Data Lakehouse, Data Mesh, and Data Fabric
(the alphabet soup of data architectures)

James Serra
Data & AI Solution Architect
Microsoft
jamesserra3@gmail.com
Blog: JamesSerra.com

I tried to figure out all these data platform buzzwords on my own…

And ended up passed-out drunk in a Denny's parking lot

Let's prevent that from happening…

What is a Data Warehouse and why use one?

A data warehouse is where you store data from multiple data sources to be used for historical and trend analysis reporting. It acts as a central repository for many subject areas and contains the "single version of truth". It is NOT to be used for OLTP applications.

Reasons for a data warehouse:

- **Reduce stress on production system**
- **Optimized for read access, sequential disk scans**
- Integrate many sources of data
- Keep historical records (no need to save hardcopy reports)
- Restructure/rename tables and fields, model data
- Protect against source system upgrades
- Use Master Data Management, including hierarchies
- No IT involvement needed for users to create reports
- Improve data quality and plugs holes in source systems
- One version of the truth
- Easy to create BI solutions on top of it (i.e. Azure Analysis Services Cubes)
- Don’t need to provide security access for many users to the production systems
- Make better business decisions by getting greater insights into your company

Why You Need a Data Warehouse

What is a data lake and why use one?

A schema-on-read storage repository that holds a vast amount of raw data in its native format until it is needed.

Reasons for a data lake:

- Inexpensively store unlimited data
- Centralized place for multiple subjects (single version of the truth)
- Collect all data "just in case" (data hoarding). The data lake is a good place for data that you "might" use down the road
- Easy integration of differently-structured data
- Store data with no modeling – "Schema on read"
- Complements enterprise data warehouse (EDW)
- Frees up expensive EDW resources for queries instead of using EDW resources for transformations (avoiding user contention)
- Wanting to use technologies/tools (i.e Databricks) to refine/filter data that do the refinement quicker/better than your EDW
- Quick user access to data for power users/data scientists (allowing for faster ROI)
- Data exploration to see if data valuable before writing ETL and schema for relational database, or use for one-time report
- Allows use of Hadoop tools such as ETL and extreme analytics
- Place to land IoT streaming data
- On-line archive or backup for data warehouse data (i.e. keep three years of data in DW and have older data in data lake with an external table pointing to it)
- With Hadoop/ADLS, high availability and disaster recovery built in
- It can ingest large files quickly and provide data redundancy
- ELT jobs on EDW are taking too long because of increasing data volumes and increasing rate of ingesting (velocity), so offload some of them to the Hadoop data lake
- Have a backup of the raw data in case you need to load it again due to an ETL error (and not have to go back to the source). You can keep a long history of raw data
- Allows for data to be used many times for different analytic needs and use cases
- Cost savings and faster transformations: storage tiers with lifecycle management; separation of storage and compute resources allowing multiple instances of different sizes working with the same data simultaneously vs scaling data warehouse; low-cost storage for raw data saving space on the EDW
- Extreme performance for transformations by having multiple compute options each accessing different folders containing data
- The ability for an end-user or product to easily access the data from any location

