



From Central BI to Federated Self-Service: A Platform-as-a-Product Reference Architecture for Data Mesh for a Multinational Manufacturer

Umashanker Bandari, Sundara Vadivel Thilagaraj

Data, Analytics & AI Solutions Architect, Chicago, Illinois, USA

Principal Software Engineer, Cary, North Carolina, USA

ABSTRACT: Many large enterprises still route analytics demand through a centralized business intelligence competency center (BICC) that owns ingestion, modeling and reporting. In multinational manufacturers with several ERP systems and plant-level databases, such centers have become delivery bottlenecks, with request backlogs measured in quarters. Data mesh proposes domain ownership, data as a product, a self-serve data infrastructure platform and federated computational governance, but it offers little implementation guidance for hybrid estates that combine public cloud with on-premise systems. This paper presents a reference architecture for a self-service data platform delivered as an internal platform-as-a-service and managed as a product. It is built on Microsoft Azure (ADLS Gen2, Azure Data Factory, Azure Synapse Analytics, Azure Purview) together with on-premise SQL Server and Power BI, and is drawn from a ten-quarter rollout across seven business domains of a global Tier-1 industrial manufacturer. We describe the layered architecture, a tiered use-case onboarding model with an explicit readiness score, workspace and capacity isolation with usage-based chargeback, a RACI split of governance duties between the central platform team and domain teams, and the product-management practices used to scale adoption. Using representative data from 64 onboarded use cases and two historical baselines, we compare the approach with a centralized warehouse and a pure data-lake model. Median time-to-onboard fell from 12.6 weeks under the BICC to 3.9 weeks at steady state, and the dataset reuse ratio rose to 0.42, at a moderate governance cost. We close with lessons for adopting data mesh incrementally rather than as a reorganization.

KEYWORDS: data mesh, self-service business intelligence, data platform, federated data governance, platform as a product, Azure Synapse Analytics, Power BI

I. INTRODUCTION

For two decades the dominant pattern for enterprise analytics has been a centrally governed data warehouse fed by extract-transform-load pipelines and served through reports built by a dedicated team, often organized as a business intelligence competency center (BICC) [1,2]. It produces consistent, reconciled numbers, but its weakness is throughput: every question needing a new source or subject area joins one queue, served by a few specialists who must learn each domain before modeling it.

In the case organization, a global Tier-1 industrial manufacturer, supply chain, quality and manufacturing teams relied on spreadsheets and plant-level databases because the BICC backlog could not absorb their requests. Median lead time for a new use case through the BICC was 12.6 weeks, and an earlier divisional data lake had produced loosely documented datasets that few teams trusted.

Data mesh, articulated by Dehghani in 2019 and refined in 2020 [3,4], reframes the problem as one of organizational scaling. It proposes four principles: domain-oriented ownership of analytical data, data as a product, a self-serve data infrastructure platform, and federated computational governance. The published material is deliberately technology-agnostic and leaves open questions that dominate real deployments: how to package infrastructure for domain teams with mixed skills, how to isolate workspaces without multiplying cost, which governance decisions stay central, and how to introduce all this where most operational data still lives on-premise. Self-service BI research likewise reports that tools alone do not create self-service [5,6].

This paper reports the design and a ten-quarter rollout of a federated self-service data platform that answers these questions for one hybrid Microsoft Azure and on-premise estate. The platform is offered to domains as an internal



platform-as-a-service and run as a product in the sense of Team Topologies [7], with a roadmap, service levels and users rather than a handover date. The contributions are:

- 1) A layered reference architecture for data mesh on a hybrid estate, mapped to Azure Data Lake Storage (ADLS) Gen2, Azure Data Factory (ADF), Azure Synapse Analytics, Azure Purview, on-premise SQL Server and Power BI.
- 2) A tiered use-case onboarding model with an explicit readiness score and decision rules that route each use case to self-serve, assisted or platform-built delivery.
- 3) A workspace and capacity isolation scheme with a usage-based chargeback formula, and a RACI split of governance duties between the central platform team, a data council and domain teams.
- 4) An evaluation on representative data from 64 use cases against centralized-warehouse and pure-lake baselines, with lessons for incremental adoption.

Sections 2 and 3 cover related work and the case context, Section 4 the architecture, Section 5 the evaluation, Section 6 lessons and limitations, and Section 7 concludes.

II. BACKGROUND AND RELATED WORK

A. Centralized warehousing and BI competency centers

The enterprise data warehouse tradition offers two canonical designs. Inmon's corporate information factory builds a normalized, subject-oriented and integrated warehouse from which departmental marts are derived [8]. Kimball's bus architecture builds dimensional marts that share conformed dimensions, so that facts from different business processes can be combined consistently [9]. Data Vault 2.0 adds an insert-only hub-link-satellite layer for auditable integration of many sources [10]. All three assume a central team that owns integration logic and the semantic model. That assumption enables conformance but makes the central team, institutionalized as the BICC [1], a serial resource.

