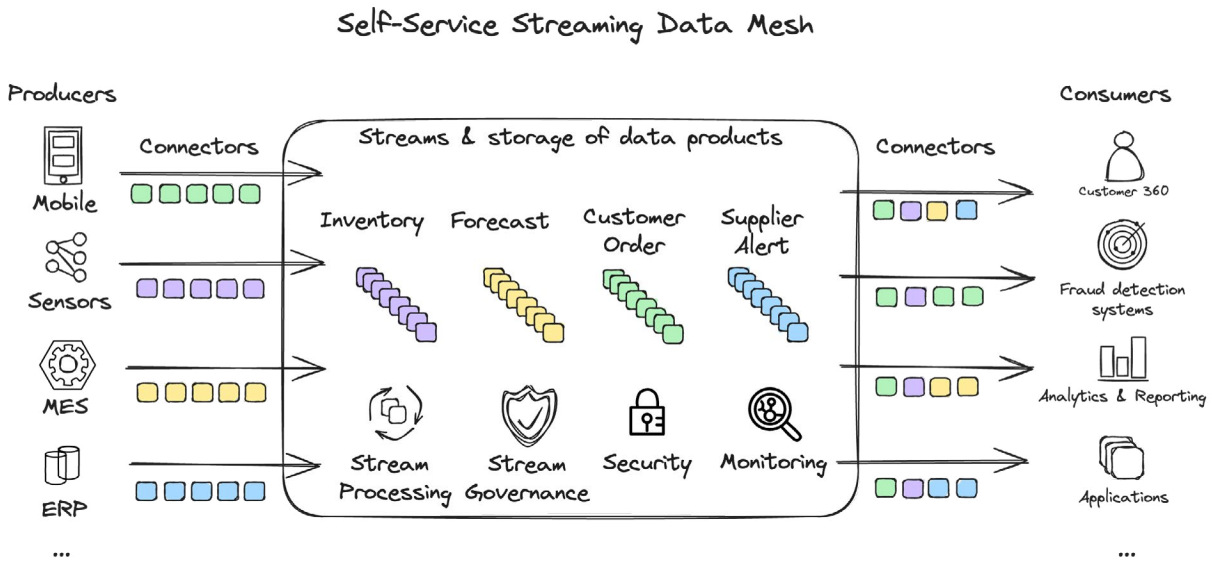
The Kappa architecture is a more natural fit for Kafka-based data systems as it ingests all data through streaming pipelines before distributing the data to downstream systems for real-time, batch, and off-line, and operational processing needs.

The Kappa architecture is replacing the traditional Lambda architecture in most companies because "real-time data beats slow data" according to Kai Waehner. This is primarily because it relieves teams from the complexity of keeping two parallel paths in sync (as in a Lambda).

Organizational Considerations

In the shift toward self-service, each team becomes emboldened and entitled to decide for themselves which is the best source of truth for their particular use case. This is a fundamental shift from the previous central services approach to data engineering in which a central team decides for you. In the previous approach, a single team acts as the arbiter of what data is best for each use case.

In the data mesh approach, each team decides for themselves by selecting in a free (generally open) "marketplace" of data products. The decision making is distributed and bottom-up instead of top-down. To facilitate the speed and agility of this approach, each team needs the tools and information required to be able to make the best decisions.



Technology Considerations

As with DevOps tooling choices, technologies should be chosen that enable integration, automation, and APIs. The goal is to create tooling and experiences for data mesh users which require little-to-no human intervention by a centralized team to complete the most common tasks. If, for example, a team requires a new data asset, they should be able to fulfill the request without approval from a centralized team. The workflow may automatically create the needed resources in pre-production environments to enable development to get started without delay, and then kick off an approval process for the production environment.

Implementing this requires integration with roles/permissions for users in each team, integration with the development environment, and a system to coordinate the request. Confluent provides RBAC and multiple interfaces (APIs, CLI, and UI) to Confluent Cloud to enable granular, secure, and controlled access to data. If you need to implement more complex approval workflows, you will need to implement a service which manages that workflow and tracks the state of approval through the workflow, but the creation of relevant data assets itself can all be handled through API calls.

Provisioning all of this is made both possible and efficient through tools like Terraform and Confluent Cloud's auto-scalability features. Because Confluent Cloud resources are serverless, you can easily self-provision as many or as few resources as are required for each use case. Additionally, Data Portal enables teams to explore and use real-time data products in a self-service way. And whether you're only sharing streams within your organization or to connect with other organizations, it's all configurable within Confluent Cloud UI and with simple API calls.

Providing self-service tools and technologies inevitably means there will be upskilling and a learning curve for some new tooling. Our goal should be to minimize and ease that learning curve as much as possible by introducing as few technologies to the developers as possible, while still seeing as much return on the investment in the tooling created to simplify the experience. Guardrails are important, and are a necessary investment in the future health of your data architecture.

