



# Enterprise Data Mesh

SOLUTIONS, USE CASES & CASE STUDIES

Fall 2021



# Introduction

Data Mesh is an emerging hot topic for enterprise software that puts focus on new ways of thinking about data. Data Mesh aims to improve business outcomes of data-centric solutions, as well as to drive adoption of modern data architectures.

From the business point of view, Data Mesh introduces new ideas around **'data product thinking'** and how it can help to drive a more cross-functional approach to business domain modeling and creating high-value data products.

From the technology side, there are three important and new focus areas for data-driven architecture:

1. distributed, **decentralized data architecture** that help organizations move away from monolithic architectures
2. **event-driven data ledgers** for enterprise data in motion
3. **streaming-centric pipelines** to replace legacy batch type tooling, handle real-time events, and provide more timely analytics

Oracle's focus on the Data Mesh has been in providing a platform that can address these emerging technology requirements, including tools for data products, decentralized event-driven architectures, and streaming patterns for data in motion.

Investing in a Data Mesh can yield impressive benefits, including:

- **total clarity into data's value chain**, through applied 'data product thinking' best practices
- **>99.999% operational data availability**<sup>1</sup>, using microservices based data pipelines for data consolidation and data migrations
- **10x faster innovation cycles**<sup>2</sup>, shifting away from ETL, to continuous transformation and loading (CTL)
- **~70% reduction in data engineering**<sup>3</sup>, gains in CI/CD, no-code and self-serve data pipeline tooling, and agile development

Read on for a look at some impressive case studies and positive results from early adopters in this approach.

# Beware the hype...

Since Data Mesh is a rising hot topic and still in the early days of maturity, there may be some marketing content that uses the words “data mesh” but the described solutions do not actually fit the core approach.

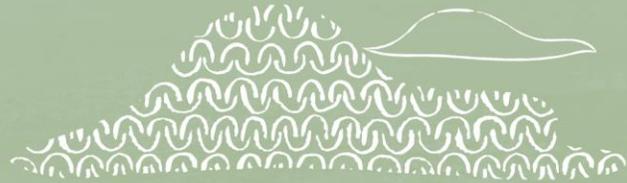
A proper Data Mesh is a mindset, an organizational model and an enterprise data architecture approach...it should have some mix of data product thinking, decentralized data architecture, event-driven actions and a streaming centric ‘service mesh’ style of microservices design.

A Data Mesh is not a...

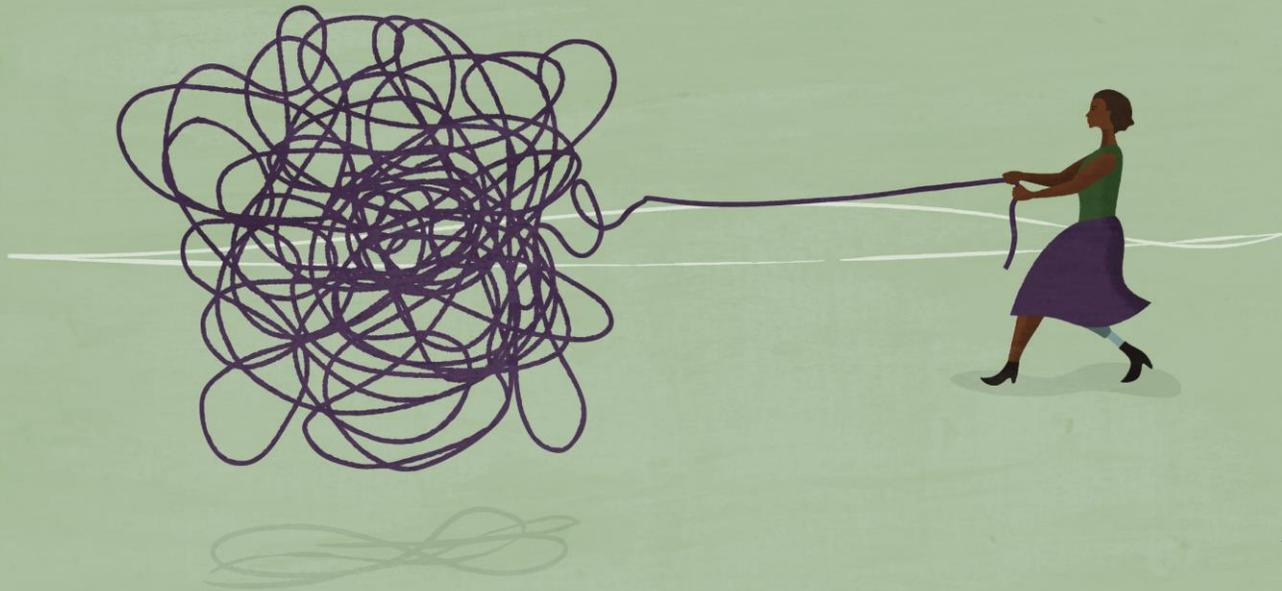
- **Single Cloud Data Lake** ...even with ‘domains,’ catalogs & SQL access
- **Data Catalog / Graph** ...a data mesh needs a physical implementation
- **Point-Product** ...no vendor has a singular product for Data Mesh
- **IT Consulting Project** ...strategy/tactics still require platforms and tools
- **Data Fabric** ...which is broadly inclusive of monolithic data architectures
- **Self-Service Analytics** ...easy-to-use UX can front a mesh or a monolith

As the popularity of Data Mesh continues to increase, there will be many bandwagon vendors/consultants, so it’s important to *beware of the hype!*





*Data mess!*



# Why data mesh?

Because the old ways are not working well. Most business transformation initiatives fail. Most of the time and costs for digital platforms are sunk into ‘integration’ efforts<sup>4</sup>. Monolithic tech architectures of the past are cumbersome, expensive, and inflexible. Additionally:

- 70-80% of digital transformations fail<sup>5</sup>
- Rise of distributed architectures, app, data and cloud architectures are all becoming less centralized/monolithic
- Cloud lock-in is real<sup>6</sup>, and can become more costly<sup>7</sup>
- Data Lakes rarely succeed<sup>8</sup>, and are only analytics focused
- Organizational silos exacerbate data sharing issues<sup>9</sup>
- Everything is speeding up, pace of innovation, the speed of IT events, your competition are all moving faster than ever
- Cost of operational data outage is rising<sup>10</sup>

Data Mesh is no silver bullet, but the principles, practices and technologies have been aligned to focus on solving some of the most pressing, unaddressed modernization objectives for data-driven business initiatives.

Note: an excellent primer to why Data Mesh is needed is Zhamak Dehghani's 2019 paper, "...from Monoliths to Data Mesh"<sup>9</sup>

# New concept for data

## Data Mesh approach:

1. Emphasizes cultural change, as a **mindset shift towards thinking of data 'as a product'** – which in turn can prompt organizational and process changes to manage data as a tangible, real capital asset of the business.
2. Calls for **alignment across operational and analytic data domains**. A Data Mesh aims to link data producers directly to data consumers and **remove the IT middleman** from the processes that ingest, prepare and transform data resources.
3. Technology platform built for **'data in motion' is a key indicator of success** – a two-sided platform that links enterprise data producers and consumers. Data Mesh core is a **distributed architecture for on-prem and multi-cloud data**.



# Defining the data mesh

## 1.) Outcomes focused

**Data product thinking** – mindset shift to data consumer point of view

- data domain owners responsible for KPIs/SLAs of data products

**Alignment for Ops & Analytics** – no more ‘throwing data over the wall’

- same technology mesh and data domain semantics for all

**Data in motion** – as a core competency for producing data products

- Remove the ‘man in the middle’ – by making data events directly accessible from systems of record and providing self-service real-time data pipelines to get the data where needed

## 2.) Rejects monolithic IT architecture

**Decentralized architecture**

- an architecture built for decentralized data, services and clouds

**Event-driven data ledgers**

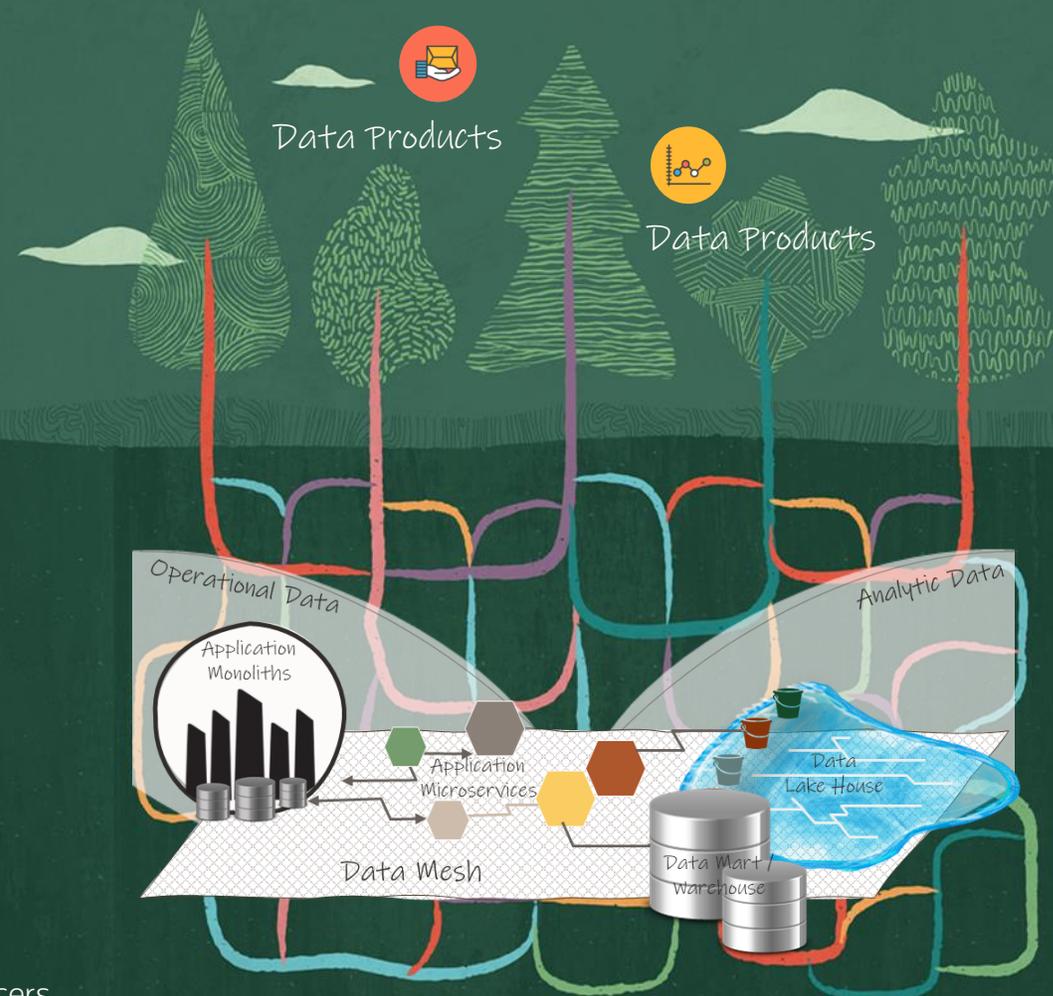
- designed to handle events of all types, formats and complexity

**Streaming-centric data pipelines**

- stream processing by default, centralized batch processing by exception

**Self-service, governed platform**

- built to empower developers and directly connect data consumers to data producers
- security, validation, provenance and explainability built-in



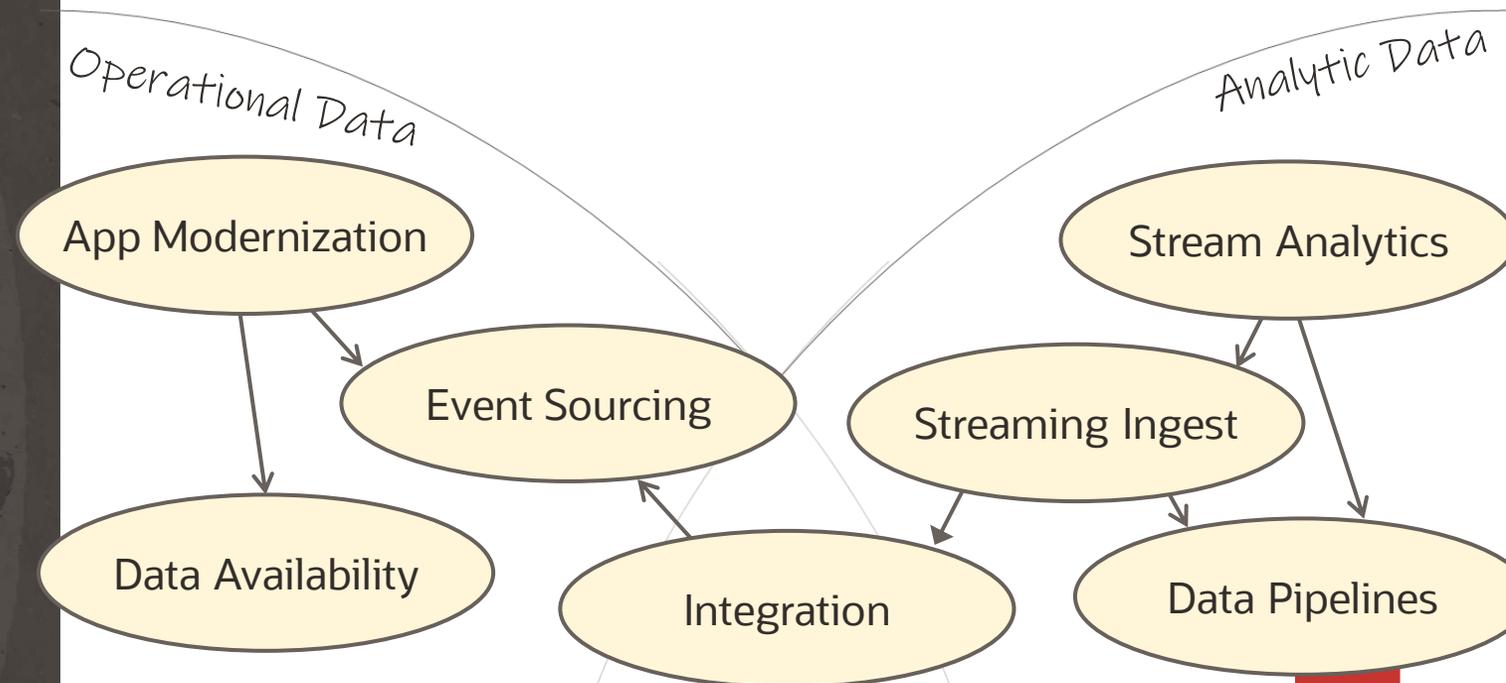
## Chapter 1

# Seven Data Mesh Use Case Examples

A successful Data Mesh fulfills use cases for Operational as well as Analytic Data domains.

The following seven use cases illustrate the breadth of capabilities that a Data Mesh brings to enterprise data.

*“By integrating real-time operational data and analytics, companies can make better operational and strategic decisions.”<sup>11</sup>*



# Data Mesh Use Case

## APPLICATION MODERNIZATION

Looking beyond 'lift and shift' migrations of monoliths to the cloud, many organizations also seek to retire their monolithic applications of the past and move towards a more modern microservices application architecture for the future.

But legacy app monoliths typically depend on big monolithic databases, raising the question, "*how to phase the migration plan to decrease disruption, risks, and costs?*"

A Data Mesh can provide an important operational IT capability for customers doing phased transitions from monoliths to mesh architecture. For example:

- Sub-domain offloading of DB transactions
  - Eg; filtering data by 'bounded context'
- Bi-directional transaction replication for phased migrations
- Cross-platform sync (eg; mainframe to DBaaS)

In the lingo of microservices architects, this approach is using a bi-directional **Transaction Outbox**<sup>12</sup> to enable the **Strangler Fig**<sup>13</sup> migration pattern, one **Bounded Context**<sup>14</sup> at a time.