Data Lake with DW use cases

Data Lake

Staging & preparation

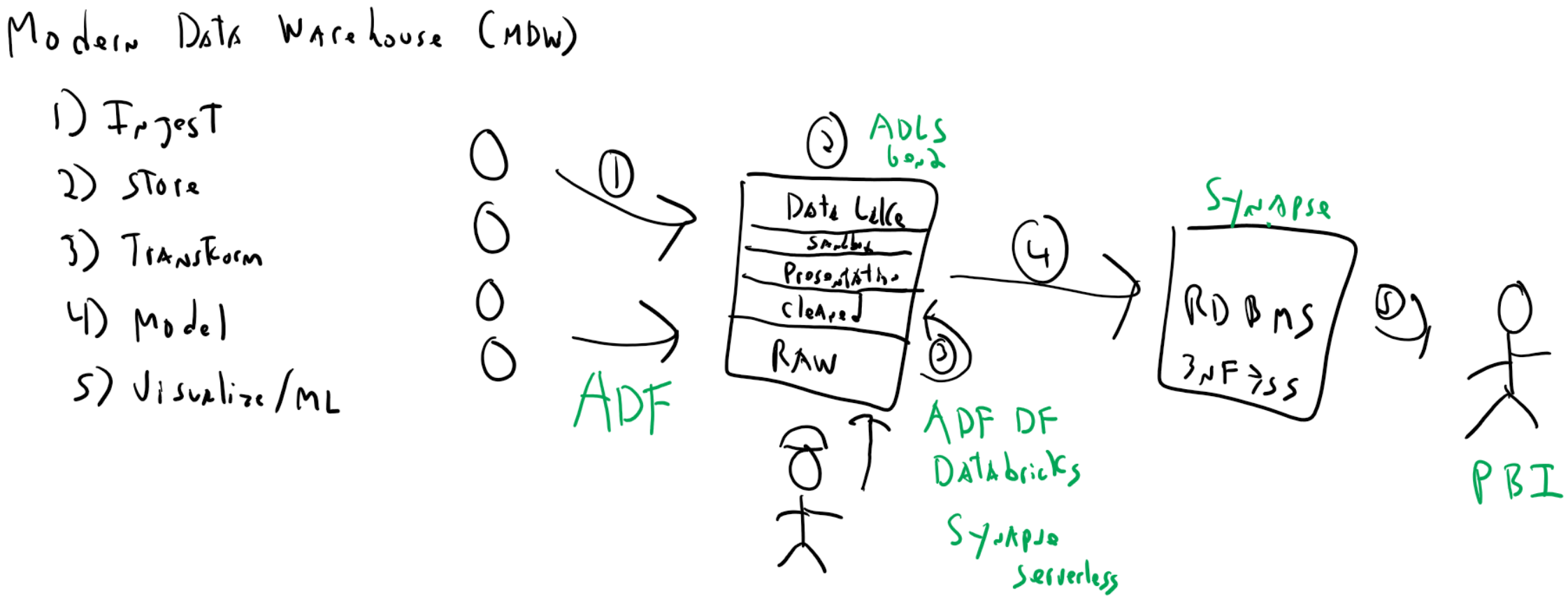
- Data scientists/Power users
- Batch processing
- Data refinement/cleaning
- ETL workloads
- Store older/backup data
- Sandbox for data exploration
- One-time reports
- Quick access to data
- Don’t know questions

Data Warehouse

Serving, Security & Compliance

- Business people
- Low latency
- Complex joins
- Interactive ad-hoc query
- High number of users
- Additional security
- Large support for tools
- Dashboards
- Easily create reports (Self-service BI)
- Know questions

Modern Data Warehouse



Data Fabric

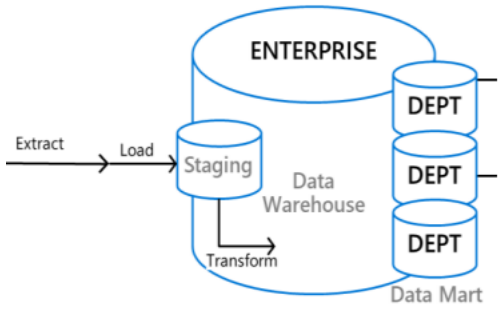
Data Fabric adds to a modern data warehouse:

- Data access
- Data policies
- Metadata catalog/Lineage
- Master Data Management (MDM)
- Data virtualization
- Real-time processing
- Data scientist tools
- APIs
- Building blocks/Services
- Products