Governing all of these user-created data assets and other resources presents a challenge which requires specialized features. To cope with this, Confluent Cloud provides Stream Governance with data contracts, which go above and beyond schemas by enabling data product evolution controls and more detailed, field-level data quality specifications for your data products.

Business Benefits

The primary business benefits of shifting toward a self-service model include:

1. Freeing up your Data Engineering team from the toil of approving every data connection.
2. Removing the central bottleneck choking the velocity of teams working on data processing pipelines.
3. Enabling more teams and developers to participate in the work of engineering data pipelines.

Use Case Example

BMW Group is the world's leading premium manufacturer of automobiles and motorcycles. With over 31 plants in 15 countries and a global sales network in 140 countries, BMW Group needed to make all IoT data generated by its production facilities and worldwide sales network available in real time to anyone across the business.

Using Confluent, BMW Group implemented a hybrid cloud data architecture where application teams can help themselves to a centrally managed service that facilitates effortless exchange of data between all applications and systems—from ingesting sensor data from the shop floor to continuously transforming, filtering, and enriching data as it moves into Azure cloud services for consumption by different teams and applications. As more machines, robots, and assembly lines produce unprecedented volumes of data, Confluent's fully managed Data Streaming Platform eliminates manual provisioning and data infrastructure management through elastic scaling. Confluent swiftly distributes refined data in real time to any internal consumer, enabling teams at BMW Group to onboard multiple new applications every week, streamline factory production, and create and test new products to fastrack IoT innovation.

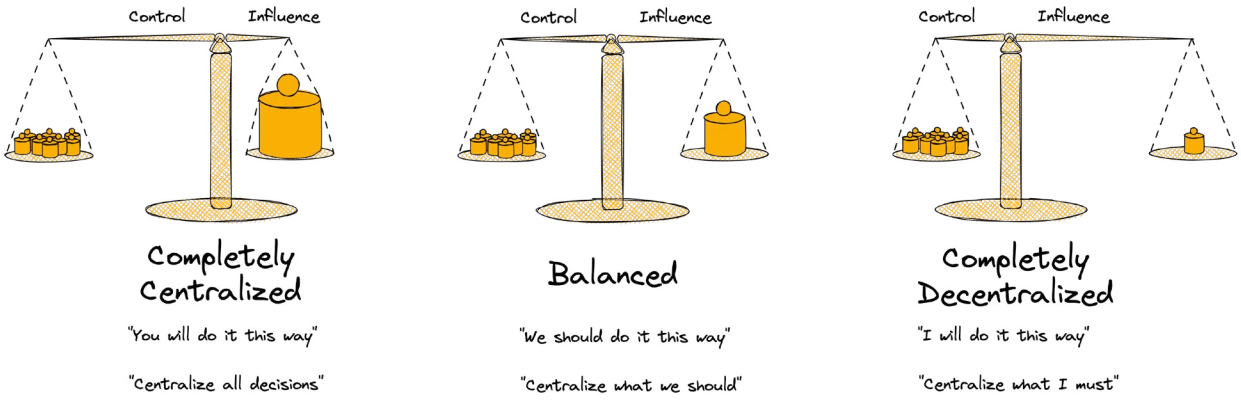
PRINCIPLE 4

Federated Computational Governance

THE FOURTH PRINCIPLE shifts the balance of decision making, infers a change in the balance of ownership, and elevates the importance of computational lineage in the data product. The shift from centralized data ownership to a more decentralized model introduces new challenges in coordination and thus requires new approaches to ensure architectural decisions remain aligned with organization-wide objectives and requirements. Governance must be rethought to remain effective, efficient, and meet business objectives for data.

Balanced Governance

The purist view is that we need to completely federate data product ownership in the data mesh; this however comes with complexities and costs which are, in my customers' experience, simply counterproductive to the practical concerns of coordinating data product ownership, derived data product ownership, and managing cross-cutting concerns such as privacy, regulatory compliance, and differential security.



What is needed is a "semi-planned" approach I call "Balanced Governance," which seeks to strike the right balance between autocratic centralization and federated decentralization on a case-by-case basis. This middle path leverages a representative central authority to ensure longer-term nonfunctional concerns are consistently designed into the system, and focuses on guardrails and common design principles to coordinate the design approach of many federated teams. It maintains scalability by leaving the bulk of decision making to the individually federated teams, while seeking to give them enough information, guidance, and guardrails for them to "make the right choices." For more on this approach see Shift Toward Balanced Governance.

Organizational Considerations

A data mesh without centralized guidance will quickly become yet another "spaghetti mess" architecture of misaligned connections which cannot be easily integrated. A data mesh with too much centralized oversight will grind to a halt. What is the answer?