B. Data lakes and the lakehouse

Dixon introduced the data lake as a store of raw data in its natural state, from which many consumers can draw, in contrast to the pre-shaped data mart [11]. Lakes lowered ingestion cost, but surveys documented missing metadata, unclear ownership and weak quality control, the so-called data swamp [12,13]. Delta Lake adds ACID transactions, schema enforcement and time travel over Parquet files in cloud object storage [14], and the lakehouse proposal argues that open formats plus such metadata layers can support warehouse workloads and advanced analytics on one copy of the data [15]. These advances narrow the technical gap between lake and warehouse but do not assign ownership, the organizational question data mesh raises. Catalogs such as Google's Goods show that post hoc metadata harvesting aids discovery, but that provenance and semantics are hard to recover after the fact [16].

C. Self-service business intelligence

Imhoff and White define self-service BI as facilities that make business users less dependent on IT, through easy data access, easy tools and fast deployment [5]. Alpar and Schulz distinguish levels of self-service, from using prepared reports to creating new information products, and note that higher levels demand both data literacy and governed data foundations [6]. Power BI adoption guidance from 2021 names ownership models from business-led self-service to enterprise-managed delivery [17]. The literature agrees that self-service without a governed, reusable data layer multiplies unreconciled datasets.

D. Data mesh, platform teams and federated governance

Dehghani positions data mesh as a response to the failure modes of both monolithic warehouse and lake: centralized ownership, pipeline-oriented decomposition and hyper-specialized teams [3]. The 2020 article adds a logical architecture of data product quanta, a multi-plane platform and computational policies embedded in each product [4]. It borrows bounded contexts from domain-driven design [18] and echoes Conway's observation that systems mirror communication structures [19]. Skelton and Pais supply our team vocabulary: stream-aligned teams deliver value, a platform team reduces their cognitive load, and enabling teams coach [7]. Weber, Otto and Österle show that the right assignment of governance decision rights depends on context [20], and Khatri and Brown decompose governance into separately allocable decision domains [21]. DAMA-DMBOK provides the function catalog from which such allocations are made [22]. Microsoft's 2021 enterprise-scale analytics guidance describes data landing zones that map loosely to mesh concepts [23]. Missing is an end-to-end account of combining these pieces in a hybrid manufacturing estate, with measured effects.



III. CASE CONTEXT AND DESIGN REQUIREMENTS

The case organization operates dozens of plants across North America, Europe and Asia, with a landscape shaped by acquisitions: SAP ECC with a planned S/4HANA migration at the core, JD Edwards and Infor M3 in acquired business units, several smaller ERPs, and manufacturing execution and quality systems backed by plant-level SQL Server databases. The enterprise warehouse ran on on-premise SQL Server with SSIS pipelines and a Kimball-style dimensional model for finance and sales. Reporting used Power BI on a shared Premium capacity.

Before the program, the eleven-person BICC handled intake, modeling, pipelines and reports. Four quarters of its ticket history showed 23 completed new-use-case requests with a median lead time of 12.6 weeks, roughly half of its queue time. A separate divisional data lake (2017 to 2019) had hosted 17 use cases. It ingested quickly but lacked ownership metadata, and most consumers rebuilt their own cleansing logic.

From these observations and interviews with domain leads we derived six requirements:

- **R1 Domain ownership with a safety net.** Domains own their data products; teams with limited engineering skill can deliver with assistance.
- **R2 Hybrid ingestion.** On-premise sources are reachable without inbound firewall ports or per-domain integration infrastructure.
- **R3 Isolation without sprawl.** Each domain gets template-provisioned storage, compute and reporting workspaces.
- **R4 Enterprise consistency where it matters.** Group finance, master data and regulated data stay under enterprise standards.
- **R5 Discoverability and trust.** Every published dataset has an owner, documentation, a freshness objective and visible lineage.
- **R6 Measurable adoption.** The platform exposes its own usage, cost and lead-time metrics.

IV. REFERENCE ARCHITECTURE AND OPERATING MODEL

A. Layered architecture

Fig. 1 shows the four layers of the architecture and the governance plane that spans them. The **hybrid source estate** at the bottom is reached only through ADF self-hosted integration runtimes installed in the organization's data centers and at larger plants; they make outbound connections to Azure over a private network link, which satisfies R2 without inbound firewall changes.

The **central platform services** layer is what the platform team builds and operates. Ingestion is metadata-driven: a control table in an Azure SQL database lists each source object with its connection, extraction mode (full, watermark or change tracking), schedule and target path, and a small set of generic ADF pipelines iterate over it. Raw data lands in an immutable ADLS Gen2 zone partitioned by source, object and load date. Shared Synapse resources comprise a serverless SQL pool for exploration and a dedicated SQL pool that continues to host the conformed finance model migrated from the on-premise warehouse. Infrastructure-as-code templates in Azure DevOps pipelines create domain resources, identity follows a strict Azure Active Directory group model, and a monitoring stack collects pipeline, query, storage and Power BI activity metrics.

