

Anas Siddiqui

WHEN DOES DECENTRALIZATION PAY OFF?

An Empirical and Simulation Study
of Data Mesh Operating Models

Cross-domain data dependencies mined from public infrastructure manifests, a queueing model with proofs, and a verified discrete-event simulation of central, mesh, “theater” and hub-and-spoke operating models.

ORIGINAL PROPOSAL SUPERVISED BY

Prof. Guenther Ruhe
Prof. Tim Reimer

AFFILIATION

University of Calgary
Alberta, Canada

CODE, DATA AND SIMULATIONS

 github.com/big-oh/data-mesh-decentralization

Mining pipeline, discrete-event simulator, experiment scripts, figures
and LaTeX source. MIT License.



When Does Decentralization Pay Off? An Empirical and Simulation Study of Data Mesh Operating Models

Anas Siddiqui
University of Calgary
Calgary, Alberta, Canada

Abstract—Whether a large organization should adopt data mesh hinges on how data dependencies between its domains are structured and grow, yet this structure has not been measured and existing evidence on data mesh is almost entirely qualitative. We mine 566 public infrastructure manifests of Norway’s labour and welfare administration (NAV) to reconstruct its declared cross-team data-dependency graph and five years of its growth. The graph is sparse (3.7% of team pairs) and hub-concentrated (degree Gini 0.53 versus 0.24 in matched random graphs); edges grew quadratically at a stable low density ($\alpha = 2.03$, 95% CI [1.99, 2.31]), contradicting our hypothesis of sub-quadratic growth, while catalogued data products grew sub-quadratically ($\alpha = 1.37$). We then prove that a central data team’s capacity is bounded under communication overhead regardless of headcount, that mesh stability is governed by the most connected domains, and that relabeling teams as domains without transferring ownership can never beat centralization. A verified discrete-event simulation of central, mesh, “theater” and hub-and-spoke operating models at equal headcount, pre-registered before any comparative output, shows that the all-pairs dependency assumption penalizes the central model rather than the mesh, and that the mesh’s delivery advantage depends chiefly on diseconomies of scale in central teams: without them, the two models deliver within 8% of each other. Cross-domain demand, dependency structure and reuse matter far less. Data, code and pre-registration are public.

Index Terms—data mesh, data architecture, socio-technical systems, mining software repositories, discrete-event simulation, queuing, sensitivity analysis, Conway’s law

I. INTRODUCTION

Data mesh proposes to replace the central data team and its monolithic lake or warehouse with domain teams that publish analytical data as products, supported by a self-serve platform and federated, automated governance [1]–[3]. Its central claim is organizational: a single team that must understand every source and serve every consumer becomes a bottleneck as sources and use cases multiply, whereas teams that already hold the domain knowledge can scale out [1], [4]. The counter-claim is also organizational: decentralization multiplies the interfaces between teams and invites duplicated effort, inconsistent semantics and integration debt [4], [5].

Seven years after the idea was introduced, organizations still decide whether to adopt it largely on the basis of practitioner heuristics. The academic literature has matured, but it remains predominantly qualitative: gray-literature reviews [4], [6], interview studies [7], [8], case studies [9], [10] and

catalogues of design decisions [11], [12]. Several of these explicitly call for quantitative work [7], [8], and the most comprehensive review lists the lack of empirically validated methods for weighing the costs and benefits of a migration as an open challenge [4]. Practitioner retrospectives meanwhile report a recurring anti-pattern in which existing IT teams are relabeled as “domains” without any transfer of ownership or authority [13], and interviews with ten large incumbents find that data ownership has rarely moved even where mesh principles were adopted [8].

Both sides of the argument turn on the same structural quantity: *how data dependencies between domains are distributed and how they grow as domains are added*. If every domain eventually depends on every other, coordination costs grow with the number of pairs, $N(N - 1)$, a scaling familiar from Brooks’ analysis of team communication [14] and assumed by the only formal economic model of data mesh we are aware of [5]. If dependencies are sparse and concentrated in a few hub domains, the arithmetic changes. To our knowledge, this structure has not been measured in a real organization, and no study has quantified what it implies for the delivery performance of centralized and decentralized operating models.