What is needed is a "semi-planned" approach that centralizes just enough concerns to maintain the most effective balance between velocity, quality, and value. What is needed is for the central team to provide prescriptive guidance which is sufficiently rich and full enough for others to understand and follow while making their own design decisions and yet flexible and adaptable enough to allow for innovation. It is a delicate balance which needs constant refinement driven by a perspective of constant improvement which is informed by all parties to the challenge of both developing and using this prescriptive guidance.

If you're looking for inspiration, I often reflect on the challenge of the United States highway system that has federal level organizations and state level organizations that need to work in collaboration to build, maintain, and manage traffic flows on the vast system of freeway, highways, byways, and routes. The federal organization sets the high-level plans and direction of the overall system as well as some low-level standards for consistent user experience. There is some design oversight to ensure the highways align at state borders, but generally it has been up to each state to each state highway organization to design freeways for themselves, according to their own (federated) state laws and governance. The more central oversight, the slower projects go as they queue for approval. Thus, when I'm part of the data mesh oversight (central authority), I challenge myself to think of the bare minimum of rules and regulations that are needed to achieve the desired outcome from the participants in the broader system.

The best approach to this is a Data Streaming Center of Excellence whose mission is to coordinate oversight of centralized concerns for the data mesh while drawing in the insights and feedback of the organization as a collective whole. The CoE should be a representative body composed of all data mesh participating organizational units which have producers and consumers.

Technology Considerations

Many of the primary concerns of the CoE are technical in nature. According to another law of computer architecture,

> *Every architectural change increases overall system complexity unless it is specifically designed to reduce it.*

One of the primary duties of the CoE is to manage and minimize the complexity of the data mesh. This arises in activities such as combining similar data products into one common data product, setting common technical standards to which data products align, and in ensuring that cross-cutting functional concerns (such as privacy, auditing, or regulatory compliance) are appropriately accounted for in data product designs.

Additionally, the CoE must track and provide continuous guidance to ensure long-term nonfunctional objectives are consistently designed into the system. These concerns include interoperability, observability, traceability, lineage, evolvability, and others. Practitioners and governance can implement oversight for these concerns by carefully applying data contracts, Stream Lineage, and support for tagging and business metadata provided by Confluent's advanced Stream Governance.

Business Benefits

The primary benefits of Balanced Governance are:

1. Rather than dogmatically applying a one-size-fits-all approach, we pragmatically tune the governance of the system on a case-by-case basis to optimize for the best balance between team velocity versus organizational security, regulations, and compliance mandates. (Some requirements must be more centralized controls—those are more tightly controlled, reviewed, and validated to ensure compliance. In practice, these are actually relatively few and far between.)
2. Maximizing velocity for most decisions which can be reasonably decided with little or no central oversight. Most design decisions do not require tight controls and may be left to the discretion of the domain experts themselves to make the best decisions within their requirements, the guardrails of the CoE, and the objectives of the overall data mesh without the need for delays for approval. Validation may be automated with integration tests or relatively lightweight approval processes.
3. We create a culture that engages all stakeholders of the data mesh in the governance process. Every team has the opportunity to participate in the governance of the overall system as part-time or full-time members of review committees or as voting members ratifying changes to the guidelines and guardrails for all teams to follow. In this way, we can build a culture of "us" wherein the data mesh is owned by all participating stakeholders and not "someone else." This mindset may seem unimportant, but my customers have found it invaluable to create a personal sense of ownership in shared infrastructure like the data mesh; it encourages transparency, bias for action, and empowers every developer to "think globally, act locally" when working with the data mesh.

Use Case Example

ACERTUS is an automotive logistics company that must react to real-time events, from order placement to carrier assignment to vehicle pick-up and delivery—with real-time customer notifications throughout the process.

Previously, ACERTUS developers used emails and spreadsheets to manage schemas and found it difficult to guarantee data quality (e.g., checking orders for duplicate VINs). With Confluent, they're able to strike the right balance between decentralized and centralized governance, establishing global standards while enabling autonomous teams. For example, ACERTUS uses Schema Registry to draft schemas and allow the broader team to easily review, comment, or make changes. They can enforce and easily version and track schemas. This eliminated previous back-and-forth and ensured data quality. Additionally, they use Stream Lineage to visualize interactions between microservices and Stream Catalog for discoverability and sharing thousands of schemas. The result is faster development of microservices for real-time customer communications, allowing their quote and order management systems to seamlessly communicate with their fulfillment systems.

Patterns and Practices

Strategize for Success to Get to a Data Mesh

Demonstrate Value Quickly

PERHAPS THE MOST important strategic choice you can make is to emphasize demonstrating the financial or some other tangible value of data mesh as quickly as possible.