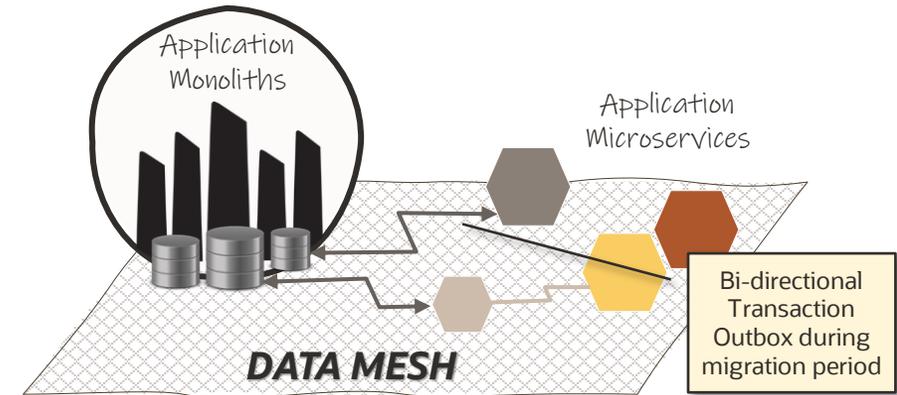


Figure 1: data mesh foundation for monolith migrations

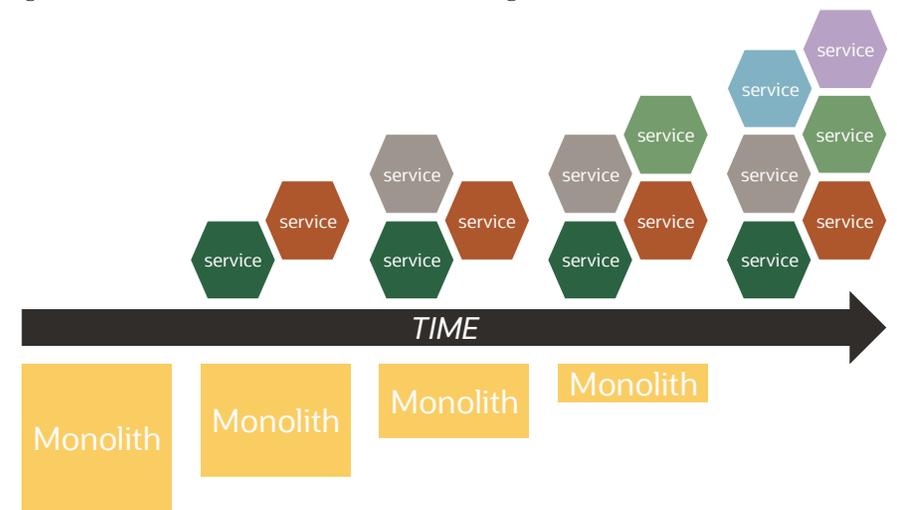


Figure 2: strangler fig pattern for monolith decomposition and phased migrations

# Data Mesh Use Case

## DATA AVAILABILITY & CONTINUITY

Business-critical applications require very high KPIs and SLAs around resiliency and continuity. Regardless of whether these applications are monolithic, microservices or something in between, *they can't go down!*

For mission-critical systems a distributed eventual-consistency data model is usually not acceptable. However, these apps must operate across many data centers. This begs the question, “*how can I run my Apps across more than one data center while still guaranteeing correct and consistent data?*”

A Data Mesh can provide the foundation for decentralized yet 100% correct data across sites. For example:

- Very low latency logical transactions (cross-platform)
- ACID capable guarantees for correct data
- Multi-active, bi-directional and conflict resolution

Regardless of whether the monoliths are using ‘sharded data sets’ or the microservices are being setup for cross-site HA, the Data Mesh can provide high speed, correct data at any distance.

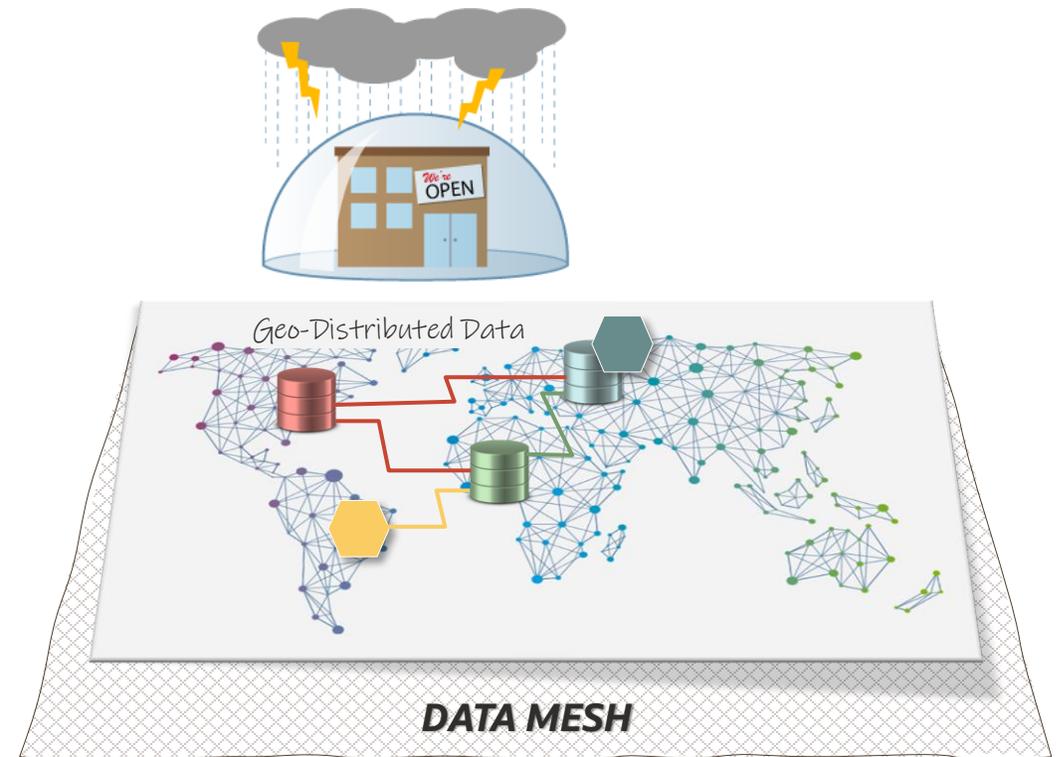


Figure 1: data mesh for geographically distributed data events

# EVENT SOURCING & TRANSACTION OUTBOX Use Case

A modern 'service mesh' style platform uses events for data interchange. Rather than depending on batch processing in the data tier, data payloads flow continuously when events happen in the application or data store.

For some architectures, microservices need to exchange data payloads with each other. Other patterns require interchange between monolithic applications or data stores. This begs the question, "*how can I reliably exchange microservice data payloads among my apps and data stores?*"

A Data Mesh can supply the foundation tech for microservices centric data interchange. For example:

- Microservice to Microservice (w/in Context)
- Microservice to Microservice (across Contexts)
- Monolith to/from Microservice

Microservices patterns like Event Sourcing, CQRS, and Transaction Outbox<sup>12</sup> are commonly understood solutions – a Data Mesh provides the tooling and frameworks to make these patterns repeatable and reliable at scale.

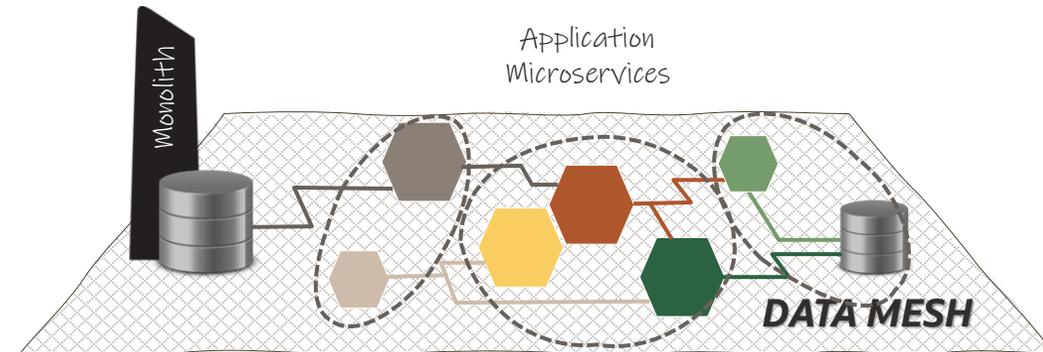


Figure 1: event-based interop among various Apps, Microservices and DBs

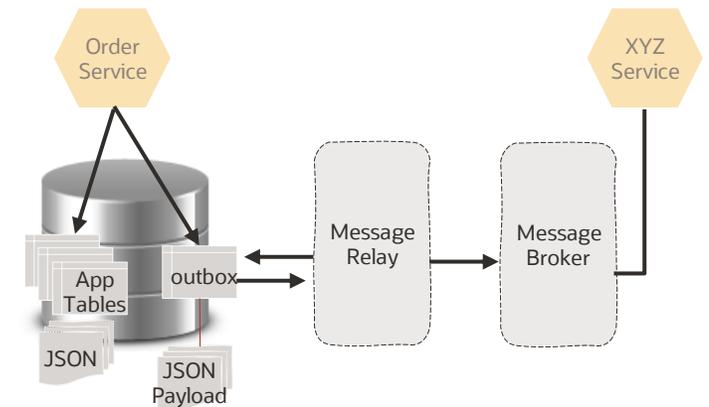


Figure 2: generic pattern for Transaction Outbox  
(note: there are Data Mesh variations/optimizations for this pattern)



# Data Mesh Use Case

## EVENT-DRIVEN INTEGRATION

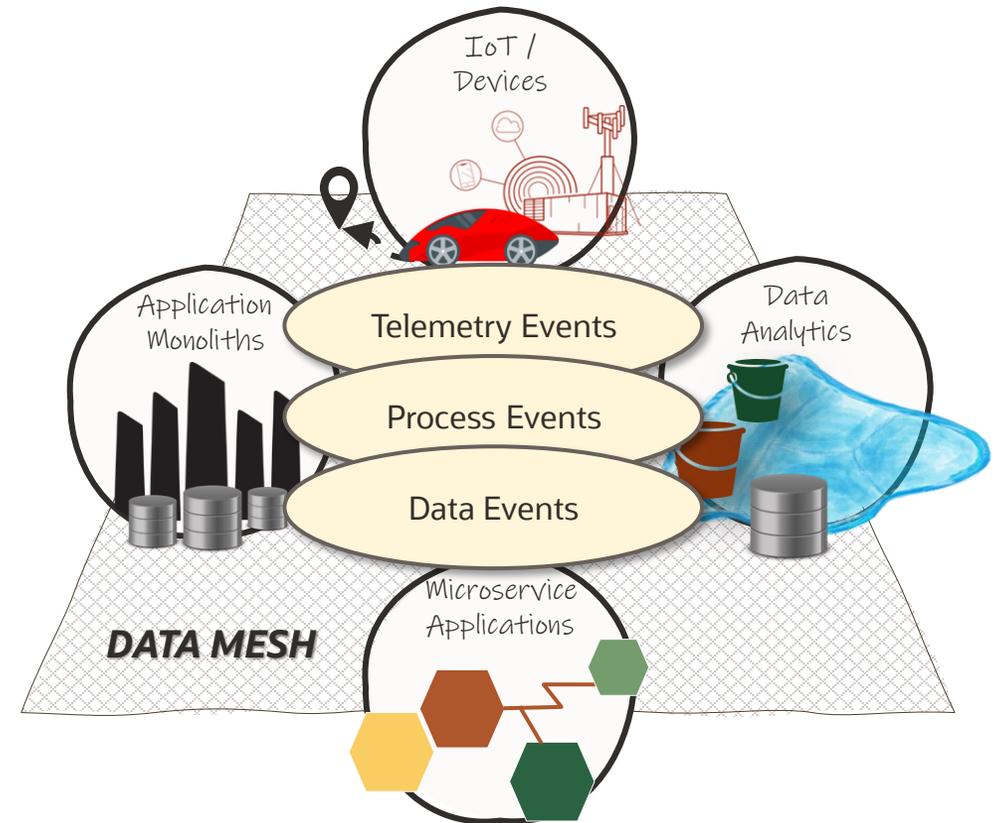
Beyond microservice design patterns, the need for enterprise integration extends to other IT systems such as DBs, business processes, applications and physical devices of all types. A Data Mesh provides the foundation for integrating **data in motion**.

Data in motion is typically event-driven. A user action, a device event, a process step or a data store commit can all initiate an event with a data payload. These data payloads are crucial for integrating IoT systems, business processes and databases, data warehouses and data lakes.

A Data Mesh supplies the foundation tech for real-time integration across the enterprise. For example:

- Connecting real world device events to IT systems
- Integrating business processes across ERP systems
- Aligning operational DBs with analytic data stores

Large organizations will naturally have a mix of old and new systems, monoliths and microservices, operational and analytic data stores – a Data Mesh can help to unify these resources across differing business and data domains.



# STREAMING INGEST (FOR ANALYTICS)

Analytic data stores may include data marts, data warehouses, OLAP cubes, data lakes and data lake house technologies.

Generally speaking, there are only two ways to bring data into these analytic data stores:

1. Batch / Micro-batch loading – on a time scheduler
2. Streaming Ingest – continuously loading data events

A Data Mesh provides the foundation tech for a streaming data ingest capability. For example:

- Data events, from databases, data stores etc.
- Device events, from physical device telemetry
- Application events, logging or business events

Ingesting events by stream often reduces the impact on the source systems, improves the fidelity of the data (important for data science) and can empower a real-time analytics use case where valuable to the data product owners.

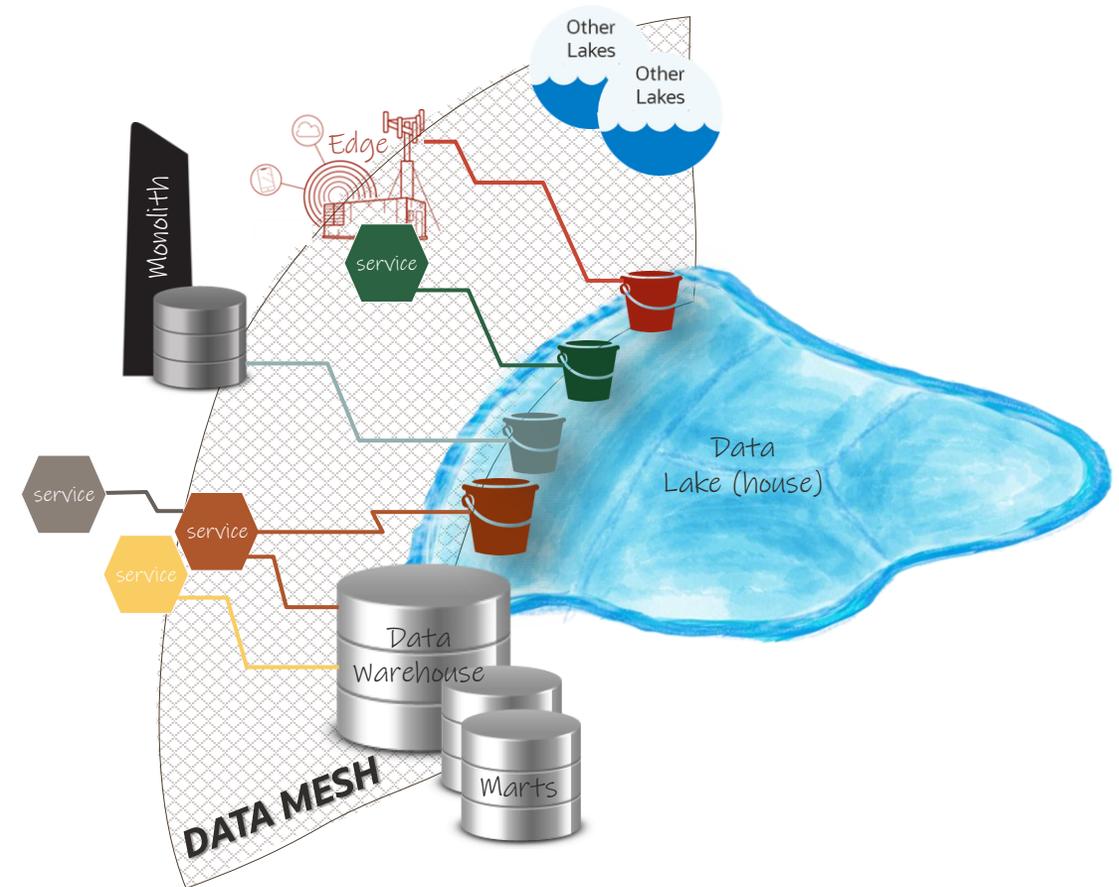


Figure 1: leveraging a Data Mesh for common data ingest across Data Lakes, Data Warehouses, and Data Marts

# STREAMING DATA PIPELINES

Once ingested into the analytic data stores, there is usually a need for 'data pipelines' to prepare and transform the data across different data stages or data zones. This is a process of data refinement often needed for the downstream analytic data products.

A Data Mesh can provide an independently governed data pipeline layer that works with the analytic data stores, providing the following core services:

- Self-service data discovery and data preparation
- Governance of data resources across domains
- Data transformation into required data product formats
  - Eg; streaming ETL
- Data verification, by policy, to assure consistency

These data pipelines should be capable to work across different physical data stores (such as marts, warehouses, lakes etc) or as a "pushdown data stream" within analytic data platforms that support streaming data, such as Apache Spark and other data lake house technologies.