Each **domain workspace** is a template-provisioned bundle: a resource group, an ADLS Gen2 container with curated and product zones stored as Delta tables [14], a Synapse database schema and an optional Spark pool, and three Power BI workspaces for development, test and production connected by deployment pipelines. A domain team controls everything inside its workspace and reads only what other domains publish. This keeps the lakehouse property of a single copy of data per product [15] while the ownership boundary follows the domain, as data mesh prescribes [4].

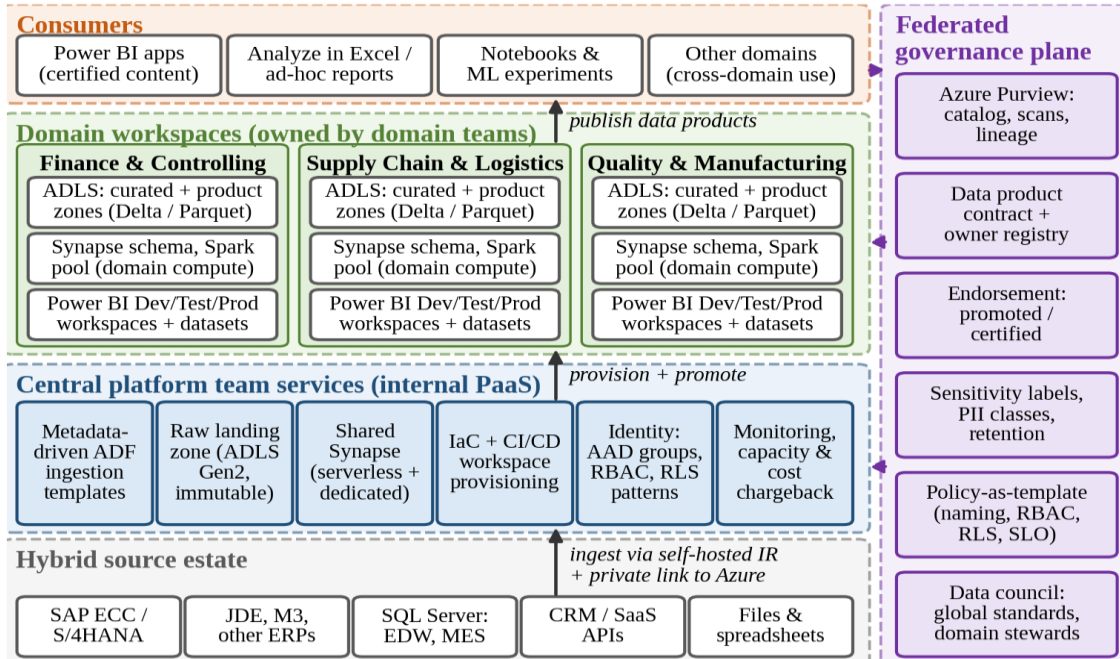


Fig.1. Reference architecture of the federated self-service platform: central platform services (internal PaaS), template-provisioned domain workspaces, a federated governance plane that applies policies to every layer, and consumers of published data products.

The **federated governance plane** holds the elements that must be uniform: Azure Purview for scanning, classification and lineage, the product contract registry, Power BI endorsement (promoted and certified), sensitivity labels, and policy templates for naming, role-based access control (RBAC), row-level security (RLS) and service-level objectives (SLOs). Governance is computational where possible: naming, tags and contract fields are validated in the deployment pipeline, so a non-compliant product cannot reach production.

B. Data products and their contract

A data product, the unit of ownership and publication, consists of one or more Delta tables in a product zone, a Synapse view layer that fixes the public schema, and usually a Power BI dataset that adds measures and RLS. Its contract fields (Table 1) operationalize the data-as-a-product qualities of being discoverable, addressable, trustworthy and self-describing [3], overlapping the FAIR principles [24]. The freshness target follows the SLO discipline of site reliability engineering [25]: each product declares an objective, for example 95% of daily loads complete by 06:00 local plant time, and the monitoring stack reports attainment.

Table.1. Data product contract: mandatory metadata fields registered for every certified dataset.

Field group	Fields	Enforcement
Identity	product_id, domain, name, version (semantic), description, bounded context	Naming policy checked in CI pipeline
Ownership	product owner, technical owner, data steward, support channel	AAD group must exist; owner sign-off at certification
Interface	output ports (Delta path, Synapse view, Power BI dataset), schema, grain, keys, upstream lineage	Schema file in repository; breaking change requires new major version
Quality and SLO	freshness target, completeness threshold, reconciliation rule, last check result	Checks run after each load; status surfaced in catalog
Security	sensitivity label, PII columns, RLS role model, allowed consumer groups	Label and RLS reviewed by steward; scanned by Purview



Two modeling rules keep federation from fragmenting shared semantics. First, enterprise-wide dimensions such as legal entity, material, customer and calendar are themselves data products, owned by a master data domain and consumed by reference; domains may extend them but may not fork them, applying Kimball's conformed dimensions [9] across ownership boundaries. Second, a product that joins data from two domains belongs to the domain whose business question it answers, and it declares both upstream products in its lineage.