Build Tooling to Practices

The best data mesh experiences focus on building and leveraging the tooling that aligns to the practices that have already proven themselves valuable in the real world without the tooling. These first-order best practices are inherently valuable because they enable data mesh practitioners to be successful.

In our experience, projects that build tools based on best intentions are doomed to fail. Data mesh tooling projects that attempt to predict the best workflow for users are making a very risky investment in what is essentially a guess. Our recommendation is to focus effort on the practices that are based on user experiences, user research, and product development best practices.

Simplify Requirements, Lean Toward Data Openness

Many customers begin with the belief that their data mesh must implement very complex rules for data access on all of their data. When pressed on this requirement, it is almost always the case that there is only a small fraction of data that requires limited access. In general, data wants to be free, so a key goal of data mesh is democratization of data.

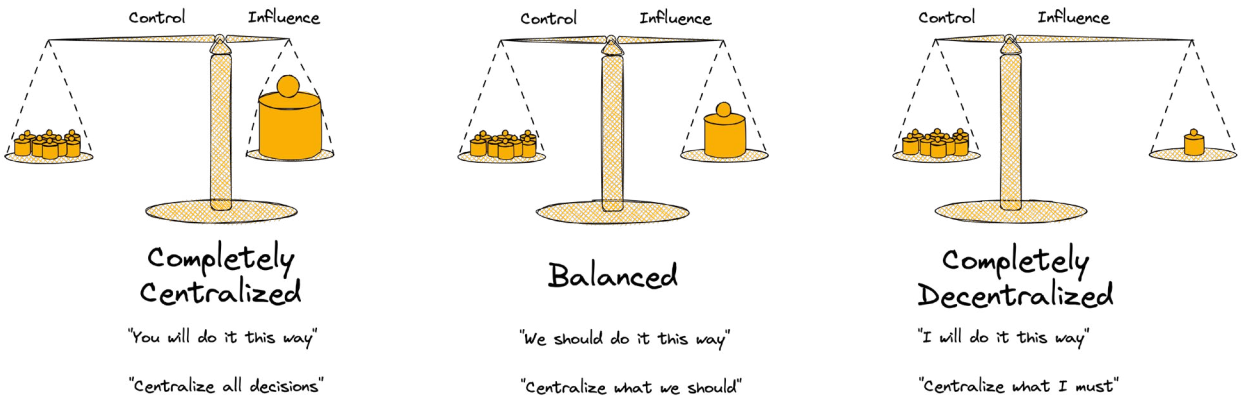
The more freedom of access users have to the data, the more creative uses they will be able to realize. Put another way, restrictions on access to data limit the total addressable market and the barriers to entry for use of a data product. Our recommendation is to start with as much openness as possible, be creative with masking and encryption to allow openness, and then only restrict access where absolutely necessary.

Shift Toward Balanced Governance

What Is Balanced Governance?

FEDERATED GOVERNANCE IN its purest form means there is no centralized authority and no centralized common concerns. Every decision may be negotiated.

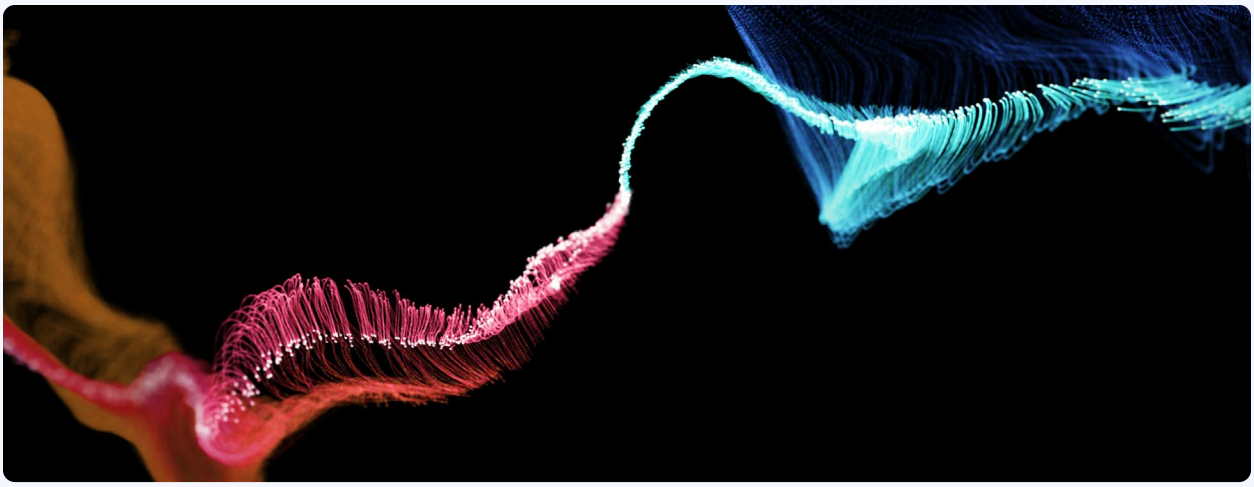
Our customers have found this pure vision to be a noble goal but impractical in practice because there are always practical, common, global concerns, such as security, privacy, regulatory concerns, and many more. We've found they trend toward something I call "Balanced Governance," which seeks to strike the best possible balance between centralized concerns and the value of decentralized velocity. In a Balanced Governance approach, the central governance authority—whether enterprise architecture, a CoE, or other—follows a servant leadership principle to encourage individual teams to guide *them* to "centralize what we should."