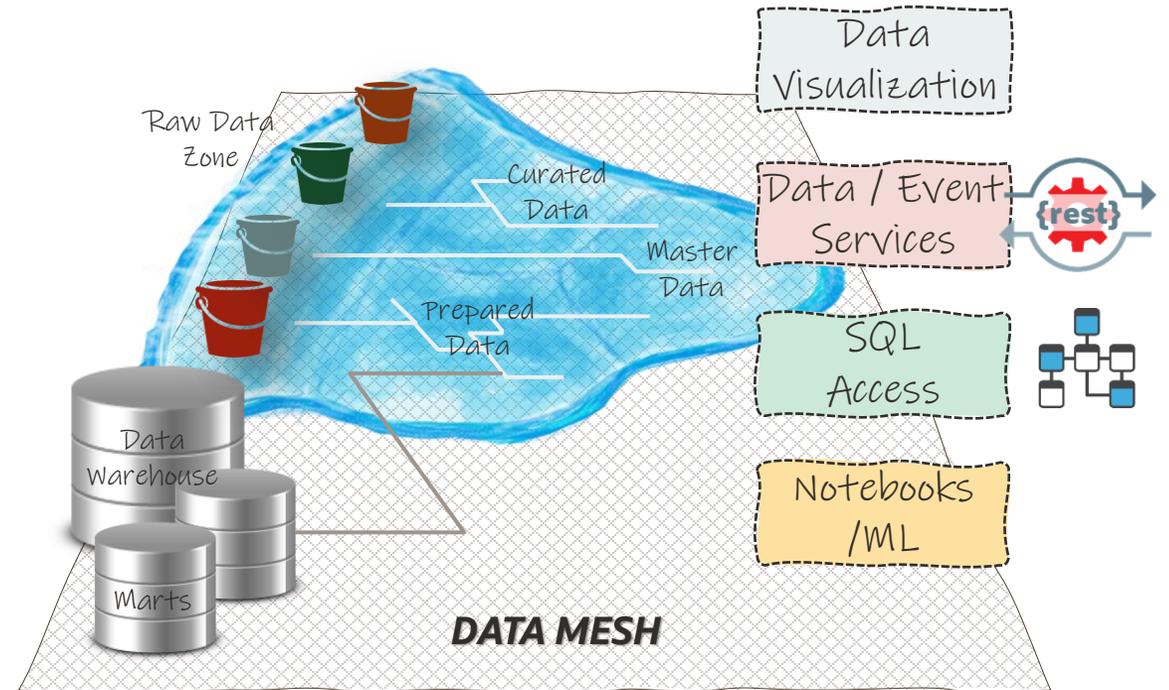


Figure 1: a data mesh can create, execute and govern streaming pipelines within a Data Lake

# STREAMING ANALYTICS

Events are continuously happening. The analysis of events in a stream can be crucial for understanding what is happening from moment to moment.

This kind of time-series based analysis of real-time event streams may be important for real world IoT device data, but also for understanding what is happening in your IT data centers or across financial transactions (eg; fraud monitoring).

A full featured Data Mesh will include the foundation capabilities to analyze events of all kinds, across many different types of event time windows. For example:

- Simple event stream analysis (eg; Web events)
- Business activity monitoring (eg; SOAP/REST events)
- Complex event processing (eg; multi-stream correlation)
- Data event analysis (eg; on DB/ACID transactions)

Like data pipelines, the stream analytics may be capable of running within established data lake house infrastructure, or separately – as native cloud services for example.

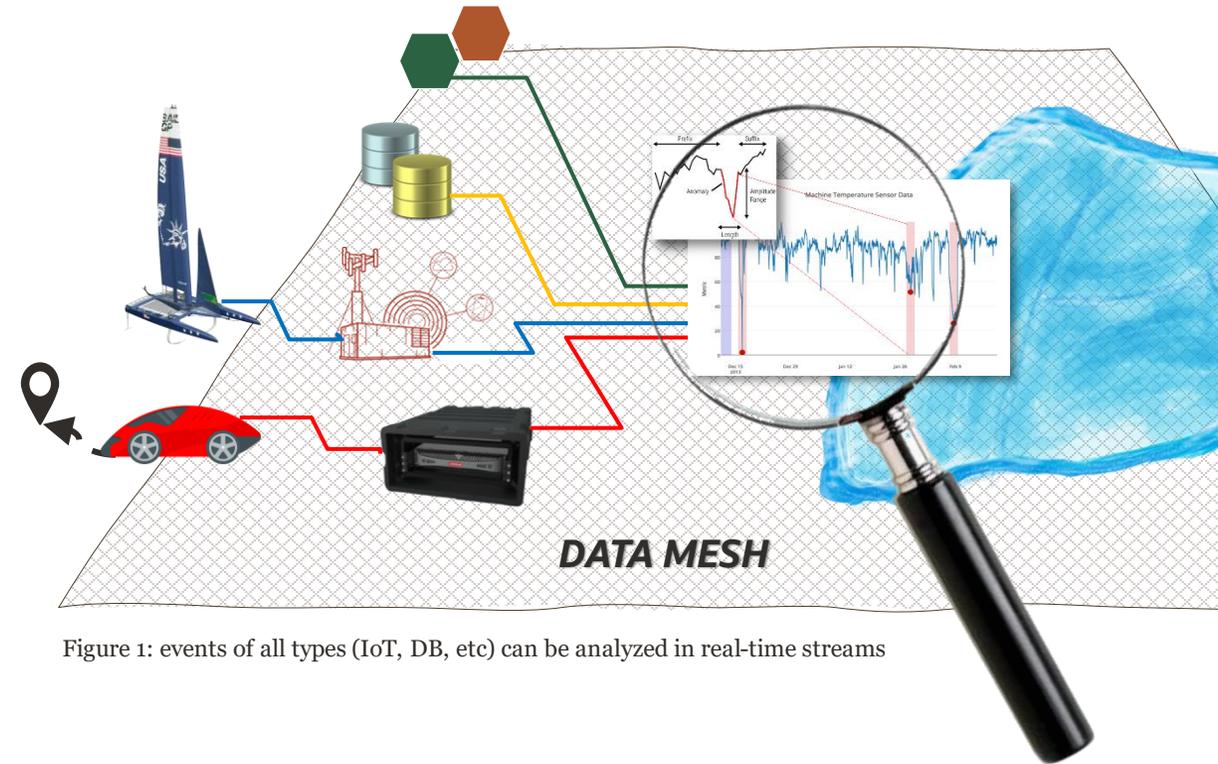
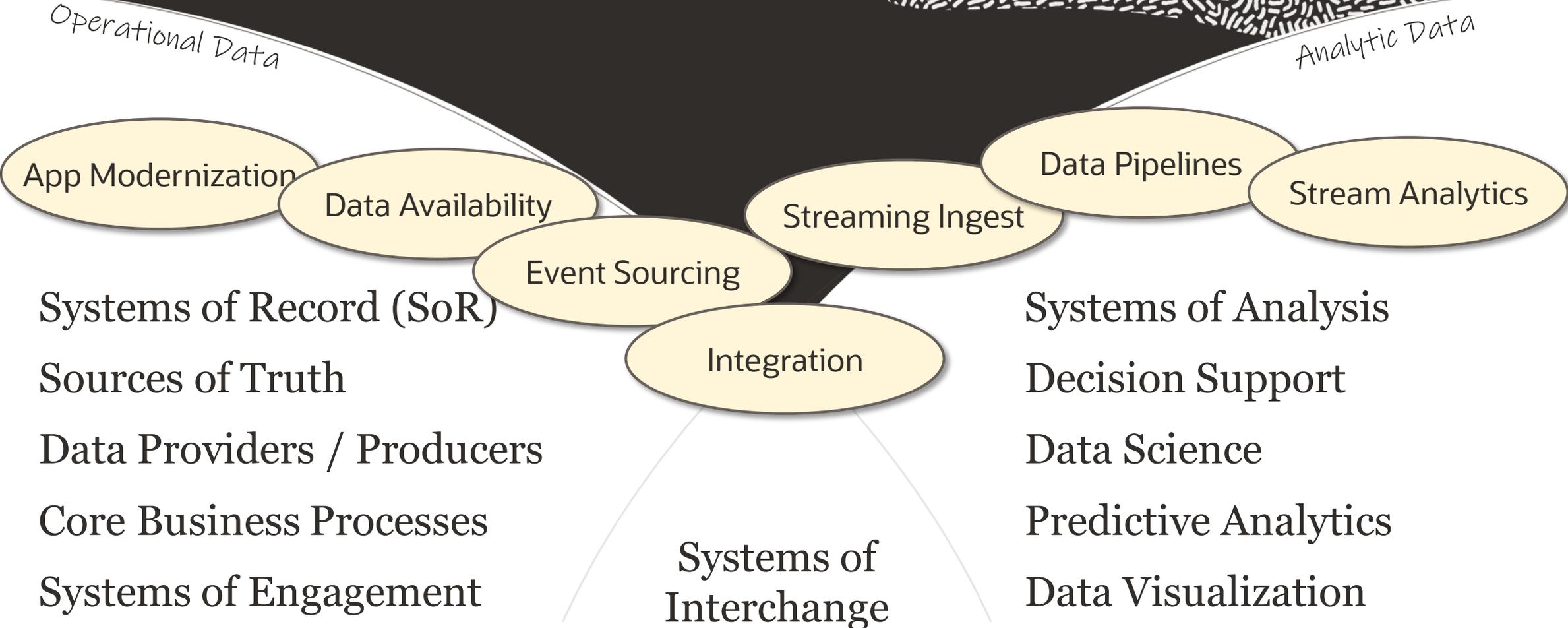


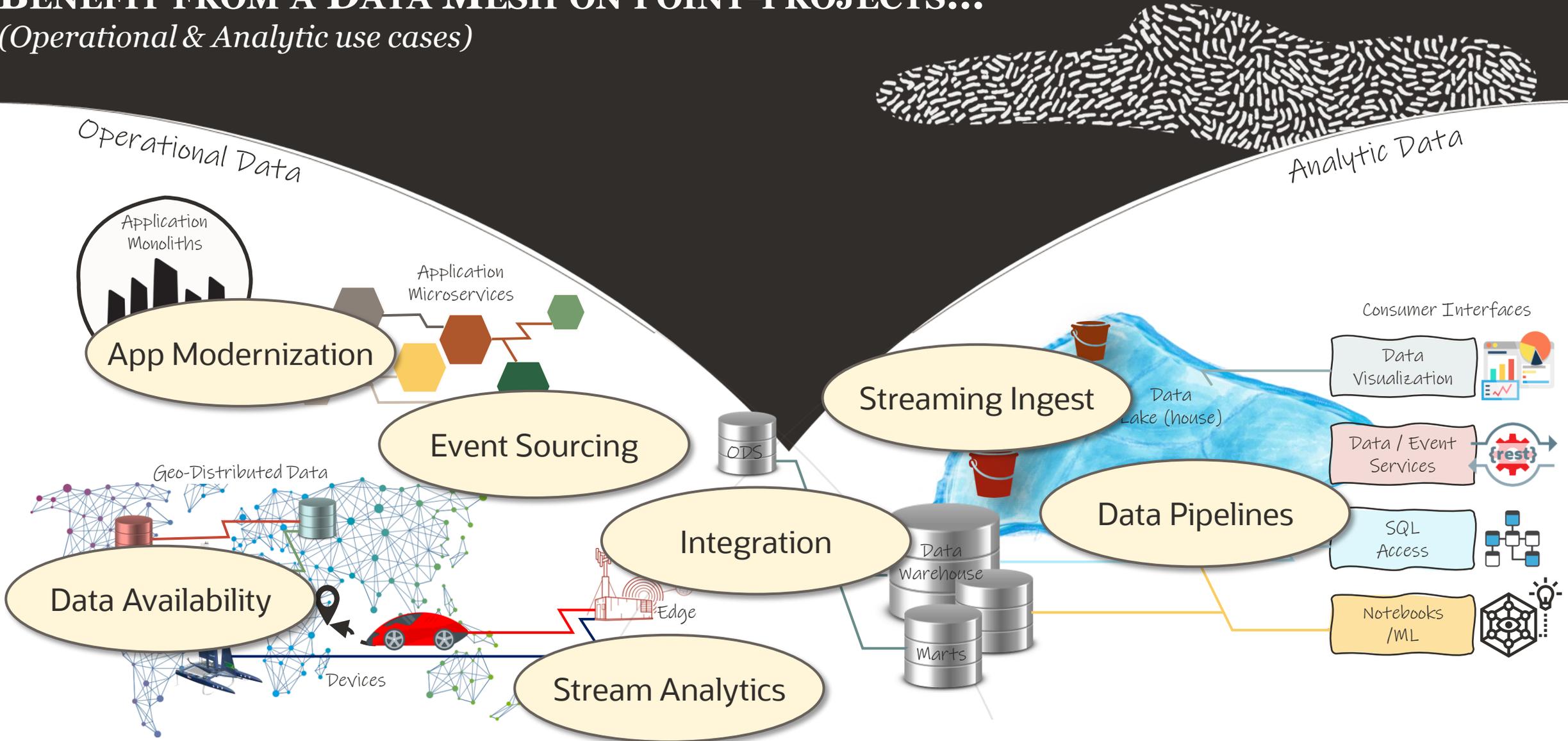
Figure 1: events of all types (IoT, DB, etc) can be analyzed in real-time streams

# DATA MESH USE CASES APPLY TO OPERATIONAL & ANALYTIC SYSTEMS



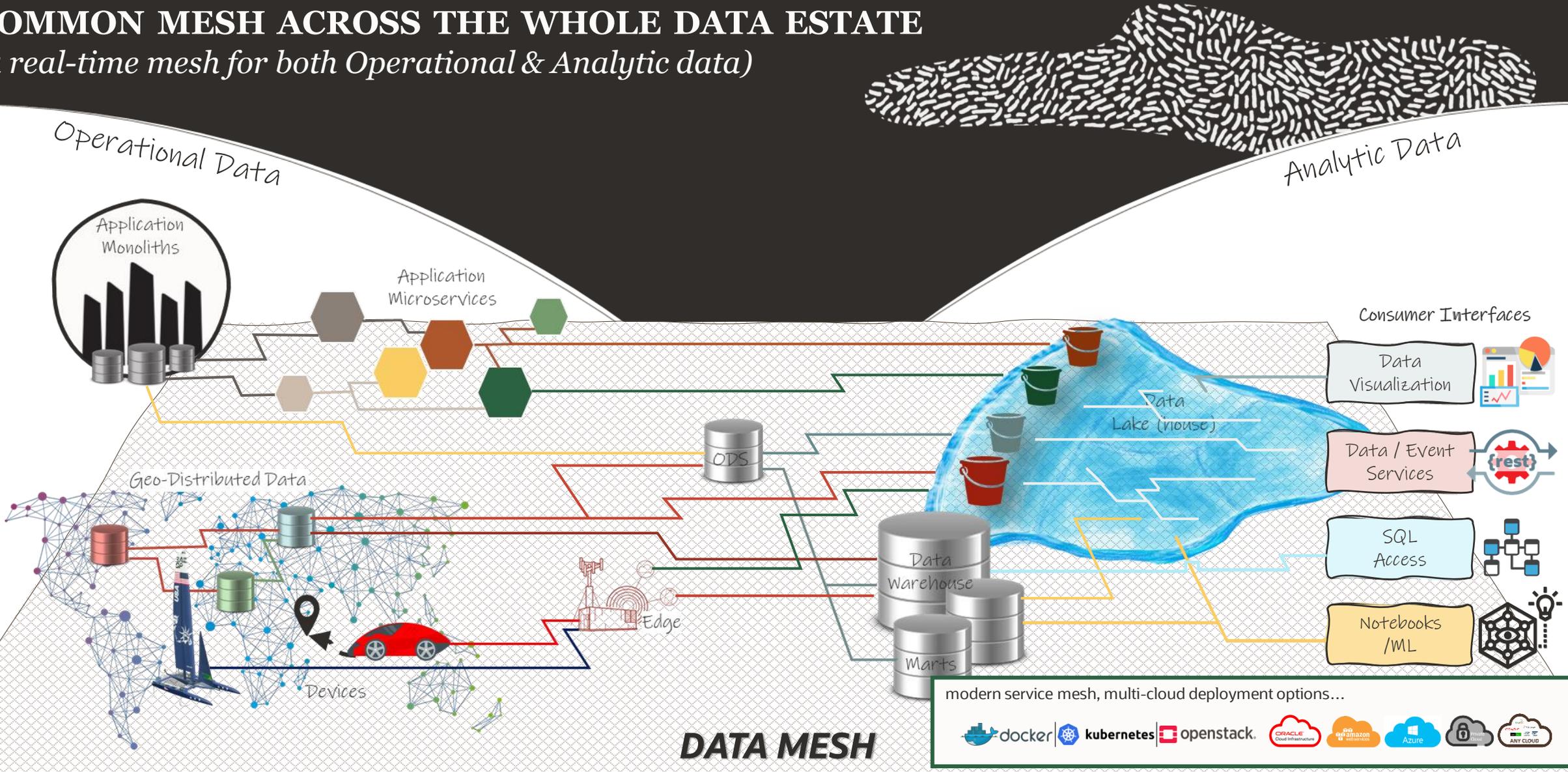
# BENEFIT FROM A DATA MESH ON POINT-PROJECTS...