C. Use-case onboarding model

Onboarding is where a BICC accumulates queue time. Fig. 2 shows the five-stage lifecycle. A domain submits a one-page use-case canvas describing business value, the accountable owner, required sources, target users and data sensitivity. The platform team then scores readiness and routes the use case to one of three delivery tiers. The score combines three factors, each normalized to the range 0 to 1:

$$S = 0.40 \cdot Q + 0.35 \cdot I + 0.25 \cdot (1 - X) \quad (1)$$

where Q is domain team competency from a short skills checklist; I is ingestion coverage, the share of required source objects already present in the landing zone; and X is cross-domain coupling, the share of required entities owned by other domains and not yet published. Use cases with $S \geq 0.70$ and no restricted-data flag enter **Tier A** (self-serve): the domain builds everything with office-hour support. Use cases with $0.40 \leq S < 0.70$ enter **Tier B** (assisted): a platform engineer is embedded for two to four weeks, in the enabling-team mode of Team Topologies [7]. Use cases with $S < 0.40$, or those that serve group-level reporting, enter **Tier C** and are built by the platform team with the domain as product owner. The council calibrated the weights after two quarters against observed effort; ingestion coverage proved the strongest predictor.

Provisioning is fully automated. Canvas approval triggers a pipeline that creates missing domain resources and registers a draft product contract. Certification is a checklist review by the domain steward covering contract completeness, reconciliation against a reference figure, RLS test results and sensitivity labeling. A refused certification returns the product to the build stage with written findings, which happened for about one in five first submissions. Certified products are published to the production workspace and registered in the catalog, and they are recertified quarterly or on any breaking change.

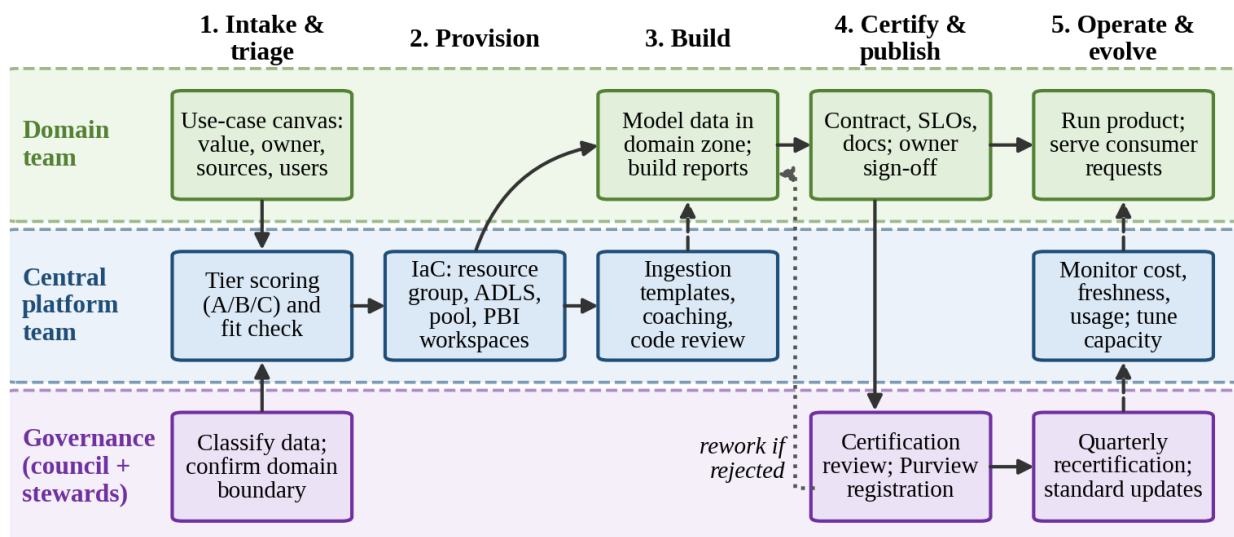


Fig.2. Use-case onboarding lifecycle as a swimlane across domain team, central platform team and governance roles. Dashed arrows denote support and monitoring flows; the dotted arrow is the rework loop when certification is refused.

D. Workspace and capacity isolation

Full isolation of every resource would multiply fixed cost, so we isolate selectively (R3). Storage is isolated per domain container with access control lists bound to AAD groups following a fixed pattern (domain, zone, role); cross-domain read access is granted only to product zones. Serverless SQL compute is shared, with logical isolation only. Spark pools are created per domain only when a domain runs regular engineering workloads, because idle dedicated compute was



the largest avoidable cost in early quarters. Certified production workspaces run on shared Premium capacity, with a second capacity reserved for group finance at month-end close; development workspaces use shared (Pro) capacity. Cost is allocated monthly to each domain d with a simple, auditable formula:

$$K_d = F + \sum_r p_r \cdot u_{\{d,r\}} + C_{cap} \cdot (v_d / \sum_j v_j) \quad (2)$$

where F is a flat platform fee per active domain covering the central team's shared services, $u_{\{d,r\}}$ is the metered usage of resource r by domain d (storage terabytes, serverless terabytes scanned, Spark vCore-hours, pipeline activity runs) priced at p_r , C_{cap} is the monthly cost of shared Power BI capacity, and v_d is the domain's share of capacity CPU time from the capacity metrics.