In this way, mandates do not come from "them"; when done correctly, all teams feel as though their concerns and ideas are properly represented in the centralized authority's decisions so that architectural mandates are expressed unironically as "we should do it this way."

This is not to say that *all* choices are decided democratically—businesses are not democratic institutions—some requirements force a decision to be made. There should, however, be the sense that decisions have been made in a way that respects differing perspectives, requirements, and which empowers voices to be heard and respected. If they are not heard, teams will go off and produce shadow IT.

It is our recommendation to seek Balanced Governance as a practical middle ground, which allows some concerns to be centrally mandated while still being flexible enough to enable federating governance of design decisions wherever possible.



Why Balanced Governance?

In my experience, I have found the advice to use federated governance to be impractical in most situations. In general, it requires an extremely open culture and one in which teams are empowered to reach across boundaries to effect changes they need across team boundaries. This kind of environment requires extreme levels of discipline, testing automation, and developer independence. Additionally, over time, federated approaches lead to cross-domain misalignments, siloing of data, and duplication of effort.

In a completely federated approach, individual teams are free to make design decisions except for the barest minimum of requirements. This approach grants the greatest possible freedom for individual teams to pursue their local requirements as quickly and as efficiently as possible. What my customers have observed is that these groups tend to be locally optimal, and over time become less well suited for goals which require global optimizations—such as interoperability. In essence, interoperability is a burden (or cost) that must be borne by development teams that they will avoid if they are allowed to. When it comes time to leverage the value of interoperability, the lack of alignment to a common standard and approach makes it cost-prohibitive to reap the benefits of cross-domain data integration.

On the other end of the spectrum—in a completely centralized approach—we see inefficiencies due to drag on velocity by the autocratic centralized authority. Each team is forced to submit their designs and must wait for the central planning authority to review and approve their approach before they may continue or even begin implementation. This approach is excellent for ensuring global goals and standards are considered and maintained. However, as the scope of the challenge under control of the central authority scales, the complexity quickly outpaces the central authority's ability to keep up with the pace of requests and changes. This approach does not scale well as it misses out on the ability for individual teams to optimize locally.

I have found that all but the most progressive (read: digital native) organizations are uncomfortable with the level of freedom, openness, and autonomy required of individual developers to make a truly federated approach work. Most organizations are much more cautious about what changes are made to their codebase and by whom those changes are made. Generally speaking, corporations are too risk averse to allow a fully federated approach which is flexible enough to avoid the problems that emerge over time.

Additionally, I have found that many (read: very large) organizations are starting from a highly federated structure. In this situation, the challenge isn't "make your organization more federated." These organizations generally present with a high level of siloing of information and misalignment of approaches which make integration very challenging. In such organizations, it isn't necessary to make the organization *more* federated. Instead, the challenge is how to bring together and align the various federated components in a way which is mutually beneficial, often without a significant funding stream to support the integration effort. In short, organizations want (and expect) their data to integrate, and don't expect it to cost anything.

This is the crux of the challenge that data mesh attempts to rectify—the central planning scalability problem. A decentralized approach does solve the problems of scalability, and initially it does seem to scale well, but the challenge isn't in initial implementation, it is in ongoing maintenance and alignment of the system. There is a natural tendency for individual actors to try to introduce "chaos" into the system—little misalignments which are locally optimal, but which reduce global optimality. Over time, if not kept in check, these little misalignments add up unless there is continuous oversight and course corrections to reestablish global optimality.

The key takeaway is that governance and oversight of any large, complex system requires constant care, feeding, and course corrections to keep the system running smoothly and on track for global concerns like ease of data integration.