This paper revisits a question first posed in an earlier research proposal by the author—*is data mesh suitable for large organizations, and which factors determine success?*—and makes it falsifiable. We measure the cross-domain dependency structure of a large public organization that has adopted data mesh principles, use it to ground an analytical queueing model and a discrete-event simulation of four operating models (a central team, a federated mesh, a “theater” mesh that relabels without decentralizing, and a hub-and-spoke hybrid), and identify the factors that decide between them with parameter sweeps. Hypotheses, parameters and the analysis plan were pre-registered in the public repository before any comparative simulation output existed; deviations are reported explicitly.

Contributions.

- 1) *Empirical*: the first measurement, to our knowledge, of a real organization’s cross-domain data-dependency graph and its growth, mined from 566 public infrastructure manifests in 129 repositories of the Norwegian Labour and Welfare Administration (NAV), with a test of the all-pairs assumption.

- 2) *Analytical*: stability results showing that a central team’s capacity is bounded under communication overhead regardless of headcount, that mesh stability is governed by the most connected domain rather than by the number of domains, and that relabeling without decentralization can never outperform the central model.
- 3) *Simulation*: an open, verified discrete-event model of four operating models, with decision maps that show which implementation factors decide the comparison.
- 4) *Artifacts*: data, code, tests, pre-registration and all results in a public repository.

Section II reviews related work, Section III states the research questions, Section IV the methodology, Section V the formal model, Sections VI–VII the simulation design and implementation, Section VIII the results, and Sections IX–XI discussion, threats to validity and conclusions.

II. BACKGROUND AND RELATED WORK

A. Data mesh

Dehghani introduced data mesh as a response to three failure modes of centralized analytical platforms: the proliferation of sources and consumers that a single platform cannot absorb, pipelines decomposed by technical stage so that one feature touches every stage, and hyper-specialized central engineers who are remote from both the source systems and the consumers [1]. The proposal rests on four principles—domain ownership, data as a product, a self-serve data platform and federated computational governance [2], [3]—and early academic treatments positioned it as a paradigm shift relative to warehouses and lakes [15]. Practitioner guidance has since shifted toward data contracts and product standards [16], [17], and a 2026 retrospective by consultants who implemented many meshes reports that hybrid platforms, enabling centers of excellence and policy-as-code are the patterns that work, while relabeling existing teams as domains is the most common failure [13].

B. Empirical studies of adoption

A systematic review of 114 gray-literature sources characterized the principles, components, benefits and drawbacks of data mesh and derived a research agenda that includes methods for assessing migration costs and benefits [4]. Interviews with 15 practitioners identified the transition to federated governance, the shift of responsibility for data products and understanding of the concept as the main difficulties [7]. A study of ten large Swiss incumbents found that all adopted mesh principles in some form but that governance was becoming more, not less, centralized, and ownership largely stayed where it was [8]. A case study at NAV, the organization we study, describes its move from a central warehouse toward distributed data management [9], and a longitudinal study examines the tensions of becoming data-driven in a large public-sector organization [10]. Further work contributes readiness approaches [18], design-decision catalogues for self-serve platforms [11] and for federated governance [12], case experience [19], a review of research gaps [20], and delimitations of

data mesh from data fabrics, spaces and platforms [21]–[23]. All of this evidence is qualitative.

C. Governance mechanisms

Several prototypes automate parts of federated governance: contract-based verification of data declassification [24], rule-based enforcement of access policies derived from legal text [25], scalable placement of policy-as-code decision points for data products [26], and LLM-assisted evaluation of access requests [27], now also exposed to AI agents [28]. A gray-literature review consolidates the practice of data contracts [6]. These works evaluate mechanisms, not the organizational trade-off between operating models.

D. Formal and quantitative models

Quantitative treatments are scarce. A position paper contrasts centralized lakes with meshes conceptually [29]; architectural frameworks compare requirements across architecture families [23]; and a recent proposal argues for AI-augmented hub-and-spoke lakehouses without reporting outcome data [30]. The closest work is a game-theoretic model in which N domains under-invest in the generality of their data products because the benefits accrue to others [5]. Its welfare loss and technical debt grow with the number of domain pairs, $N(N - 1)$; its calibration is illustrative and its author identifies empirical validation as future work. We complement it by *measuring* the dependency structure it assumes and by modeling delivery dynamics (queues, coordination latency, change propagation) rather than investment equilibria.

E. Coordination theory and network growth

Conway observed that system structure mirrors organizational communication structure [31]; the “mirroring” hypothesis has since been tested quantitatively [32], coordination has been framed as the management of dependencies [33], and socio-technical congruence links unmet coordination needs to lower productivity [34]. Brooks argued that communication paths grow as $n(n - 1)/2$ and erode the productivity of large teams [14]—the mechanism behind the capacity model in Section V. For network growth, real graphs often *densify*, with edges growing as N^a for $1 < a < 2$ [35], and heavy-tailed degree distributions (hubs) are common [36], though claims of power laws require careful testing [37].