(Operational & Analytic use cases)



# ...ACHIEVE MAXIMUM VALUE BY OPERATING A COMMON MESH ACROSS THE WHOLE DATA ESTATE

(a real-time mesh for both Operational & Analytic data)



## Chapter 2

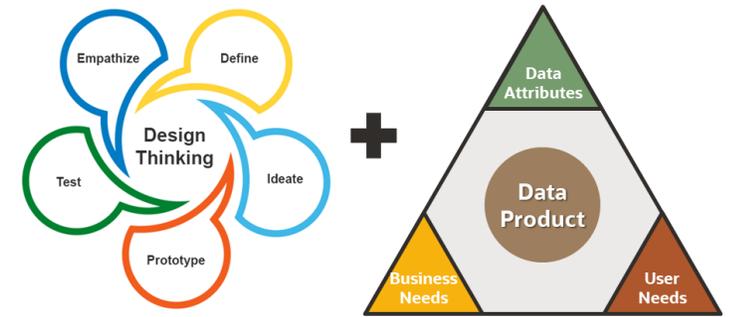
# Four Key Attributes of a Data Mesh

A Data Mesh should not be just a new buzz word on top of an old tech architecture.

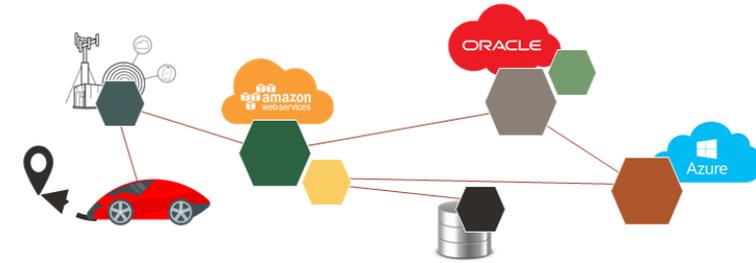
As Data Mesh aims to bring unique value, it must have unique attributes that are distinctly different from commonplace solutions that have already been around for decades.

These are four key attributes to be aware of.

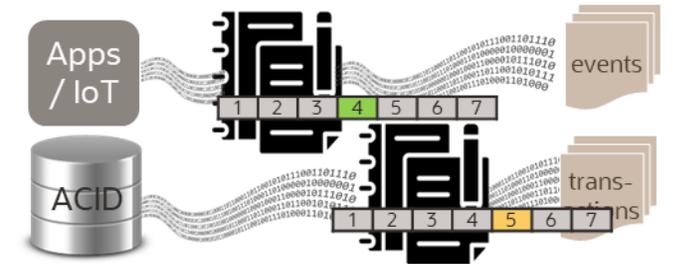
## 1 DATA PRODUCT THINKING



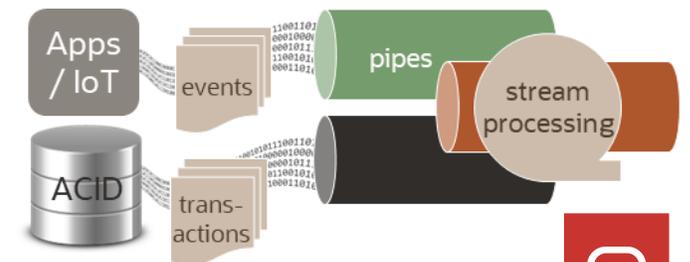
## 2 DECENTRALIZED DATA ARCHITECTURE



## 3 EVENT-DRIVEN DATA LEDGERS

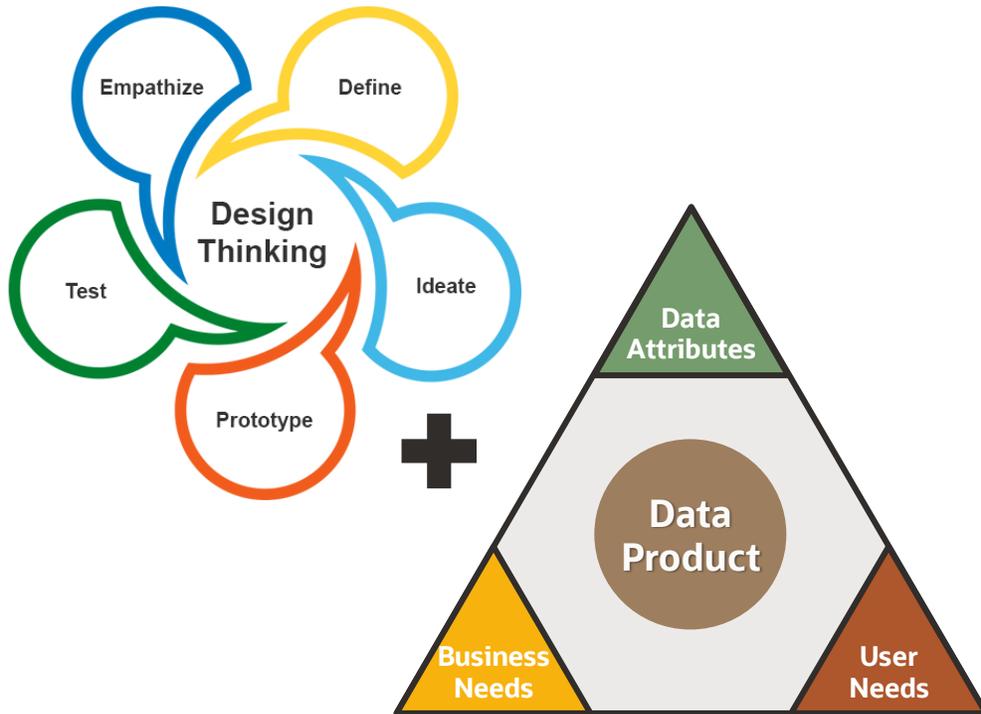


## 4 POLYGLOT DATA STREAMING



# Data Mesh Attribute

## 1.) DATA PRODUCT THINKING



A mindset shift is the most important first step towards a Data Mesh. The willingness to embrace the learned practices of innovation is the springboard towards successful modernization of data architecture.

These learned practice areas include:

- **Design Thinking** – for solving ‘wicked problems’
- **Jobs to be Done Theory** – customer focused innovation, and the Outcome-Driven Innovation process

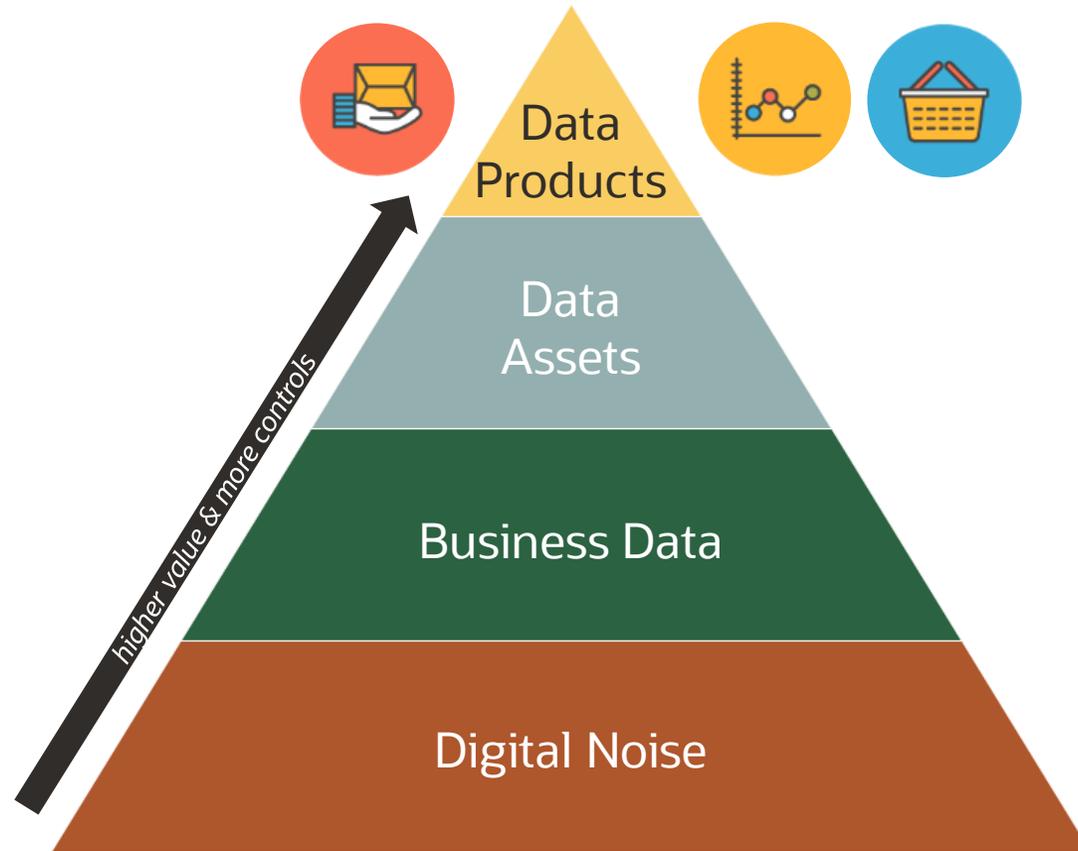
Design Thinking methodologies bring proven techniques that help break down the organizational silos frequently blocking cross-functional innovation. The Jobs to be Done Theory is the critical foundation for designing data products that fulfil specific end-consumer goals, or jobs to be done – it defines the product’s purpose.

The data product approach initially emerged from the data science community but is now going mainstream, being applied to all aspects of the data management discipline. It keeps the focus on the business outcomes, the data consumers... rather than the IT tech.

Data product thinking can be applied to other data architectures, but it is an essential part of a data mesh.

# Data Mesh Attribute

## 1a.) DATA PRODUCTS



Products of any kind, from raw commodities to items at your local store are produced as assets of value, intended to be consumed and with a specific 'job to be done.'

Data products can take a variety of forms, depending on the business domain or problem to be solved, and may include:

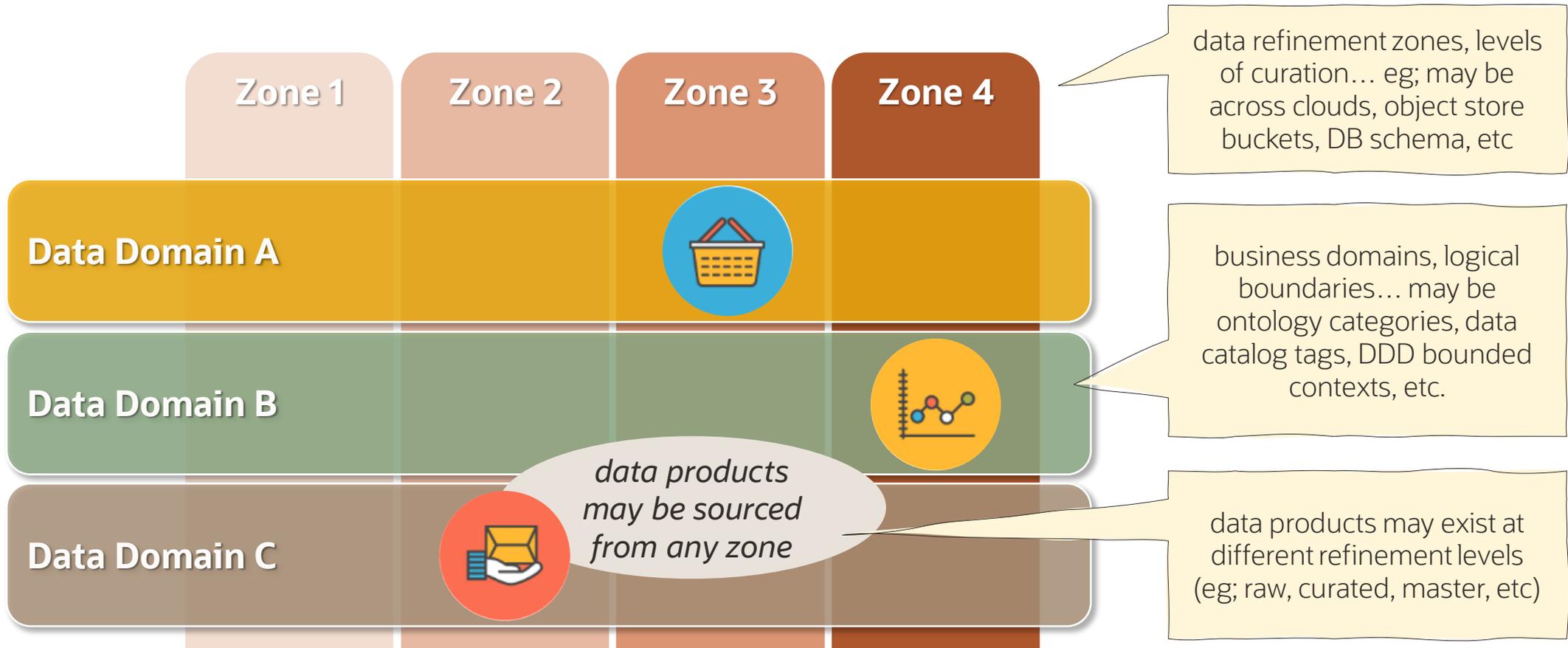
- **Analytics** – historic/real-time reports & dashboards
- **Data Sets** – data collections in different shapes/formats
- **Models** – domain objects, data models, ML features
- **Algorithms** – ML models, scoring, business rules
- **Data Services & APIs** – docs, payloads, topics, REST APIs...

A data product is created for consumers, requiring tracking of additional attributes such as:

- **Stakeholder Map** – who creates and consumes this product?
- **Packaging, Documentation** – how is it consumed?
- **Purpose & Value** – implicit/explicit value? depreciation?
- **Quality, Consistency** – KPIs and SLAs of usage?
- **Provenance, Lifecycle & Governance** – trust & explainability?

# 1b.) CROSS-FUNCTIONAL DATA DOMAINS

The 'wicked problem' is often in aligning different cross-functional teams to common data domains – domains that require shared data sets, data models, business policies and business rules.



# Data Mesh Attribute

## 2.) DECENTRALIZED DATA ARCHITECTURE

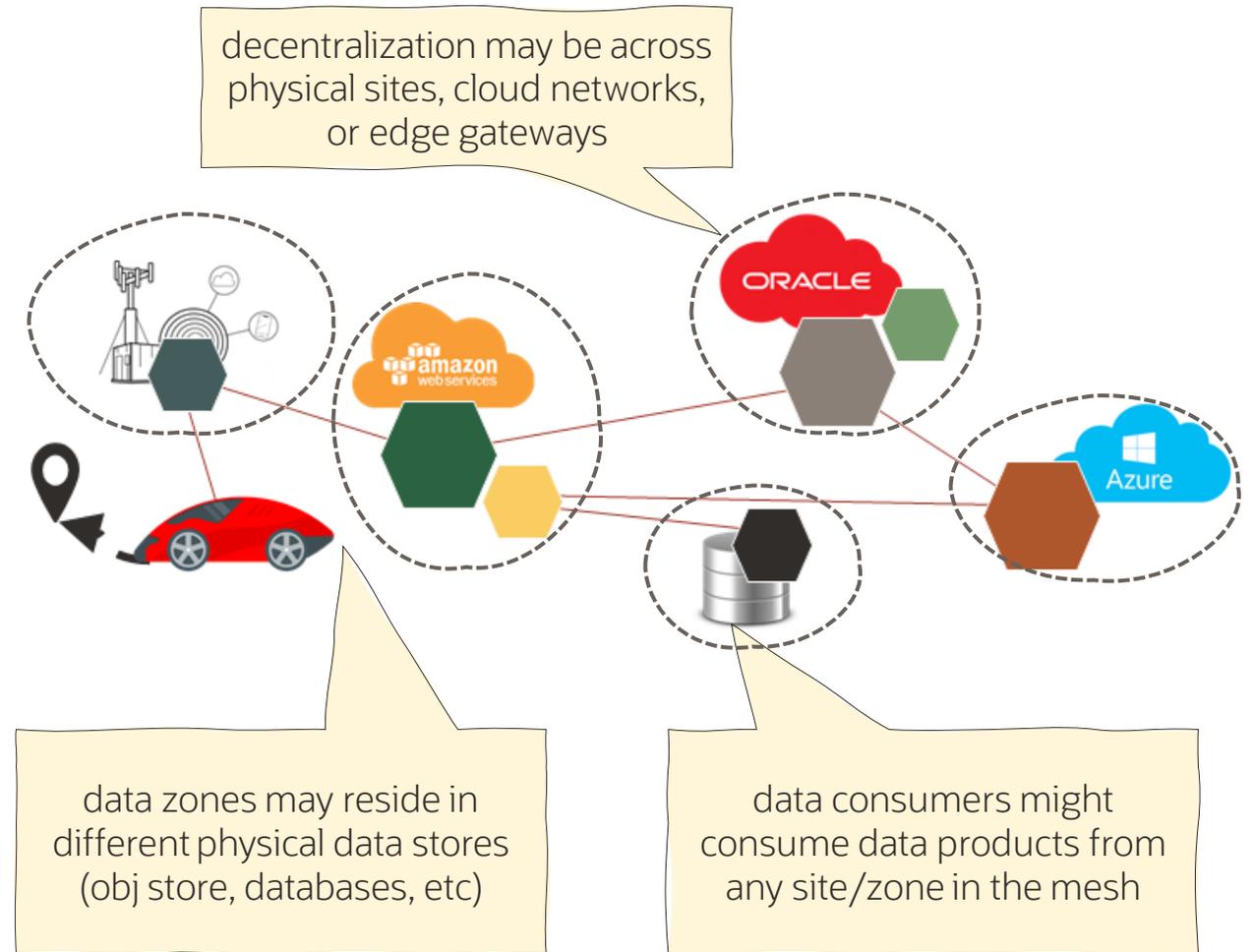
Decentralized IT systems are a modern reality, and with the rise of SaaS applications and public cloud infrastructure (IaaS), **decentralization of applications and data is here to stay.**

Application software architectures are shifting away from centralized monoliths and towards distributed microservices (a service mesh).