Grow a Center of Excellence

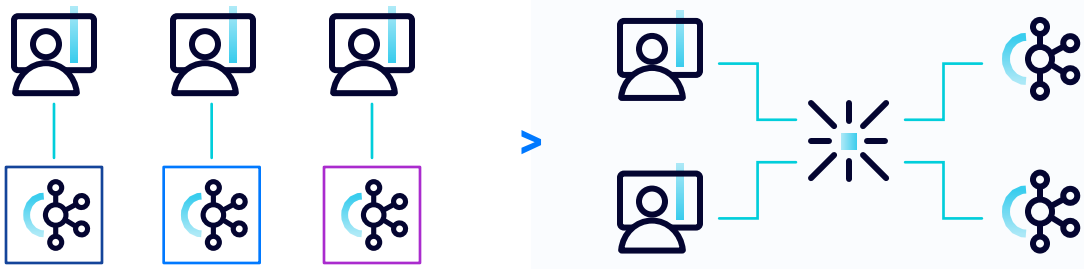
Center of Excellence

DATA MESH IS a complex concept that requires broad changes, and from which the benefits are not readily apparent. Even if there is solid enterprise will from senior leadership to carry through to implementation, there is a significant number of teams and stakeholders who need to be aligned to a new paradigm and who need to be engaged as partners in a new endeavor.

In our experience, the most successful technology transitions include a Center of Excellence based on the principles of servant leadership to manage the social aspects of the change management process. When done correctly, a Data Streaming Center of Excellence serves to build a sense of shared ownership, elevates collaborative problem-solving, engages community involvement, and draws best practices from all parts of the organization. The result is greater speed, governance, and security standards, and expanded value delivery for the business.

Following the principles of servant leadership helps to create a free flow of best practices and ideas between various teams and the central authority, so there is an effective balance between central controls and federated autonomy.

Value of a COE



SPEED

Slow: Each use case starts potentially from scratch

Limited enablement: Developers rely on individual experience and expertise; unscaled knowledge

Faster release cycles: With example code, guidance and a stable platform, app teams build apps faster

Developer empowerment: Training, enablement and standards provide guardrails and drive efficiency

GOVERNANCE

Unreliable, unscalable: With no standards, each app could create strain on platform or issues across "ities"

Improved scalability & reliability: Standards ensure data is available, trustworthy and SLAs are achievable.

VALUE

Expensive: Lack of central oversight to manage, improve FTE costs, infrastructure spend, and mitigate business risk

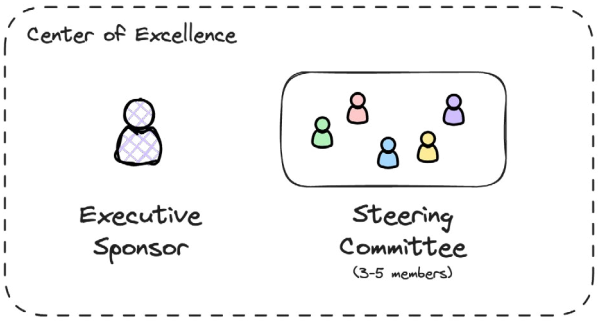
Expanded value: Ability to prioritize resources on highest value efforts, drive cost efficiencies, align to company strategy

Think About Nontechnical Change Management

The most successful transformations we've seen think as strategically about the organizational structure and political alignment as they do about technical strategy. It is just as important to think about the people, process, and financial impact of these changes as they are sometimes the hardest dimensions to change. These dimensions combine into the Organizational Change Management (OCM) approaches and levers that any organization will need to think about to truly effect change that drives the most value.

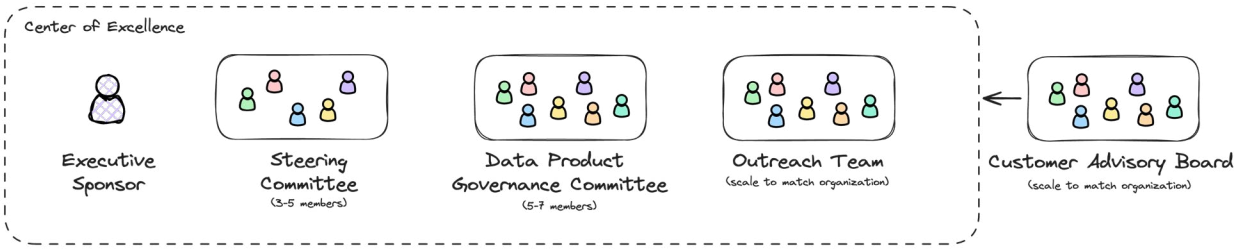
Start Small, Think Big

We have found the most efficacious way to implement the organizational shift from either federated or centralized governance to a balanced approach is by leveraging a Center of Excellence for Data Streaming (CoE) or what we call a Data Streaming Organization (DSO) to act as the focus point for the activities related to data mesh. It is tempting to try to start with a fully formed and full-scale CoE organization on Day 1, but this is actually counterproductive.