E. Splitting governance duties

The governance split applies the decision-domain view of Khatri and Brown [21] and the contingency argument of Weber et al. [20]: decisions with enterprise-wide externalities stay central, and decisions whose effects are local move to domains. Table 2 shows the resulting RACI. The data council, with representatives of each domain, is accountable for global standards, enterprise KPIs and cross-domain conflicts; each domain appoints a steward accountable for classification, access approval and certification. The key design choice was to place certification with the domain steward rather than with a central reviewer. This removed the last central queue, at the cost of variable review rigor, which the council checks through quarterly sampling audits.

Table.2. Governance RACI split (R = responsible, A = accountable, C = consulted, I = informed)

Governance activity	Data council	Platform team	Domain team	Domain steward
Global standards (naming, classification scheme, SLO tiers)	A	R	C	C
Enterprise master data and group KPI definitions	A	C	C	R
Domain data model and transformation logic	I	C	A/R	C
Sensitivity labeling and PII tagging of datasets	I	C	R	A
Access approval to domain data products	I	I	R	A
Dataset certification	I	C	R	A
Quality monitoring and incident resolution	I	C (tooling)	A/R	C
Platform security, identity, capacity and cost allocation	I	A/R	C	I
Cross-domain conflict resolution	A/R	C	C	C

F. Managing the platform as a product

The platform team treats domain teams as customers and the platform as a product with a roadmap, in the sense described by Cagan for product organizations [26] and by Skelton and Pais for internal platforms [7]. Five practices carried most of the effect. First, **golden paths**: documented, template-backed routes for the three most common use-case shapes, each with an example repository. Second, a **thinnest viable platform**: features were added only when two domains needed them, keeping the catalog operable by a team of seven. Third, **measured feedback**: a quarterly user survey plus the metrics of R6, reviewed openly. Fourth, a **champions network** of one or two builders per domain who answered most first-line questions. Fifth, **delivery metrics** from software delivery research [27]: lead time per use case and change failure rate per product.



V. EVALUATION

A. Evaluation setup

We compare three delivery models in the same organization: (i) the centralized warehouse with the BICC (23 use cases from the four pre-rollout quarters); (ii) the divisional data lake (17 use cases, 2017 to 2019); and (iii) the federated platform (all 64 use cases from ten rollout quarters, seven domains), with steady state taken over Q8 to Q10.

Evaluation setup and data disclosure: All quantitative values in this section are representative figures derived from a small evaluation of the three delivery models described above (23, 17 and 64 use cases respectively) and from practitioner observation during the rollout. Lead times were reconstructed from ticket and backlog timestamps; dataset bindings from the Power BI activity log and workspace inventory; governance effort from time-booking codes sampled over two quarters per model; and incidents from the data-quality incident log. Values were rounded, normalized and anonymized to protect confidential organizational data, and baselines were measured in different periods with different staff. The figures are therefore indicative of direction and magnitude, not statistically generalizable estimates.

B. Metrics

We use five metrics. **Time-to-onboard** T is the elapsed time in weeks from acceptance of a use case at intake to its first production release used by at least one named business user. **Dataset reuse** is measured per quarter q as the reuse ratio

$$R_q = B_q^{\text{shared}} / B_q^{\text{total}} \quad (3)$$

where B_q^{total} counts all active bindings between consuming artifacts (reports, dataflows, composite models, external queries) and datasets in quarter q , and B_q^{shared} counts bindings whose consumer lives in a different workspace from the dataset's owning workspace. **Governance overhead** G is the number of central hours spent on reviews, certifications, access requests and standards work per active use case per month:

$$G = H_{\text{central}} / (N_{\text{active}} \cdot M) \quad (4)$$

where H_{central} is central hours booked over M months and N_{active} the average number of active use cases; domain-steward hours are excluded because G measures load on the shared bottleneck. **Data trust** is proxied by data-quality incidents per 100 published datasets per quarter, and **domain autonomy** by the share of change requests to existing use cases completed without a ticket to a central team.