Data architecture will follow the same trend towards **decentralization**, with data becoming more distributed across a wider variety of physical sites and across many networks. We call this a Data Mesh.

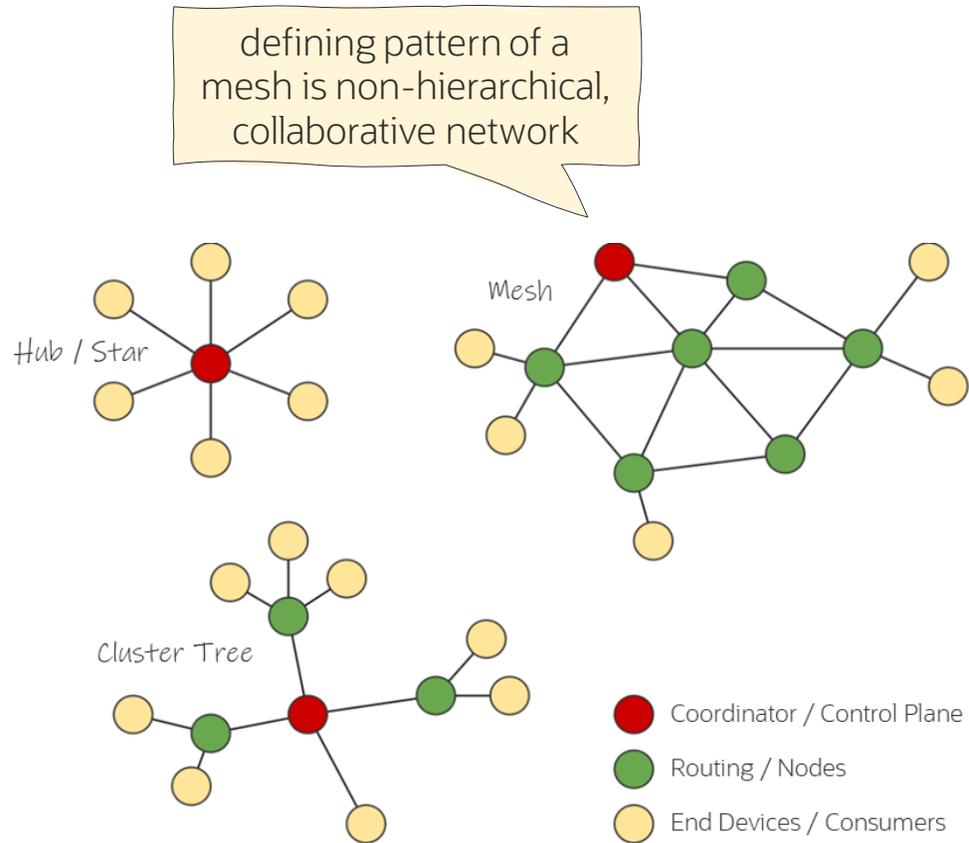
Distributed software is hard. Just as nobody does microservices architecture because it is easy, nobody should try Data Mesh believing it is simple. There are many good reasons and many benefits to having a modular decentralized data, but a monolithic and centralized data architecture is often simpler.

When the business benefits from decentralized data, Data Mesh patterns can keep the solution manageable.



# Data Mesh Attribute

## 2a.) MESH



The word ‘mesh’ means something specific – in tech, it is a particular kind of network topology setup so that a large group of non-hierarchical nodes can collaboratively work together.

Some common tech examples include:

- **WiFi Mesh** – many nodes working together for better coverage
- **ZWave/Zigbee** – low-energy smart home device networks
- **5G Mesh** – more reliable and resilient cell connections
- **Starlink** – satellite broadband mesh at global scale
- **Service Mesh** – a way to provide unified controls over decentralized microservices (application software)

Data Mesh is aligned to these mesh concepts, and provides a decentralized way of distributing data across virtual/physical networks and across vast distances.

Legacy data integration monoliths (such as ETL tools, data federation tools etc.) and even more recent public cloud services (such as AWS Glue) require highly centralized infrastructure.

A complete Data Mesh solution should be capable of operating in a multi-cloud framework, potentially spanning from on-premises, multiple public clouds, and even to the edge networks.

# Data Mesh Attribute

## 2b.) DISTRIBUTED SECURITY

In a world where data is highly distributed and decentralized, the role of information security is paramount. Unlike highly centralized monoliths, distributed systems must delegate out the activities necessary to authenticate and authorize various users to different levels of access. Securely delegating trust across networks is hard to do well.

Some considerations include:

- **Encryption at rest** – as data/events are written to storage
  - *Eg; mTLS, Certificates, SSO, Secret stores and data vaults*
- **Distributed authentication** – for services and data stores
  - *Eg; mTLS, Certificates, SSO, Secret stores and data vaults*
- **Encryption in motion** – as data/events are flowing in-memory
- **Identity management** – LDAP/IAM type services, cross-platform
- **Distributed authorizations** – for service end-points to redact data
  - *For example: Open Policy Agent (OPA) <sup>15</sup> sidecar to place Policy Decision Point (PDP) within the container/K8S cluster where the microservice end point is processing. LDAP/IAM may be any JWT capable service.*
- **Deterministic masking** – to reliably and consistently obfuscate PII data

Security within any IT system can be difficult, and it is even more difficult to provide high security within distributed systems. However, **these are solved problems with known solutions.**

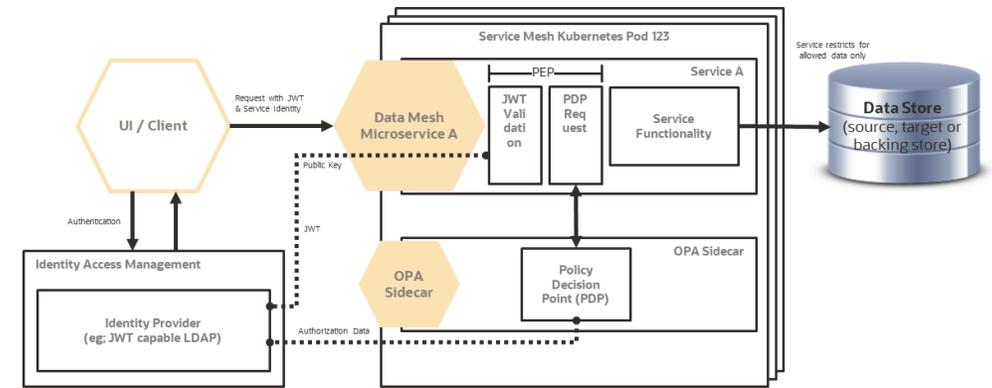


Figure 1: distributed authorizations using OPA sidecar in microservices

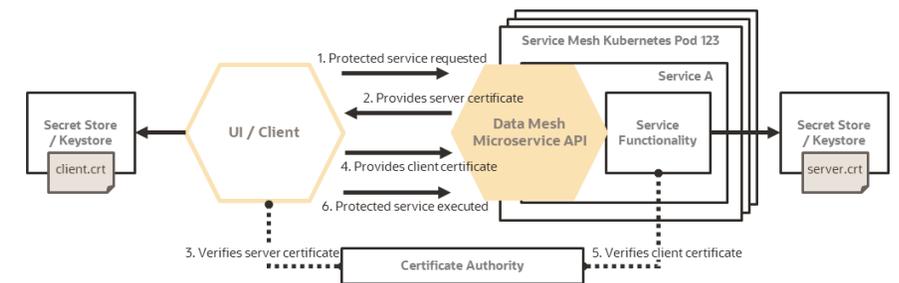
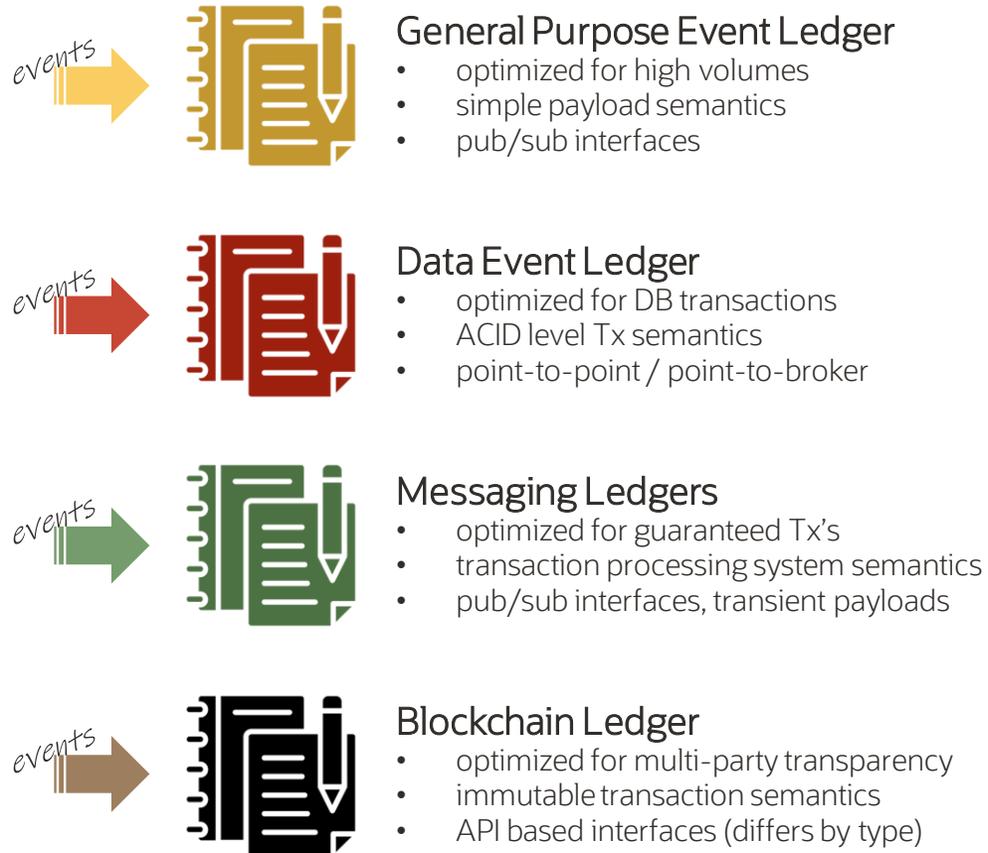


Figure 2: distributed mTLS authentication using secure certificates



# Data Mesh Attribute

## 3.) EVENT-DRIVEN DATA LEDGERS



*\*included for completeness, but not discussed in depth*

Ledgers are a fundamental component of making a distributed data architecture function. Just as with an accounting ledger, a data ledger records the transactions as they happen.

When we distribute the ledger, the data events become 'replayable' in any location. Some ledgers are a bit like an airplane flight recorder, used for high availability and disaster recovery.

Unlike centralized and monolithic data stores, distributed ledgers are purpose-built to keep track of atomic events and/or transactions that happen in other (external) systems.

A Data Mesh is not just one single kind of ledger, and can make use of different types of event-driven data ledgers, depending on the use cases and requirements:

- **General Purpose Event Ledger** – such as Kafka or Pulsar
- **Data Event Ledger** – distributed CDC/Replication tools
- **Messaging Middleware** – including ESB, MQ, JMS, and AQ
- **Blockchain Ledger** – for secure, multi-party transactions

Together, these ledgers can act as a sort of durable event log for the whole enterprise...providing a running list of data events happening on systems of record and systems of analytics.

# Data Mesh Attribute

## 4.) POLYGLOT DATA STREAMS

Data streams may vary by event types, payloads and different transaction semantics, a Data Mesh should support the necessary stream types for a variety of enterprise data workloads.

### Simple Events:

- Base64 / JSON – raw, schemaless events
- Raw Telemetry etc. – sparse events

### Basic App Logging / IoT Events:

- JSON / Protobuf – may have schema
- MQTT etc. – IoT specific protocols

### Application Business Process Events:

- SOAP/REST Events – XML/XSD, JSON etc.
- B2B etc. – exchange protocols & standards

### Data Events / Transactions:

- Logical Change Records – LCR, SCN, URID etc.
- Consistent Boundaries – commits vs. operations

### Telemetry Events (devices & things)



```
T3VyIG1pc3Npb24gaXMgdG8gaGVscCBw  
ZW9wbGUgc2VlIGRhdGEgaW4gbmV3IHdh  
eXMsIGRpc2NvdnVyaGVuc2lnaHRzLCB1  
bmXvY2sgZW5kbGVzcyBwb3NzaWJpbG10  
aWVzLg==
```

Simple, flat & record at a time

### App/Process Events (biz process & logging)



```
syntax = "proto3";  
package moviecatalog;  
message MovieItem {  
    string name = 1;  
    double price = 2;  
    bool inStock = 3;
```

Record at a time, records have simple schema

### Data Events (ACID transactions)



```
<?xml version="1.0" encoding="utf-8"?>  
<Root xmlns="http://www.acme.com">  
    <Customers>  
        <Customer CustomerID="GREAL">  
            <ContactName>Howard</ContactName>  
            <ContactTitle>Manager</ContactTitle>
```

May be deeply nested, complex schemas

Hdr-Ind	: E	<x45>	Partition	: .	<x04>
UndoFlag	: -	<x00>	BeforeAfter:	: A	<x41>
RecLength	: 64	<x0040>	IO Time	: 2011/01/24 14	
IOType	: 5	<x05>	OrigNode	: 255	<xff>
Trans Ind	: -	<x03>	FormatType	: R	<x52>
SyskeyLen	: 0	<x00>	Incomplete	: -	<x00>
AuditRBA	: 41	<x00>	AuditPos	: 92002584	
Continued	: N	<x00>	RecCount	: 1	<x01>

Operation type and time record was written

Source object → 2011/01/24 14:45:26.000.000 Insert Len 64 RBA 0

Name: DDIEC.DEPARTMENTS

After Image: 0000 000A 0000 0000 0000 0000 033E 0001 0012 0000 ! ... Partitio

Image type:

Follows DB log / transaction boundaries



# 4a.) STREAM DATA PROCESSING

Stream processing is how data is manipulated within an event stream. Unlike 'lambda functions' the stream processor maintains statefulness of data flows within a particular time window.