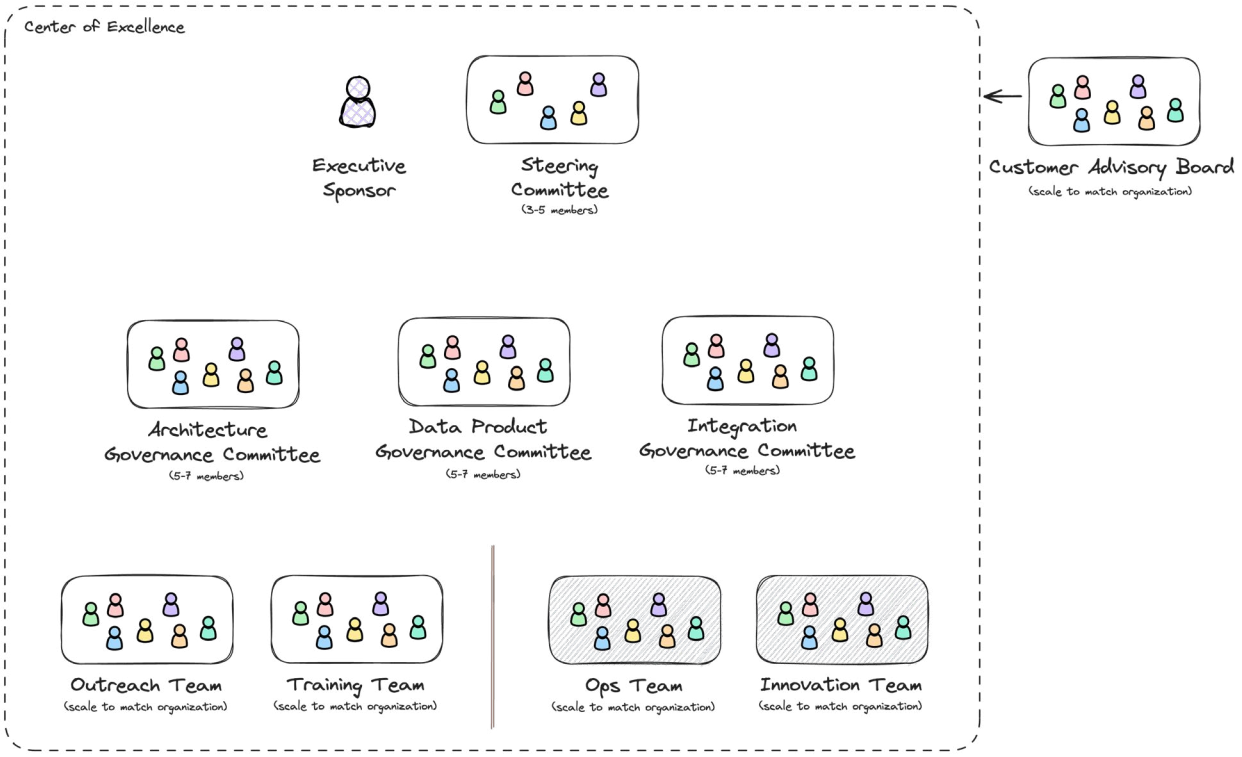
The approach we have found to work best is to start with a small core group of enthusiastic contributors who are willing to go the extra mile for the CoE organization and who are personally invested in the success of the data mesh initiative. The team should be formed from people with enough experience, organizational clout, and with enough organizational breadth so that there is good representation in the CoE.

This approach works best because the CoE Steering Committee itself is a team and needs to progress through the Forming, Storming, and Norming before it can get to the Performing Phase of group development. It is important for the team of individual contributors to work together to build mutual trust and to learn to be effective as a team. Allowing the Steering Committee to be dominated by a single party is an anti-pattern; this will tend to breed mistrust and lead to an ego-driven organization which is seen as not serving the organization.



As the needs of the broader organization arise, we add support for that to the CoE. In this way we incrementally build the CoE in an agile fashion. We present one example of a full CoE model you can work toward—shown below, but don't get hung up re-creating this model exactly.

The point of the exercise is to create just enough CoE that works perfectly for your situation. For example, some organizations choose to give their CoEs a budget for innovation and operations of the data mesh; others choose to keep the CoE entirely "on the side" and advisory for the organization. Either approach can work, and should be chosen according to the culture and budgetary realities at your organization.



The CoE or DSO should be set up to support the appropriate representative model that an organization will put in place. The model should mirror how you want the activities and capabilities to be governed. What capabilities, processes, and decision authorities you want centralized and what would be managed from various business units in a federated approach.

Forming a CoE does not always mean hiring new headcount; these are roles required that may be filled by existing headcount at partial capacity until the workload volume triggers the need for incremental headcount.