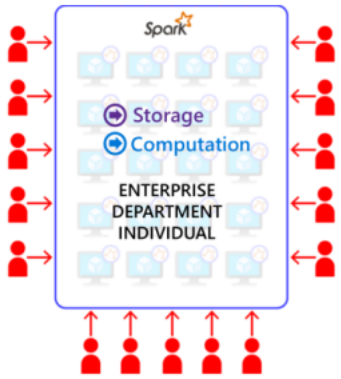
Data Fabric defined

Data Lakehouse

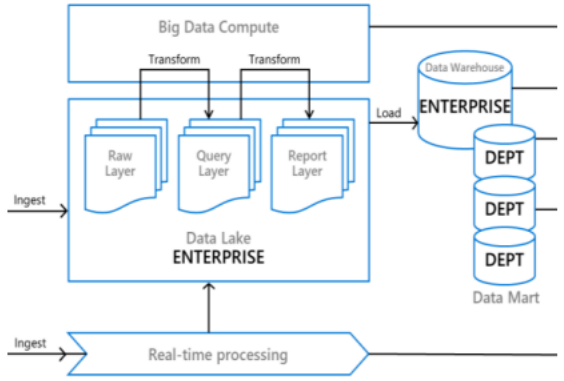
Late 1980s
Data Warehouse



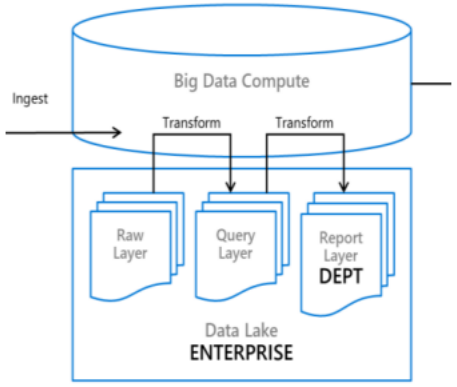
Late 2000s
Data Lake



Mid 2010s
Cloud Data Platform



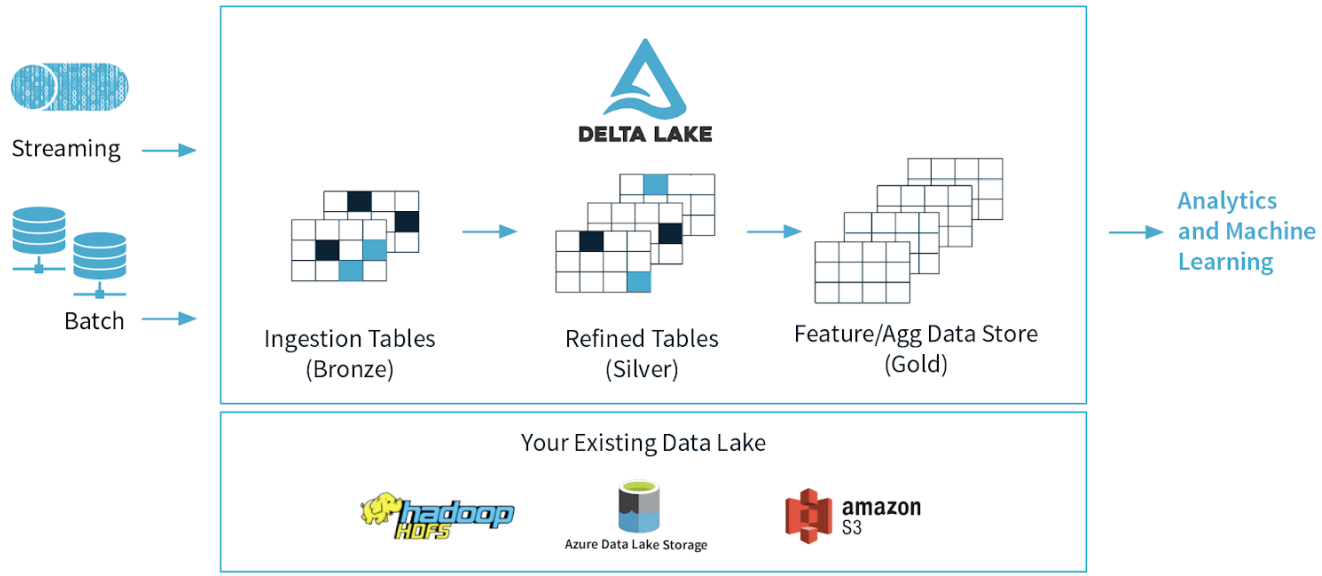
2020
Data Lakehouse



Delta Lake

Top features:

- ACID transactions
- Time travel (data versioning enables rollbacks, audit trail)
- Streaming and batch unification
- Schema enforcement
- Upserts and deletes
- Performance improvement



[Databricks Delta Lake]

Use cases for Data Lakehouse

Today's data architectures commonly suffer from four problems:

- Reliability: Keeping the data lake and warehouse consistent
- Data staleness: Data in warehouse is older
- Limited support for advanced analytics: Top ML systems don't work well on warehouses
- Total cost of ownership: Extra cost for data copied to warehouse

Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics

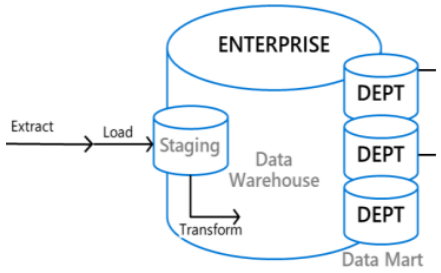
Concerns skipping relational database

- Speed: Relational databases faster, especially MPP
- Security: No RLS, column-level, dynamic data masking
- Complexity: Metadata separate from data, file-based world
- Missing features: Referential integrity, TDE, workload management; other features require locked into Spark
- People used to using a relational database