## Basic Data Filtering:

- *Thresholds, alerts, telemetry monitoring etc.*

## Simple ETL:

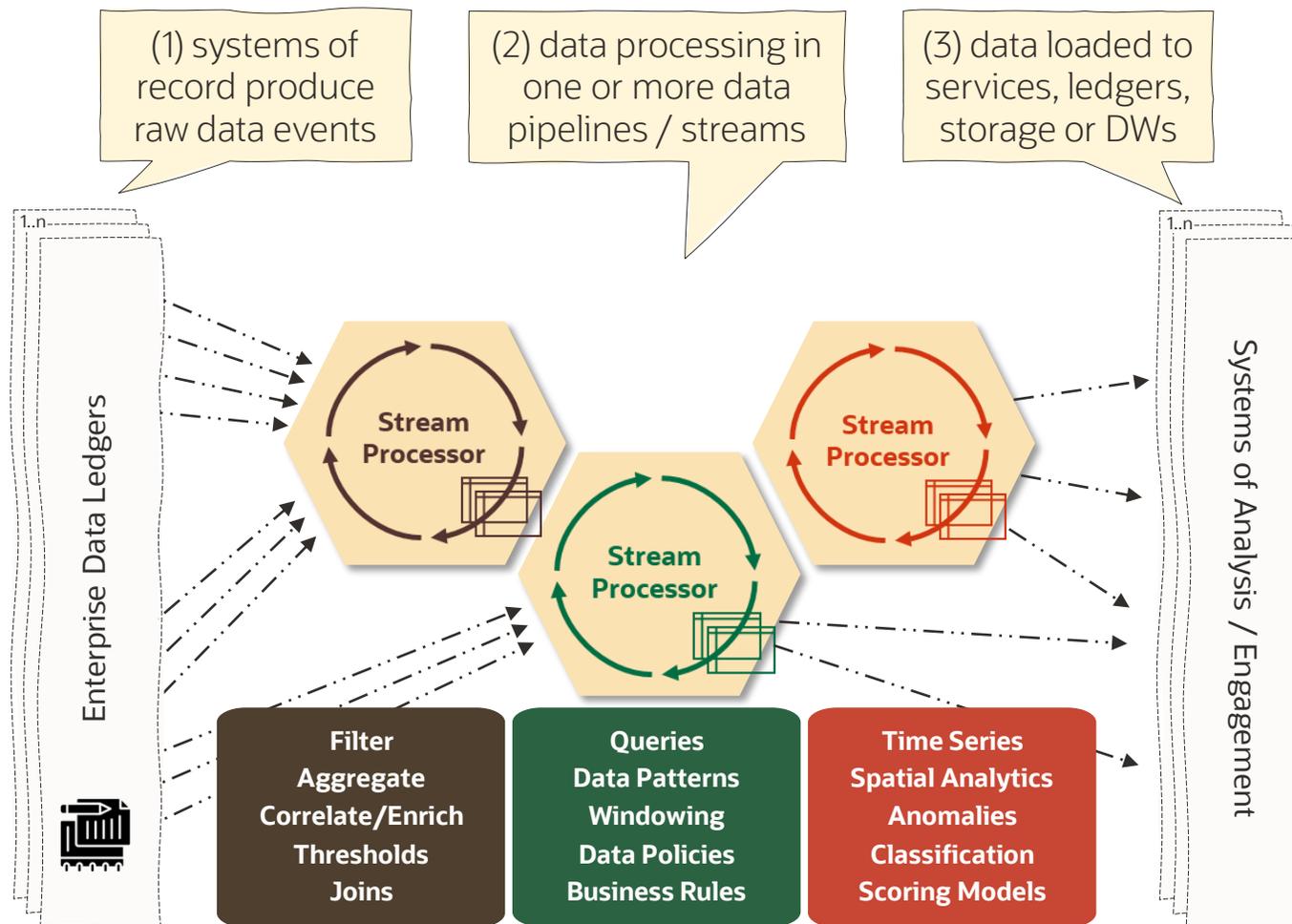
- *RegEx functions, math/logic, concatenation*
- *Record-by-record, substitutions, masking*

## CEP & Complex ETL:

- *Complex Event Processing (CEP)*
- *DML (ACID) processing, groups of tuples*
- *Aggregates, lookups, complex joins etc.*

## Stream Analytics:

- *Time series analytics, custom time windows*
- *Geospatial, machine learning and embedded AI*



# Data Mesh

## PATTERN ARCHETYPE

Operational Data

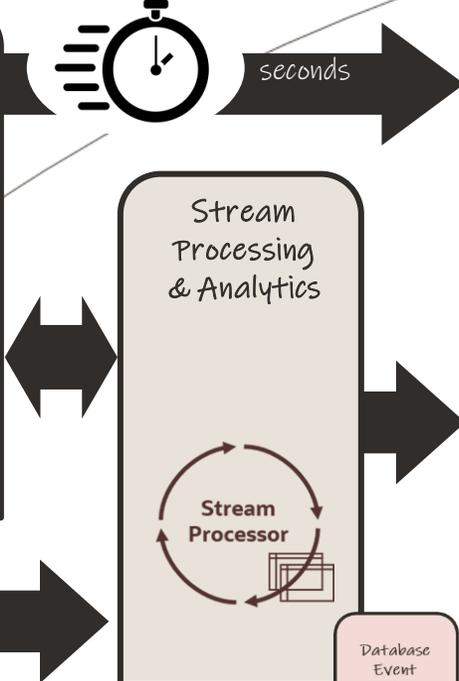
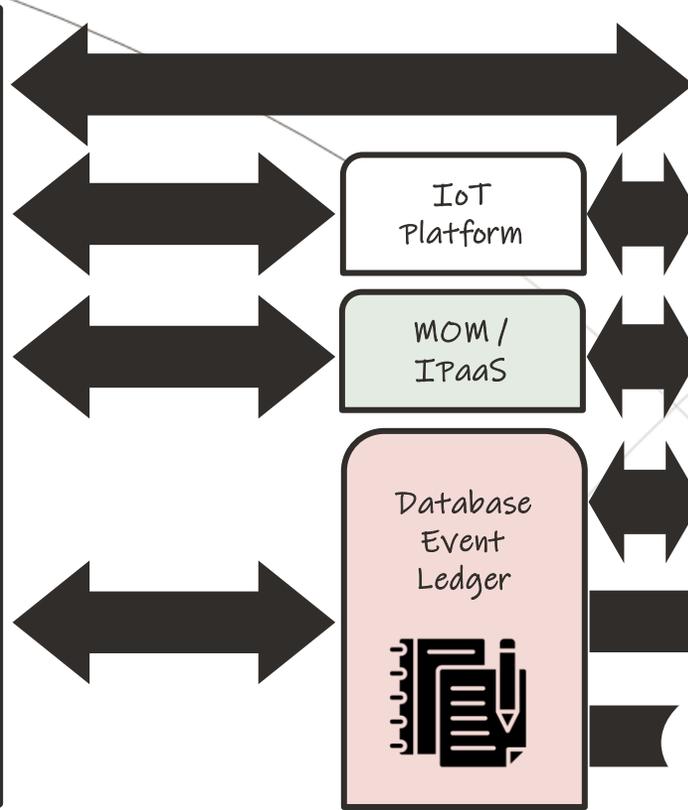
Analytic Data

IoT Events, Edge & Microservices

SaaS / App Events & Sys Logs



Operational Systems of Record



Systems of Analysis/Engagement

{ Data / Event Services }

Multi-model Database/s

Data Lake (house)

Data Warehouse



seconds



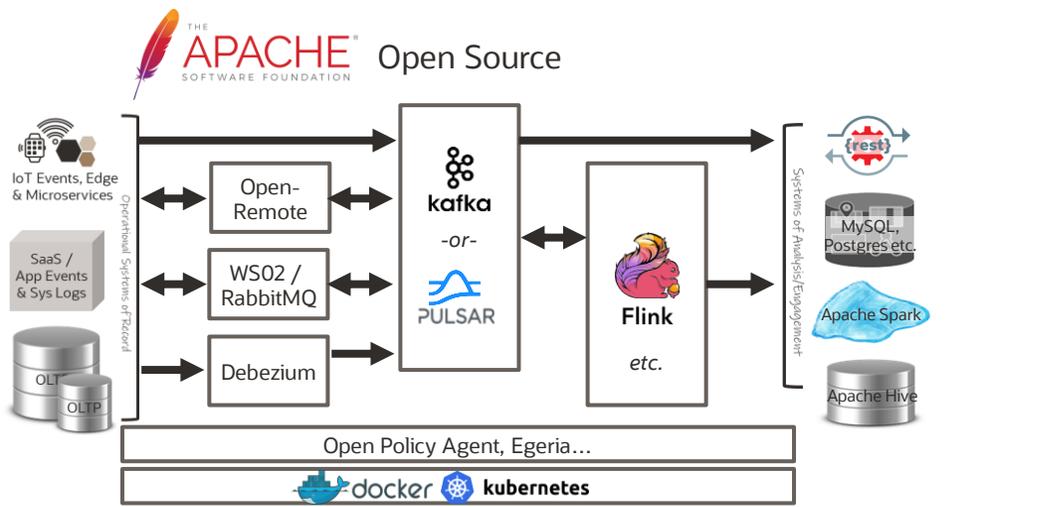
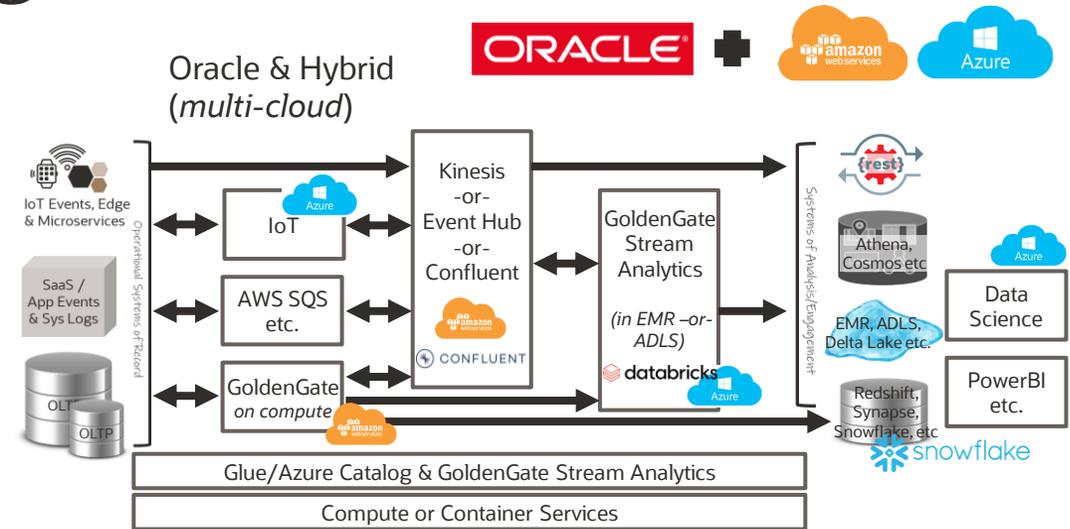
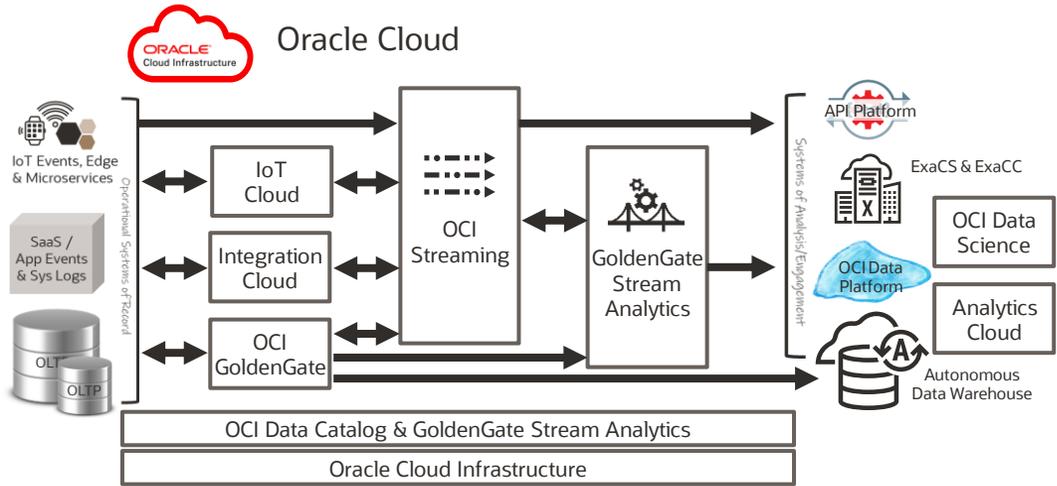
milliseconds

Governance – Security (distributed), Data Verification, Data Catalog, Registry, Policies

Serverless or Service Mesh (multi-cloud) Deployments



# CONCRETE EXAMPLES



## Noteworthy Technology Layers:

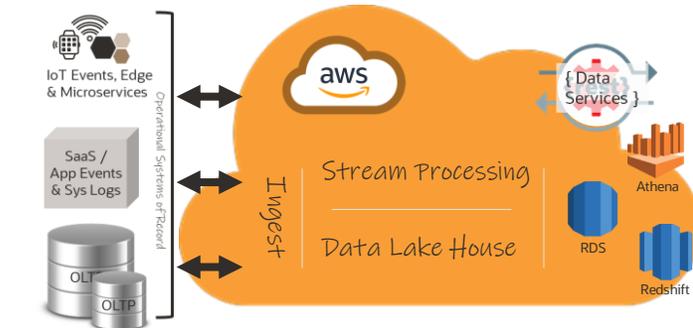
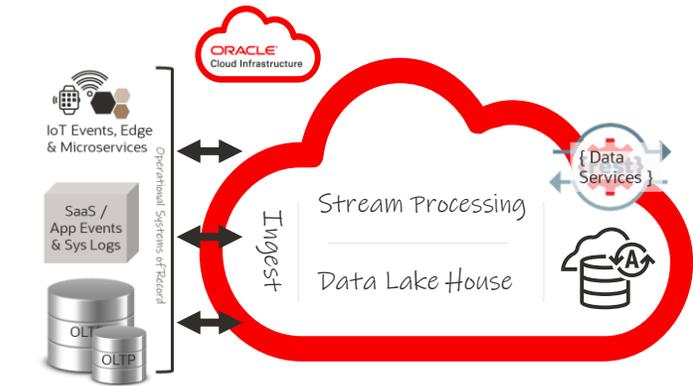
- **IoT/Edge** – for gateways, edge nodes, telemetry collection
- **Message-oriented Middleware** – for event-driven business process integrations
- **Data Events/CDC** – DB transaction events, full ACID consistency etc.
- **Event Streams** – scale-out and partitioned event store
- **Stream Processing, Analytics** – stateful, windowed complex stream processing
- **Security** – distributed authentication/authorization across VCNs
- **Data Catalog / Registry** – semantic alignment of entities, schemas, and registries
- **Data Verification** – auditable verification of data consistency across data stores
- **Serverless / Service Mesh** – depending on public cloud or self-operated



# SINGLE CLOUD OR MULTI-CLOUD

## Single Cloud:

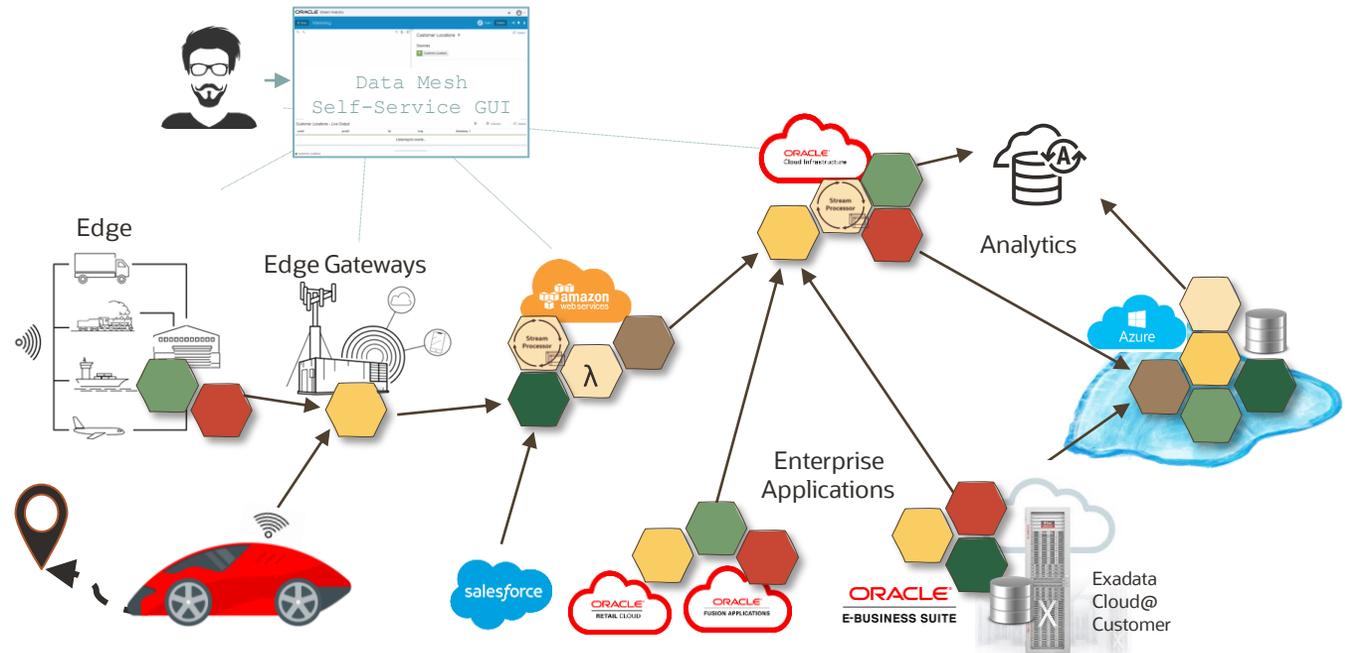
- **Simpler** – fewer networks, identity domains, etc
- **Serverless** – more opportunities to use ‘pay per use’ services
- **Homogeneous** – major commitment to single vendor solutions



## Multi-Cloud:

- **Decentralized** – introduces greater complexity (events, security, networking, etc)
- **Service Mesh & Serverless** – IT may have to operate ‘as a service’ containers & K8S
- **Heterogeneous** – empowers best-of-breed and greater portability, reuse of services

VS.



## Chapter 3

# Seven Case Studies

In consideration of the 7 use cases and 4 key technology attributes, there are good public examples of Data Mesh success.

Each of these examples are using distributed, decentralized, event-driven, real-time tech.

Several examples also leverage data product thinking, microservices, service mesh and stream processing architecture.

The Intuit logo, featuring the word "intuit" in a lowercase, blue, sans-serif font.The Netflix logo, featuring the word "NETFLIX" in a bold, red, sans-serif font.The SailGP logo, featuring the word "SAILGP" in a bold, red, italicized, sans-serif font.The PayPal logo, featuring a blue "P" icon followed by the word "PayPal" in a blue, sans-serif font.The LinkedIn logo, featuring the word "Linked" in a black, sans-serif font followed by a blue square containing a white lowercase "in".