C. Results

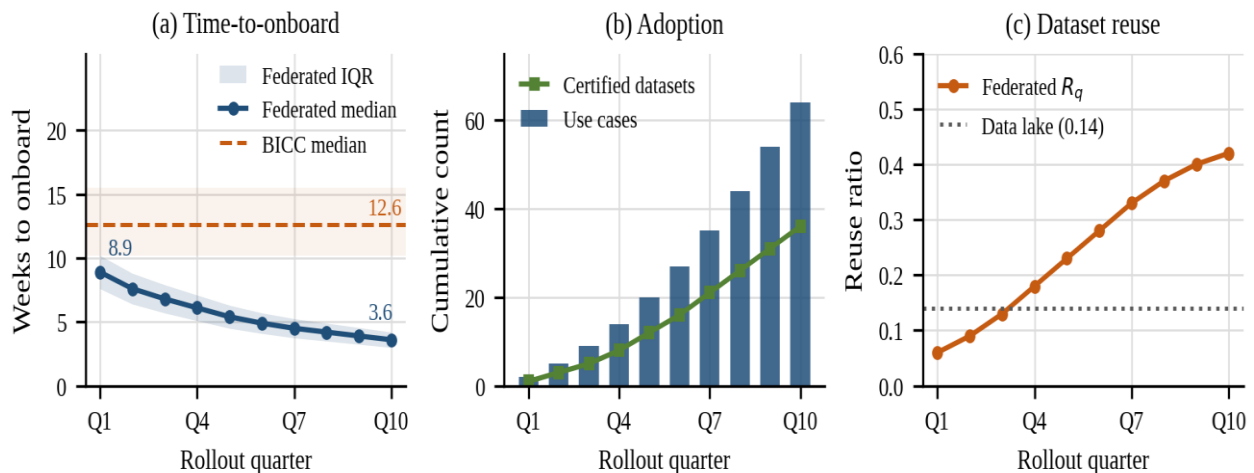


Fig.3. Rollout outcomes per quarter: (a) median time-to-onboard, federated vs. BICC baseline (bands: interquartile ranges); (b) cumulative use cases and certified datasets; (c) reuse ratio R_q from (3) vs. the data lake baseline; $n = 64$ federated, 23 BICC use cases (representative data).



Fig. 3(a) plots median time-to-onboard per rollout quarter. In Q1 the federated median was 8.9 weeks, with wide dispersion because most early use cases fell into Tier C. The median fell to 5.4 weeks by Q5 and to 3.6 weeks in Q10, with a steady-state value of 3.9 weeks over Q8 to Q10. The reduction came from removal of queue time, growing ingestion coverage and template provisioning, which cut environment setup from about two weeks to under a day. The interquartile range narrowed from 2.6 to 1.2 weeks, making delivery dates predictable.

Fig. 3(b) shows adoption. Onboarding accelerated from two use cases in Q1 to ten per quarter in Q9 and Q10, reaching 64 in total, while certified datasets grew to 36. By Q10 the tier mix had shifted to 45% Tier A, 38% Tier B and 17% Tier C, compared with 60% Tier C in the first two quarters. Fig. 3(c) shows the reuse ratio. It started below the divisional lake's 0.14, because the first use cases had nothing to reuse, crossed it in Q4 and reached 0.42 in Q10. The inflection followed publication of shared master data products (material, customer, legal entity, calendar) in Q3 and Q4.

D. Comparison of delivery models

Table 3 compares the three models on all five metrics, and Fig. 4 shows the same values normalized so that the best model on each metric scores 1, using $x_{\min}/x$ for metrics where lower is better and $x/x_{\max}$ otherwise. No model dominates. The centralized warehouse has the highest reuse (0.52), since nearly all reports bind to a few central models, and the fewest incidents (2.3), but the longest lead time and the highest central load (11.8 hours). The pure lake has the lowest overhead (3.4 hours) because little governance was performed, and pays with the lowest reuse (0.14) and most incidents (8.6). The federated platform sits between the two on reuse, trust and overhead and leads on speed and autonomy: 74% of change requests to existing use cases were completed by domains without a central ticket.

Table.3. Comparison of delivery models (representative values; federated values are steady state, Q8 to Q10).

Metric	Centralized warehouse (BICC)	Pure data lake	Federated self-service
Median time-to-onboard (weeks)	12.6	6.8	3.9
Dataset reuse ratio R	0.52	0.14	0.42
Central governance hours per active use case per month, G	11.8	3.4	5.1
Quality incidents per 100 datasets per quarter	2.3	8.6	3.4
Change requests completed without central ticket	12%	61%	74%