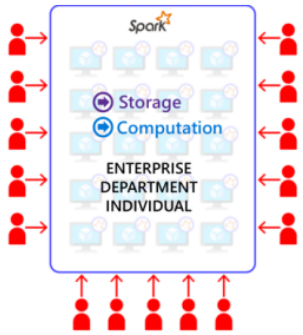
Azure Synapse: starting to see data lake only solutions because can use T-SQL, Power BI (speed, RLS)

Data Mesh

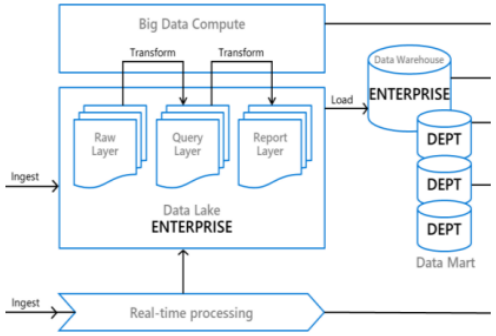
Late 1980s
Data Warehouse



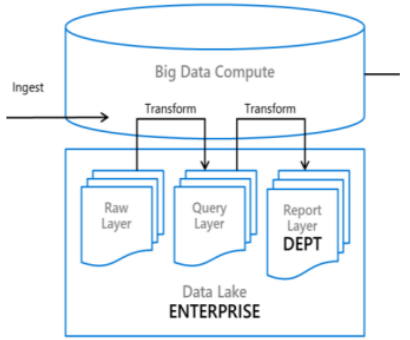
Late 2000s
Data Lake



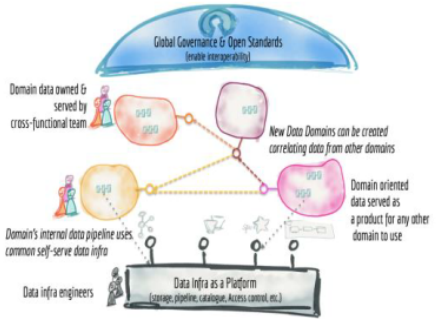
Mid 2010s
Cloud Data Platform



2020
Data Lakehouse



2021
Data Mesh??



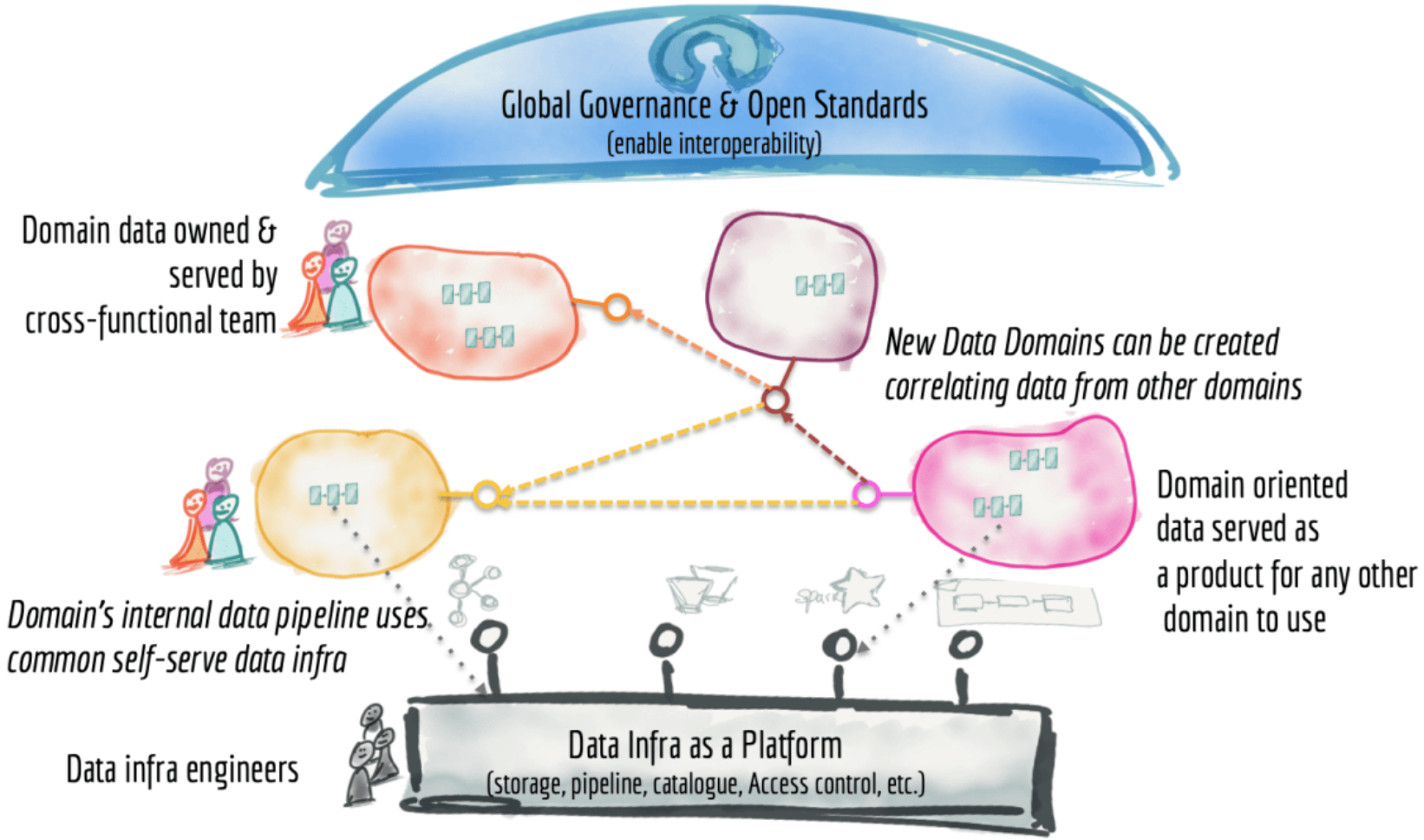
Centralization

Decentralization

Data Mesh

It's a mindset shift where you go from:

- Centralized ownership to decentralized ownership
- Pipelines as first-class concern to domain data as first-class concern
- Data as a by-product to data as a product
- A siloed data engineering team to cross-functional domain-data teams
- A centralized data lake/warehouse to an ecosystem of data products



Credit to Zhamak Dehghani

Use cases for Data Mesh

Data mesh tries to solve four challenges with a centralized data lake/warehouse:

- Lack of ownership: who owns the data – the data source team or the infrastructure team?
- Lack of quality: the infrastructure team is responsible for quality but does not know the data well
- Organizational scaling: the central team becomes the bottleneck, such as with an enterprise data lake/warehouse
- Technical scaling: current big data solutions can’t keep up with additional data requirements

Concerns with Data Mesh

- **No standard definition of a data mesh**
- **Huge investment in organizational change and technical implementation**
- Performance of combining data from multiple domains
- Duplication of data for performance reasons
- Getting quality engineering people for each domain
- Inconsistent technical implementations for the domains