# Data Mesh

## CASE STUDY CRITERIA

	Data Mesh
<b>People, Process and Methods:</b>	
Data Product Focus	●
<b>Technical Architecture Attributes:</b>	
Distributed Architecture	●
Event Driven Ledgers	●
ACID Support	●
Stream Oriented	●
Analytic Data Focus	●
Operational Data Focus	●
Physical & Logical Mesh	●

There is no single ‘perfect’ example of a Data Mesh.

Other software development and data architecture patterns, or technology categories exist and there remains substantial overlap among the most common concepts like Data Fabrics, Microservices Service Mesh, and Data Lake Houses.

For this document, we are considering Data Mesh as a type of Data Fabric. Case Studies should have *‘significant solution focus’* using technology with the following attributes:

- **Data Products** – driving cultural and process changes that affect cross-organizational data domains, and institutionalize strong management practices around data assets
- **Distributed Architecture** – decentralized, microservices-based software architecture patterns
- **Event Driven Ledgers** – durable running log of events to drive cross-domain integrations
- **ACID Support** – for polyglot streams, empowering correct and trusted data transactions
- **Stream-Oriented** – data processing on ‘data in motion’ to drive solution outcomes
- **Analytic Data Focus** – data pipelines or data products in the analytics domain (eg; OLAP)
- **Operational Data Focus** – solution focus on operational data outcomes (eg; OLTP)
- **Physical & Logical Mesh** – data is both physically and logically ‘meshed’ together

# Case Study

## INTUIT – DATA PRODUCT THINKING



Intuit has been an early proponent and leader in applying data product thinking to their enterprise data estate.<sup>16</sup> Cross-organizational alignment means that data products include people stakeholders, business processes, data pipelines, and well-defined APIs for consumption. Different domains may have internal or external data consumers, and data can take the form/shape required by the end consumer (eg; data lake tables vs. event bus topics, etc).

### People, Process and Methods:

Data Product Focus ✓

### Technical Architecture Attributes:

Distributed Architecture ✓

Event Driven Ledgers ✓

ACID Support ✓

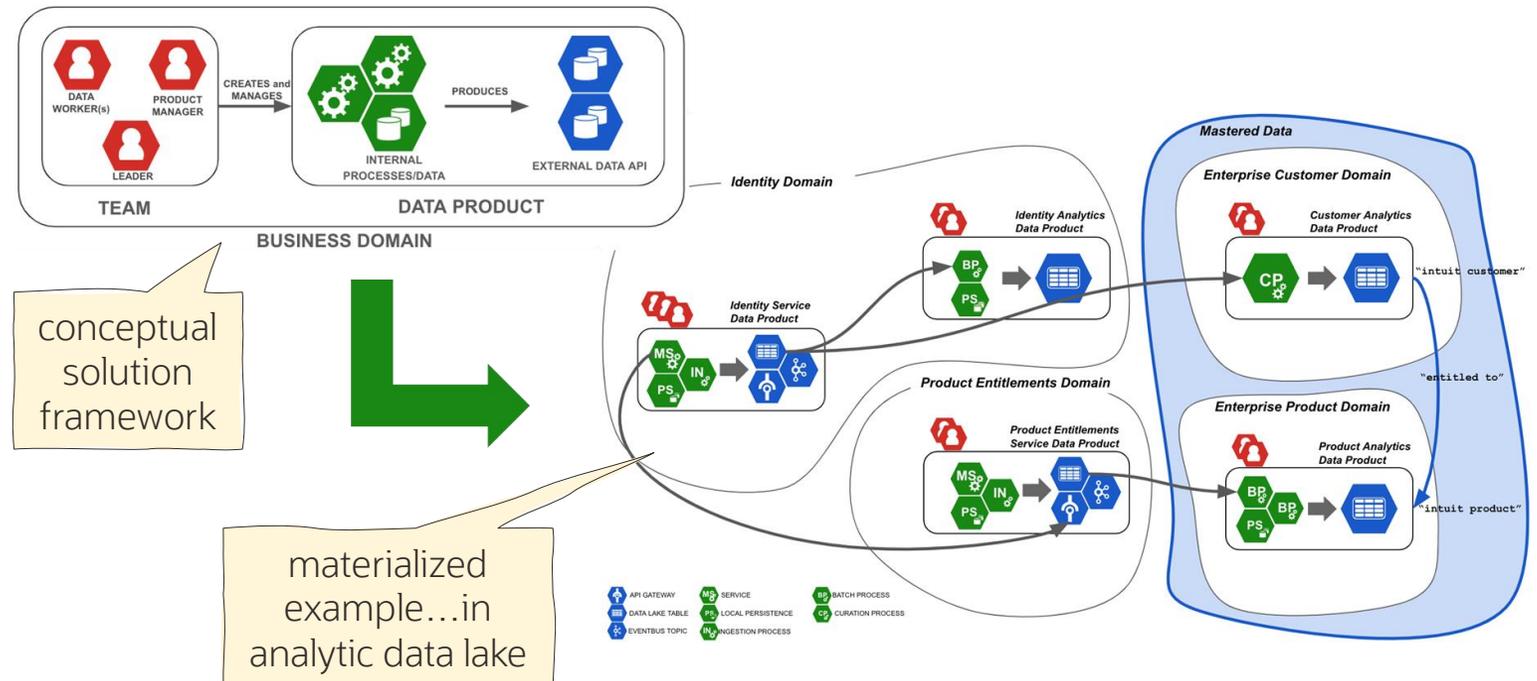
Stream Oriented ✓

Analytic Data Focus ✓

Operational Data Focus adjacent/GG

Physical & Logical Mesh ✓

GoldenGate Usage data event ledger, ingest to cloud and event bus



# Case Study

# NETFLIX – APPLICATION MODERNIZATION

## NETFLIX

### People, Process and Methods:

Data Product Focus    yes – adjacent <sup>18</sup>

### Technical Architecture Attributes:

Distributed Architecture    ✓

Event Driven Ledgers    ✓

ACID Support    ✓

Stream Oriented    n/a

Analytic Data Focus    n/a

Operational Data Focus    ✓

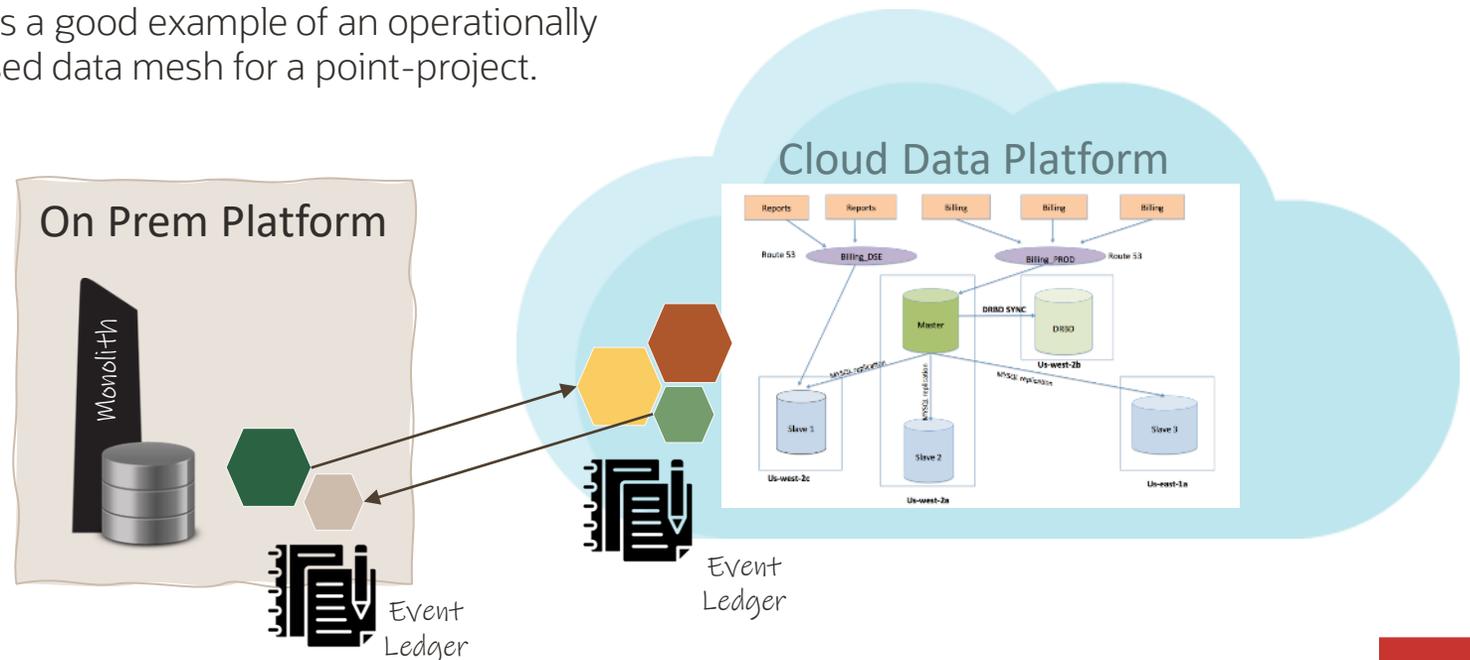
Physical & Logical Mesh    physical only

GoldenGate Use Case    data event ledger, bi-directional Tx-safe and fully consistent events

Netflix has frequently been at the cutting edge of new IT innovation and investment in data mesh is no different. Before the rise of popularity of the term, Netflix was already using a data mesh approach to perform online migration of operational apps (to the cloud), to avoid any outages that would affect customers. <sup>17</sup>

After an exhaustive review, Netflix chose an approach with several key data mesh attributes including a distributed architecture and event-based data ledgers. The target architecture was a modern microservices based application, and the real-time migration approach enabled a phased cutover approach to new platforms (infrastructure & DBs) without any downtime.

This is a good example of an operationally focused data mesh for a point-project.



# WELLS FARGO – DATA CONTINUITY



## People, Process and Methods:

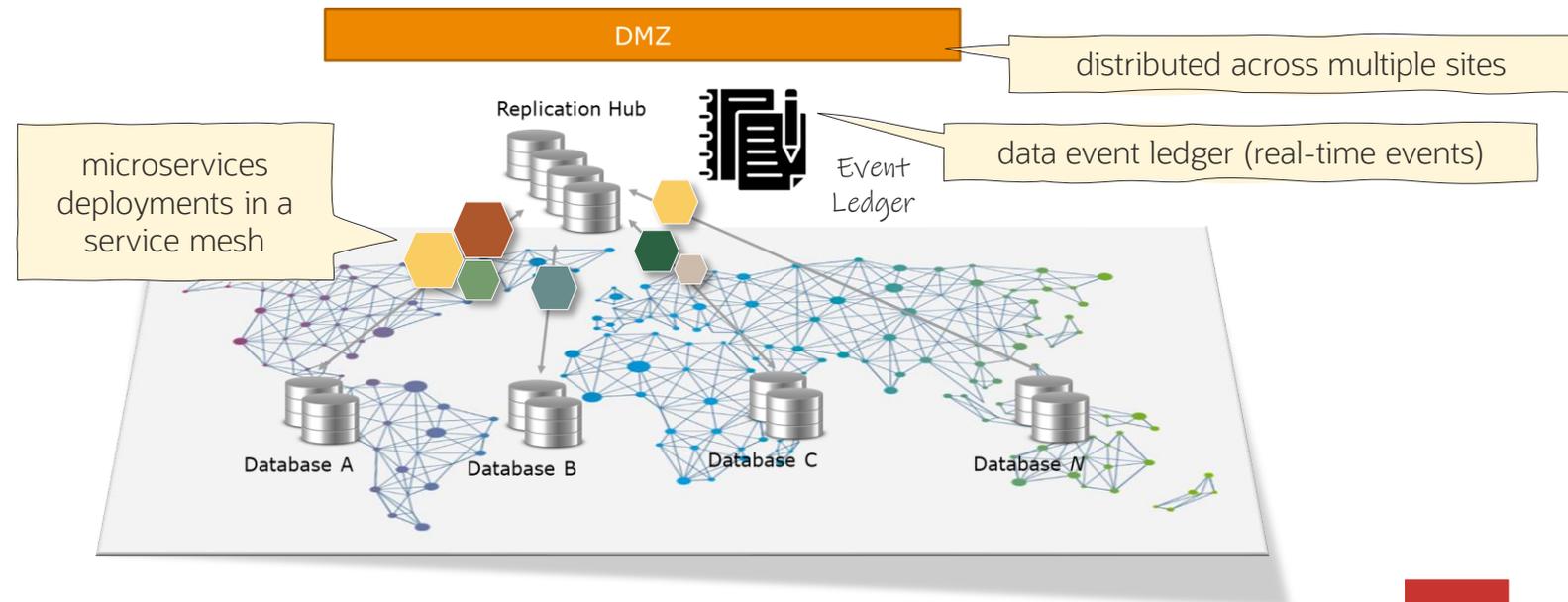
Data Product Focus    yes – adjacent <sup>20</sup>

## Technical Architecture Attributes:

Distributed Architecture	✓
Event Driven Ledgers	✓
ACID Support	✓
Stream Oriented	n/a
Analytic Data Focus	yes - adjacent
Operational Data Focus	✓
Physical & Logical Mesh	mainly physical
GoldenGate Use Case	data event ledger, fully consistent data events, microservice deployments

Wells Fargo is one of the largest banks in the world and has been heavily investing in data-driven digital transformation for many years. At the heart of data strategy is the need to ensure 100% continuity of data operations, and Wells Fargo has spoken about their use of GoldenGate microservices for these continuity use cases. <sup>19</sup>

Combining operational data events with analytics and data lakes simplifies the data architecture by reducing the number of ‘hops’ that the data must take prior to being prepared for analytics. A data mesh approach aims to reduce friction and wasted IT resources when joining up operational and analytic data.



# Case Study

## PAYPAL – MICROSERVICES PATTERNS



PayPal uses a modern microservices application architecture (distributed), and needed fast, 100% correct transactions moved asynchronously among services. They used a data mesh approach<sup>21</sup> with event-driven data ledgers to de-centralize transactions with zero data loss or corruption.

They considered alternatives like ‘event sourcing’ and ‘multi-phase commits’ but could not guarantee zero data loss, the event-driven data ledgers from GoldenGate provided the trust, correctness, and performance needed for a distributed data mesh.

### People, Process and Methods:

Data Product Focus ✓

### Technical Architecture Attributes:

Distributed Architecture ✓

Event Driven Ledgers ✓

ACID Support ✓

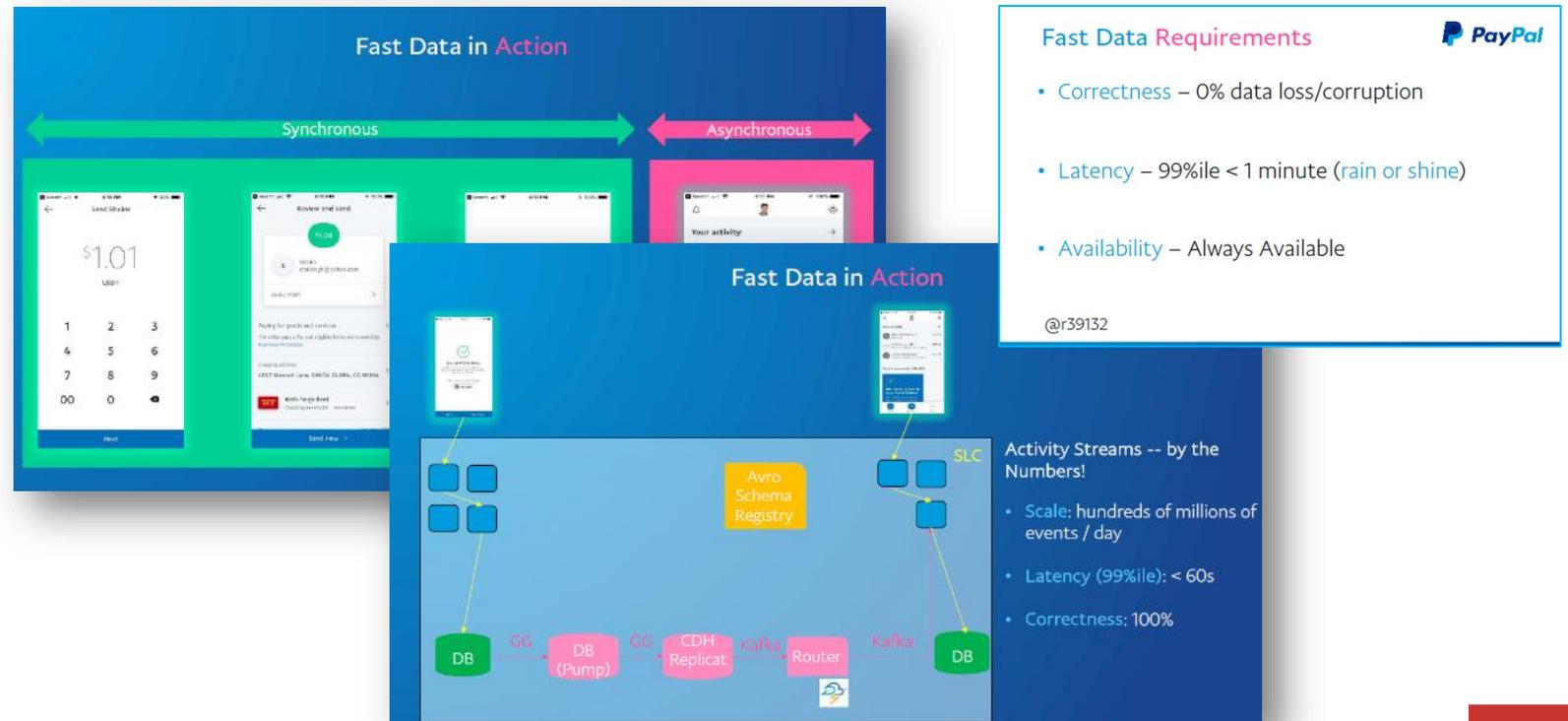
Stream Oriented yes - adjacent

Analytic Data Focus yes - adjacent

Operational Data Focus ✓

Physical & Logical Mesh domain driven

GoldenGate Use Case data event ledger, fully consistent data events, transaction outbox



# Case Study

## WESTERN DIGITAL – INTEGRATION



Western Digital has been continuously investing in digital transformation goals for several years, including the shift towards cloud-first and data-driven business practices. As a part of that journey, they make extensive use of event-driven integration tech from Oracle – including Integration Cloud and GoldenGate.<sup>22</sup>

These data mesh capabilities provide a distributed, event-driven architecture that help in both operational and analytic use cases. Operationally, the integration tech is used to modernize the ERP platforms and ultimately reduce operating costs. For analytics, the shift to a real-time cloud business means being able to continuously stream data events from applications into reporting data marts, data warehouses and data lakes.