Step by Step

Here are the key steps in implementing a Data Streaming Center of Excellence. For an in-depth discussion of each step along with examples of use case templates, runbooks, documentation, team structures and workstreams, I encourage you to watch this presentation.

1. Establish core leadership with the authority and a mandate to set the direction for the CoE.
2. Gather requirements, objectives, and constraints from multiple business and IT functions, prioritizing them into a strategic plan for success of the CoE and the organization.
3. Drive advocacy and outreach across the enterprise. Enable developer training and upskilling as needed.
4. Enable key stakeholders: architects, developers, analysts, data producers and consumers.
5. Identify and execute early proofs of value (PoV) to build buy-in and adoption of the approach.
6. Improve and/or establish governance which fulfills requirements yet is sensitive to the performance goals of the organization.
7. Provide consultative and advisory services to teams to ensure smooth adoption, and excellent execution.
8. Develop reusable patterns, common tools, and use case templates to streamline execution and simplify adoption.
9. Mature and productize offerings of the CoE to reap efficiencies of scale.
10. Commoditize the most common offerings to maximize throughput and team velocity.

Conclusion

WE'VE LEARNED A great deal about the state of data mesh as a technology, as an approach, and as a movement. In short, data mesh is rapidly maturing, but is not yet quite "plug-and-play" ready for organizations that wish to purchase a solution off the shelf, plug it in, and expect a "data mesh" light to come on with an "easy button" implementation. Implementing your data mesh still requires a fair amount of decision-making, solutioning, and problem-solving.

We've learned about the technical and sociological hurdles you'll need to overcome to successfully implement data mesh. The data mesh principles espouse the benefits of federated governance, but we caution that a practical solution should not seek to federate for federation's sake—we urge you to take a balanced approach that answers the question, "What is the most effective solution that federates as much as possible for velocity while still providing coverage for our non-functional requirements?" For some organizations, data mesh will mean some organizational change management will be needed in order to shift toward more distributed governance of your data.

Though implementing data mesh seems like a daunting endeavor, it is achievable with a well-planned and executed strategy. The effort is well worth the payoff; you'll end up with an organization that can apply more of your development staff toward the challenges of data engineering, resulting in higher data feature velocity and accelerated data value capture. The key to a successful strategy is to start small, demonstrate value early, and to rapidly iterate toward your North Star with frequent course corrections along the way.

With the current maturity of the data mesh movement, setting your strategy is an exercise that needs to consider the people, process, and technology aspects to create a holistic solution, not just a technology one. You're going to need to upskill your staff and rebalance ownership, update your processes for new concepts of governance, and rewire your solutions to enable new techniques for data sharing. It is not just simply an exercise in setting your target and rewiring your applications or databases.

Fortunately, Confluent provides all the foundational tools and features you need to begin implementing your data mesh today. You can quickly wire up data across more than 120 technologies (sources and sinks) using Confluent Connectors, process your data with Stream Processing (Flink, ksqlDB, and KStreams), and ensure the data is governed strategically with Stream Governance. Data Portal enables data product discoverability, reuse, and collaboration. And enterprise-grade security includes role-based access control (RBAC) as well as bring-your-own-key (BYOK) encryption to provide a greater degree of privacy and data integrity.

Confluent Data Streaming Platform will set your data mesh apart from those built on other data-at-rest stacks. Data meshes built on the Confluent Data Streaming Platform have several distinct advantages; they provide a common physical layer for all nodes of the data mesh, which simplifies the transfer of data via Confluent, reduces complexity, and allows consumers to be immediately updated when the data product has new data available. Additionally, the data that is made available is immutable and replayable—both excellent qualities that enable data streams to provide a system of record functionality for legacy applications that weren't designed for historical views of their data.

Confluent Professional Services has worked with numerous customers to develop their data mesh strategy and identify a North Star and vision for successful implementation. Whether your challenges are organizational, technological, or a mix of the two, we provide you with a strategy that sets you up for success. We help you get started and provide ongoing support and expert advice on best actions at every step, helping you adjust your strategy to the realities of your business and technology. It is increasingly likely that data mesh implementations need to service more complex hybrid and multicloud scenarios to support multiple business units and lines of business. In these more complex situations, your implementation will be set off on the right foot by the expert Solution Architects from Confluent Professional Services.

Take the Next Step

» Build a data mesh using the fully managed, cloud-native, complete Confluent Data Streaming Platform. Get started for free at confluent.io/get-started.

» For further reading on building decentralized data architectures with event streams, see Adam Bellemare's Practical Data Mesh.

» Watch a webinar with a Forrester Analyst and Confluent customer on Data Mesh from Concept to Implementation.

» Learn more about building data products by visiting the Solution Hub.

» Explore use cases and resources for Streaming Data Pipelines and Streaming Applications.

» Learn more about Building a Data Streaming Center of Excellence.