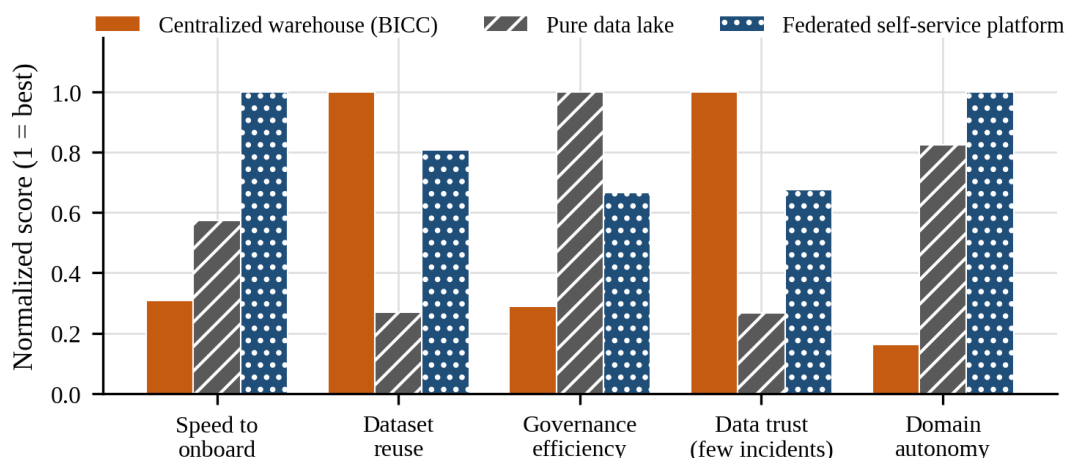


Fig.4. Normalized comparison of the three delivery models on the five metrics of Table 3; 1 denotes the best observed value on each metric (representative data).



Two aspects of the federated result need qualification. First, G excludes domain-steward effort, which averaged about 2.6 hours per active use case per month; including it gives 7.7 hours, still about a third lower than the BICC. Governance work was redistributed to domain experts, not eliminated. Second, the incident rate of 3.4 is higher than the BICC's 2.3. Most of the difference traced to Tier A products in their first two months, before reconciliation rules matured, which suggests that the certification checklist should require a longer reconciliation history for self-built products.

VI. DISCUSSION

A. Lessons learned

Start with the platform, not the reorganization. The rollout began not by declaring domains data owners but by making ownership cheap through templates, landed sources and golden paths; domains accepted ownership once exercising it was easy.

Tier the model instead of forcing pure self-service. Data mesh descriptions can read as if every domain will staff its own data engineers. The three-tier model let teams without engineers own products while borrowing engineering capacity, and the readiness score in (1) made routing decisions transparent rather than political.

Publish shared dimensions early. The reuse curve in Fig. 3(c) bent upward only after master data products appeared. They should have been built first, since they are the join keys on which reuse depends. Conformed dimensions remain as important in a mesh as in a Kimball bus [9]; what changes is who owns them.

Make cost visible. After domains saw their capacity share under (2), hourly refreshes serving daily decisions were rescheduled without central intervention.

Treat the on-premise estate as first-class. Most delay in early quarters came not from cloud services but from the hybrid edge: self-hosted integration runtimes sized for a pilot, plant networks with limited bandwidth, and source tables without change tracking. We moved runtimes to a high-availability pair per region, scheduled heavy extracts outside plant shift changes, and made enabling SQL Server change tracking a readiness item in the canvas. A mesh on a hybrid estate is only as fast as its slowest source connection.

Keep the finance core central. Group financial reporting remained a Tier C enterprise product on the dedicated SQL pool. A mesh can contain a centrally owned domain; for consolidated finance, the team closest to the data is group finance.

B. Threats to validity and limitations

The evaluation is a single-organization case with baselines measured in different periods, so the comparison is confounded by time. Tooling matured, staff gained experience and management attention was high during the rollout, each a possible independent cause of improvement. Use-case mix also differs: the BICC sample holds more finance requests, which are harder to reconcile. Samples of 17 to 64 use cases do not support significance testing. Lead times depend on ticket hygiene, and the reuse ratio counts bindings rather than query volume. Finally, the author participated in the design; we mitigated interpretation bias by preferring system-generated measures and reporting metrics on which the federated model did not lead.

C. Implications for incremental adoption

For organizations with a BICC and a hybrid estate, we suggest a staged path that keeps the center's strengths while removing it from the critical path:

- 1) Stand up the platform team and a minimal platform: metadata-driven hybrid ingestion, a landing zone, and template provisioning for one domain workspace.
- 2) Publish shared master data and calendar products under central ownership, with contracts, before onboarding domain use cases.
- 3) Onboard two or three pilot domains with Tier B support, and use them to build the first golden paths and to calibrate the readiness score.
- 4) Move certification to domain stewards once the council has agreed the checklist, and introduce sampling audits at the same time.
- 5) Convert the BICC progressively into the platform team and an enabling team, keeping enterprise finance and group reporting as centrally owned domains.



This treats data mesh as a direction for incremental change rather than a one-step reorganization, consistent with the evolutionary stance of the original articles [4].

VII. CONCLUSION AND FUTURE WORK

This paper presented a reference architecture and operating model for federated self-service analytics on a hybrid Azure and on-premise estate, and reported a ten-quarter rollout in a global Tier-1 industrial manufacturer. On representative data from 64 use cases, median time-to-onboard fell from 12.6 weeks under a central BICC to 3.9 weeks, and the dataset reuse ratio reached 0.42, while central governance effort per use case fell by more than half. The results suggest that data mesh principles can be adopted incrementally, starting from the platform and the shared data layer, while keeping central ownership where it remains appropriate.