### People, Process and Methods:

Data Product Focus      yes - adjacent

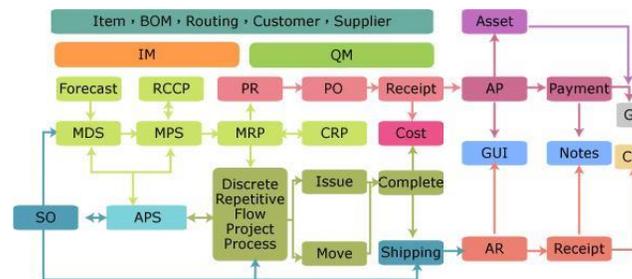
### Technical Architecture Attributes:

Distributed Architecture	✓
Event Driven Ledgers	✓
ACID Support	✓
Stream Oriented	n/a
Analytic Data Focus	✓
Operational Data Focus	✓
Physical & Logical Mesh	✓

GoldenGate Use Case  
data event ledger, stream SaaS data events into business reporting tools

### Operational

drive cost reductions and operating efficiency by consolidating ERP and moving applications to cloud<sup>23</sup>



### Edge and Analytics

align core operations data to Fast Data and Big Data initiatives – impacting customer systems of engagement as well as data science / AI initiatives<sup>24</sup>



# Case Study

## LINKEDIN – STREAMING INGEST



LinkedIn created and operates one of the world's largest Apache Kafka implementations, using the tech for both operational and analytic data events.<sup>25</sup> For 100's of applications that produce database events, those (billions per day) data events are captured and ingested to Kafka using GoldenGate data ledger.<sup>26</sup>

A modern distributed data mesh must work with raw data events as they happen. When DBs commit transactions, those data events become the source/provider data from the systems of record (SoR). Downstream stream processing tools (eg; Samza, Flink, GoldenGate Stream Analytics) can then process these data events within milliseconds of their origin.

### People, Process and Methods:

Data Product Focus



### Technical Architecture Attributes:

Distributed Architecture



Event Driven Ledgers



ACID Support



Stream Oriented



Analytic Data Focus



Operational Data Focus



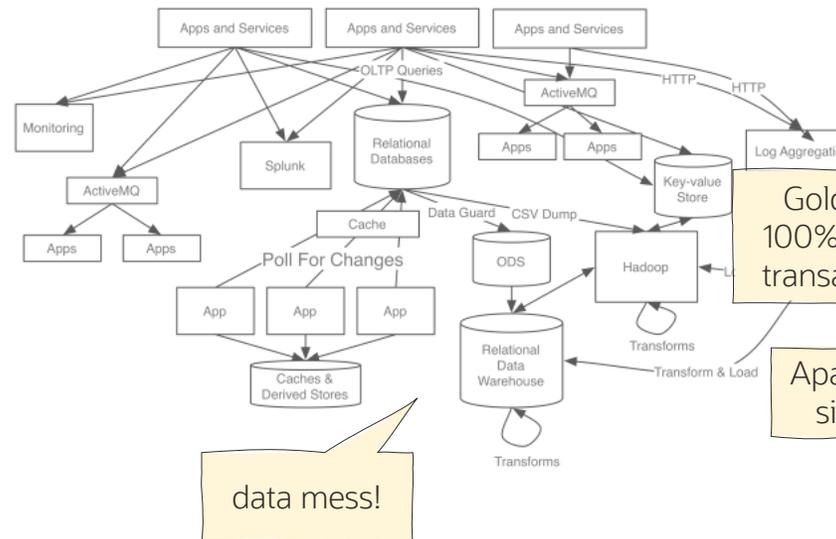
Physical & Logical Mesh



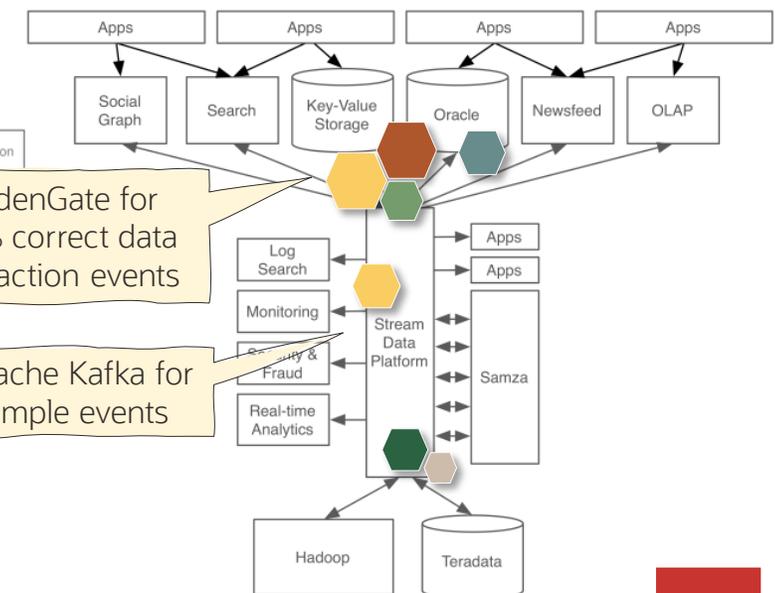
GoldenGate Use Case

data event ledger, stream DB events (DML & DDL) into Apache Kafka

Before:



After:



# Case Study

## SAILGP – STREAMING ANALYTICS

# SAILGP

### People, Process and Methods:

Data Product Focus



### Technical Architecture Attributes:

Distributed Architecture



Event Driven Ledgers



ACID Support

ingest to DW

Stream Oriented



Analytic Data Focus



Operational Data Focus



Physical & Logical Mesh

physical data

GoldenGate Use Case

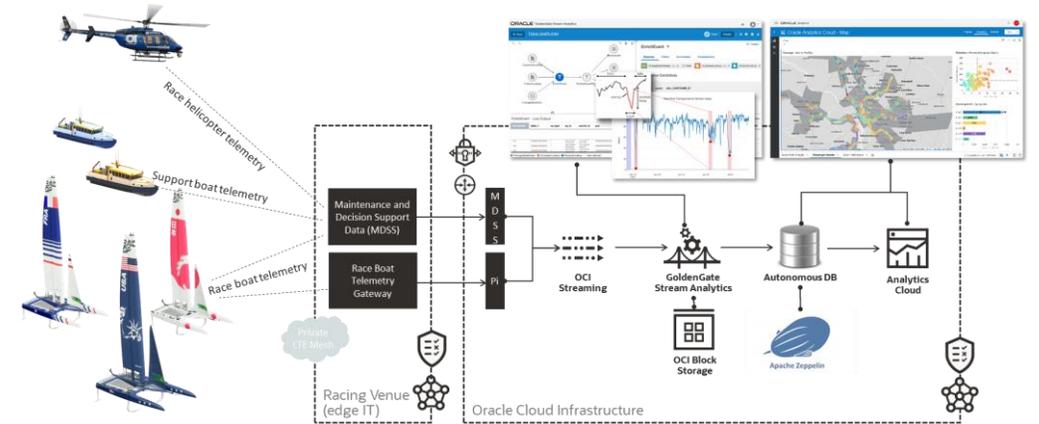
stream analytics, real-time event correlation, ETL, analysis and ingest to DW

SailGP runs one of the most exciting race venues in the world, with high tech and high-speed sail boats. Live race data and analytics are provided within milliseconds using data mesh tech.<sup>27</sup>

Distributed edge technology links race boat, support boat and race helicopter data into streaming pipelines.

Telemetry data is streamed into nearby clouds for real-time ETL, analytics and ingest to cloud data warehouse.

Data mesh tech uses GoldenGate and Kafka (Oracle Streaming). Stream analytics are used in real-time on race day to assist with support crews and broadcast networks.



# COMPARE AND CONTRAST

[best] ●●●○ [worst]

	Data Fabric			App-Dev-Integration		Analytic Data Store	
	Data Mesh	Data Integration	Meta-Catalog	Microservices	Messaging	Data Lake House	Distributed DW
<b>People, Process and Methods:</b>							
Data Product Focus	●	●	●	○	○	●	●
<b>Technical Architecture Attributes:</b>							
Distributed Architecture	●	○	●	●	●	○	●
Event Driven Ledgers	●	○	○	●	●	○	○
ACID Support	●	●	○	○	●	●	●
Stream Oriented	●	○	○	○	○	●	○
Analytic Data Focus	●	●	●	○	○	●	●
Operational Data Focus	●	○	●	●	●	○	○
Physical & Logical Mesh	●	●	○	○	●	●	○



# Data Mesh

## BUSINESS OUTCOMES

### Overall Benefits

**Faster, Data-Driven Innovation Cycles**

**Reduce Costs for Mission-Critical Data Operations**



### Operational Outcomes

**Multi-cloud data liquidity**

- unlock data capital to flow freely

**Real-time data sharing**

- Ops-to-Ops & Ops-to-Analytics

**Edge, location-based data services**

- correlate IRL device/data events

**Trusted microservices data interchange**

- “event sourcing” with correct data
- DataOps and CI/CD for Data

**Uninterrupted continuity**

- >99.999% up-time SLAs
- cloud migrations

### Analytic Outcomes

**Automate and simplify data products**

- multi-model data sets

**Time series data analysis**

- deltas / changed records
- event-by-event fidelity

**Eliminate full data copies for ODS'**

- log-based ledgers and pipelines

**Distributed data lakes & warehouses**

- hybrid / multi-cloud / global
- streaming integration / ETL

**Predictive Analytics**

- Data monetization, new ‘data services’ for sale

# Bringing it all together

Digital transformation is very, very hard and most will fail at it.<sup>5</sup> Technology, software design and data architecture are becoming increasingly more distributed, as modern techniques move away from highly-centralized and monolithic styles.

Data Mesh is a new concept for data. It is at core a cultural mindset shift to put the needs of data consumers first. It is also a real technology shift, elevating the platforms and services that empower a decentralized data architecture. Data Mesh is a deliberate shift towards highly distributed and real-time data events, as opposed to monolithic, centralized and batch style data processing. Four important attributes of Data Mesh include:

1. **Data Product Thinking** – data consumer needs ahead of IT
2. **Decentralized Data Architectures** – a distributed mesh topology
3. **Event-Driven Data Ledgers** – logs as the principal interchange
4. **Polyglot Data Streaming** – real-time processing for all data types

Use cases for Data Mesh encompass both operational data and analytic data, which is one key difference from conventional Data

Lakes/Lakehouse, and Data Warehouses. This alignment of operational and analytic data domains is a critical enabler for the need to drive more self-service for the data consumer. Modern data platform technology can help to remove the middleman in connecting data producers directly to data consumers.

Oracle has long been the industry leader in mission critical data solutions, and has fielded some of the most modern capabilities to empower a trusted Data Mesh:

- Gen2 public cloud infrastructure with >33 active regions
- Multi-model database for ‘shape-shifting’ data products
- Microservices-based data event ledger for all data stores
- Multi-cloud stream processing for real-time trusted data
- API platform, modern app-dev and self-service tools
- Analytics, data visualization and cloud-native data science

For more information, download the Data Mesh Tech Paper: <https://www.oracle.com/a/ocom/docs/techbrief-enterprisedatameshandgoldengate.pdf>

Our mission is to help people see  
data in new ways, discover insights,  
unlock endless possibilities.



# Endnotes

1. 99.999% availability: <https://www.oracle.com/a/tech/docs/maa-goldengate-hub.pdf>
2. 10x faster innovation cycles, shifting away from batch ETL (eliminate batch windows), to continuous transformation and loading (CTL) via streaming ingest
3. Data derived from real world discussions with customers who have adopted the methodologies and tools described in this document
4. Most time and cost go into integration efforts: <https://www.gartner.com/smarterwithgartner/use-a-hybrid-integration-approach-to-empower-digital-transformation/>
5. 70-80% of digital transformations fail: <https://www.bcg.com/publications/2020/increasing-odds-of-success-in-digital-transformation>
6. Cloud lock in is real: <https://www.infoworld.com/article/3623721/cloud-lock-in-is-real.html>
7. Cloud can become more costly: <https://a16z.com/2021/05/27/cost-of-cloud-paradox-market-cap-cloud-lifecycle-scale-growth-repatriation-optimization/>
8. Data lakes rarely succeed: <https://www.datanami.com/2021/05/07/drowning-in-a-data-lake-gartner-analyst-offers-a-life-preserver/>
9. From Monolithic Data Lake to Distributed Data Mesh: <https://martinfowler.com/articles/data-monolith-to-mesh.html>
10. Cost of operational outages are rising: <https://www.nextgov.com/ideas/2021/03/commercial-cloud-outages-are-wake-call/172731/>
11. Integrate ops and analytics for better decisions: <https://mitsloan.mit.edu/ideas-made-to-matter/digital-transformation-has-evolved-heres-whats-new>
12. Transaction Outbox pattern: <https://microservices.io/patterns/data/transactional-outbox.html>
13. Strangler Fig migrations: <https://martinfowler.com/bliki/StranglerFigApplication.html>
14. Bounded Context in microservices: <https://martinfowler.com/bliki/BoundedContext.html>
15. Open Policy Agent (OPA): <https://www.openpolicyagent.org/>
16. Intuit Data Products: <https://medium.com/intuit-engineering/intuits-data-mesh-strategy-778e3edaa017>
17. Netflix Migration of Billing App: <https://medium.com/netflix-techblog/netflix-billing-migration-to-aws-part-iii-7d94ab9d1f59>
18. Netflix Data Productization / Inverting: [https://canvas.stanford.edu/files/5342788/download?download\\_frd=1](https://canvas.stanford.edu/files/5342788/download?download_frd=1)
19. Wells Fargo Data Continuity: Oracle OpenWorld 2018, GoldenGate Microservices joint presentation with Wells Fargo, Joe DiCario
20. Wells Fargo Data Productization: <https://medium.com/@kshi/data-transformation-of-wells-fargo-en-f025843f5e2d>
21. PayPal Microservices CDC Events: <https://www.slideshare.net/r39132/big-data-fast-data-paypal-yow-2018>
22. Western Digital Real-time Integrations: <https://blogs.oracle.com/cloud-platform/western-digital-achieves-faster-integration-with-oracle-integration-cloud>
23. Western Digital on ERP Consolidation: <https://www.house-listing.com/technology/201903/298492.html>
24. Western Digital Fast Data and Big Data graphics: <https://gestaltit.com/tech-field-day/gestalt/western-digital-data-vision-at-sfd18/>
25. LinkedIn Use of Apache Kafka: <https://www.confluent.io/blog/event-streaming-platform-1/>
26. LinkedIn Use of Oracle GoldenGate for Streaming Ingest: Oracle OpenWorld 2019, GoldenGate joint presentation with LinkedIn
27. SailGP Stream Analytics: <https://www.oracle.com/news/announcement/sailgp-launches-second-season-with-oracle-cloud-041521.html>

ORACLE

## Connect with us



### Integrated Cloud Applications and Platform Services

Copyright © 2021, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.



Oracle is committed to developing practices and products that help protect the environment.