- Domains don’t want to wait for a data mesh
- Need incentives for each domain to counter extra work
- Self-serve approach of data requests could be challenging
- Duplication of data and ingestion platform
- Creation of data silos for domains not able to join data mesh
- Not seeing the big picture for combing data

Data Mesh: Centralized vs decentralized data architecture
Data Mesh: Centralized ownership vs decentralized ownership

Q & A

James Serra, Microsoft, Data & AI Solution Architect
Email me at: jamesserra3@gmail.com
Follow me at: @JamesSerra
Link to me at: www.linkedin.com/in/JamesSerra
Visit my blog at: JamesSerra.com

Data Fabric vs Data Mesh

If Data Fabric uses data virtualization, how is it different from Data Mesh:

- Usually only some of the data is virtualized, so still mostly centralized
- Not making data as a product (no contract with domains)
- Still have siloed data engineering team

Comparisons of Data Fabric and Data Mesh

Areas	Data Mesh	Data Fabric
Framework	Focus on data architecture	Focus on data architecture, semantic consumption, through the wide use of Ontologies
Governance	Multiple governance layers	Unified governance layer
Security	Data Products owning the domain data and applying security and governance applicable to the domain	Focuses on a comprehensive Unified Security model across the entire Data Ecosystem
Consistency	Complex mechanics to ensure consistency of data	Focused on enabling and ensuring trust by applying automatic consistency
Implementation	Is complex, even to start a small implementation due to the need of understanding and segregating domain data	By far simpler, due to the inherent use of Data Virtualization, meta data and knowledge graphs