Future work will proceed in three directions. First, we plan to extend computational governance from deployment-time validation to runtime policy enforcement. Second, the planned S/4HANA migration will test whether product contracts insulate consumers from a major source-system change. Third, we intend to add machine-learning feature sets as a product type and test whether the readiness score transfers to data science use cases.

REFERENCES

- [1] G. J. Miller, D. Bräutigam, and S. V. Gerlach, *Business Intelligence Competency Centers: A Team Approach to Maximizing Competitive Advantage*. Hoboken, NJ, USA: Wiley, 2006.
- [2] H. Chen, R. H. L. Chiang, and V. C. Storey, "Business intelligence and analytics: From big data to big impact," *MIS Quarterly*, vol. 36, no. 4, pp. 1165–1188, 2012.
- [3] Z. Dehghani, "How to move beyond a monolithic data lake to a distributed data mesh," *martinfowler.com*, May 2019.
- [4] Z. Dehghani, "Data mesh principles and logical architecture," *martinfowler.com*, Dec. 2020.
- [5] C. Imhoff and C. White, "Self-service business intelligence: Empowering users to generate insights," *TDWI Best Practices Report*, The Data Warehousing Institute, 2011.
- [6] P. Alpar and M. Schulz, "Self-service business intelligence," *Business & Information Systems Engineering*, vol. 58, no. 2, pp. 151–155, 2016.
- [7] M. Skelton and M. Pais, *Team Topologies: Organizing Business and Technology Teams for Fast Flow*. Portland, OR, USA: IT Revolution Press, 2019.
- [8] W. H. Inmon, *Building the Data Warehouse*, 4th ed. Indianapolis, IN, USA: Wiley, 2005.
- [9] R. Kimball and M. Ross, *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*, 3rd ed. Indianapolis, IN, USA: Wiley, 2013.
- [10] D. Linstedt and M. Olschmke, *Building a Scalable Data Warehouse with Data Vault 2.0*. Waltham, MA, USA: Morgan Kaufmann, 2015.
- [11] J. Dixon, "Pentaho, Hadoop, and data lakes," *James Dixon's Blog*, Oct. 2010.
- [12] H. Fang, "Managing data lakes in big data era: What's a data lake and why has it become popular in data management ecosystem," in *Proc. IEEE Int. Conf. Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2015, pp. 820–824.
- [13] C. Giebler, C. Gröger, E. Hoos, H. Schwarz, and B. Mitschang, "Leveraging the data lake: Current state and challenges," in *Proc. Int. Conf. Big Data Analytics and Knowledge Discovery (DaWaK)*, LNCS vol. 11708, Springer, 2019, pp. 179–188.
- [14] M. Armbrust et al., "Delta Lake: High-performance ACID table storage over cloud object stores," *Proc. VLDB Endowment*, vol. 13, no. 12, pp. 3411–3424, 2020.
- [15] M. Armbrust, A. Ghodsi, R. Xin, and M. Zaharia, "Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics," in *Proc. Conf. Innovative Data Systems Research (CIDR)*, 2021.
- [16] A. Halevy, F. Korn, N. F. Noy, C. Olston, N. Polyzotis, S. Roy, and S. E. Whang, "Goods: Organizing Google's datasets," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 2016, pp. 795–806.
- [17] Microsoft, "Power BI adoption roadmap," *Microsoft Power BI documentation*, 2021.
- [18] E. Evans, *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Boston, MA, USA: Addison-Wesley, 2003.
- [19] M. E. Conway, "How do committees invent?" *Datamation*, vol. 14, no. 4, pp. 28–31, 1968.
- [20] K. Weber, B. Otto, and H. Österle, "One size does not fit all—A contingency approach to data governance," *ACM Journal of Data and Information Quality*, vol. 1, no. 1, Art. 4, 2009.



- [21] V. Khatri and C. V. Brown, "Designing data governance," Communications of the ACM, vol. 53, no. 1, pp. 148–152, 2010.
- [22] DAMA International, DAMA-DMBOK: Data Management Body of Knowledge, 2nd ed. Basking Ridge, NJ, USA: Technics Publications, 2017.
- [23] Microsoft, "Enterprise-scale for analytics and AI," Microsoft Cloud Adoption Framework for Azure documentation, 2021.
- [24] M. D. Wilkinson et al., "The FAIR guiding principles for scientific data management and stewardship," Scientific Data, vol. 3, Art. 160018, 2016.
- [25] B. Beyer, C. Jones, J. Petoff, and N. R. Murphy, Eds., Site Reliability Engineering: How Google Runs Production Systems. Sebastopol, CA, USA: O'Reilly Media, 2016.
- [26] M. Cagan, Inspired: How to Create Tech Products Customers Love, 2nd ed. Hoboken, NJ, USA: Wiley, 2017.
- [27] N. Forsgren, J. Humble, and G. Kim, Accelerate: The Science of Lean Software and DevOps. Portland, OR, USA: IT Revolution Press, 2018.