III. PROBLEM STATEMENT AND RESEARCH QUESTIONS

Problem. Large organizations deciding whether to adopt data mesh lack a quantitative basis for the decision. The decision hinges on how cross-domain data dependencies are structured and how they grow—a quantity that both advocates and critics invoke but that has not been measured—and on how that structure interacts with team capacity, coordination latency and governance to determine delivery performance under centralized and decentralized operating models.

Main question. *Under empirically grounded cross-domain dependency structures, under which organizational conditions does a federated data mesh outperform a centralized data*

team in delivery lead time, and which conditions determine the outcome? We answer it through three sub-questions:

RQ1 How are declared cross-domain data dependencies structured in a large organization practicing data mesh, and how does their number scale with the number of participating domains?

H1a The dependency graph is sparse and hub-concentrated, relative to a uniform random graph with the same numbers of nodes and edges.

H1b Edges grow sub-quadratically, $E \propto N^a$ with $1 < a < 2$, rather than as $N(N-1)$.

RQ2 Given realistic dependency structures, where in the space of domain count, cross-domain demand and central-team overhead does a mesh (or hybrid) deliver faster than a central team, and how does a nominal (“theater”) mesh compare?

H2a The mesh’s domain-team stability threshold under the all-pairs graph K_N lies below the largest $N \in [10, 160]$ at which the mesh remains stable on empirically structured graphs.

H2b At NAV’s size ($N=70$), the mesh’s lead-time disadvantage relative to the central model is larger on K_{70} than on the observed graph.

H3 Theater lead time is never below central lead time.

RQ3 Which of the modeled factors most change the mesh-versus-central lead-time ratio?

H1a and H1b were stated before the historical data were mined; H2a, H2b, H3 and the RQ3 analysis were pre-registered before any comparative simulation output was produced (notes/preregistration.md in the repository).

Scope. We study *delivery* performance (time from a request for a new analytical use case to its availability) and *sustainability* (queue stability) at equal headcount. We do not model data quality, regulatory risk, or the cost of the platform itself.

IV. METHODOLOGY

The study combines repository mining (RQ1), an analytical queueing model (Section V) and a discrete-event simulation with parameter sweeps (RQ2, RQ3). The mined structure feeds the models; the analytical results verify the simulator and explain its behavior.

A. Case and data source

NAV, Norway’s public labour and welfare administration, has been moving from a central data warehouse toward distributed, domain-owned data management inspired by data mesh [9]. Its application platform (NAIS) lets teams declare Kafka topics as Kubernetes resources (kafka.nais.io/v1, kind Topic); each manifest names the owning team and lists, in an access-control list (ACL), the teams and applications allowed to read, write or readwrite the topic [38], [39]. Many teams keep these manifests in public GitHub repositories, and some annotate them with catalog metadata in a DCAT-based [40] vocabulary (dcat.data.nav.no/*), which we interpret as publication

to a data-product catalog. The manifests therefore constitute a machine-readable, versioned record of *declared* cross-team data sharing.

B. Mining and graph construction (RQ1)

We retrieved every file in the navikt organization matching the Topic kind via the GitHub code-search API (566 files in 129 repositories; retrieved 23 September 2026). A read ACL for team C on a topic owned by O yields a directed edge $O \rightarrow C$ (data flows from producer to consumer); a write ACL for team W yields $W \rightarrow O$; same-team entries are ignored and parallel edges merged. About a third of the manifests are Handlebars templates rendered at deploy time with per-environment variable files. We render them with the nearest production variable file (105 topics) or, when rendering fails but a production pool is declared nearby, keep the literal ACL entries (60 topics); 70 topics whose environment cannot be resolved are excluded from the *primary* graph and used only in a sensitivity variant. The primary graph thus contains production topics only. We also report a *DCAT-only* graph restricted to catalogued topics.

For the growth analysis we made blobless partial clones of all repositories, found every path that ever contained a Topic manifest (including deleted ones, to avoid survivorship bias; renames are treated as delete-plus-add), parsed all 3,892 first-parent versions on the default branch, and replayed them to month-end snapshots. The final reconstructed snapshot contains 178 edges against 180 in the directly mined snapshot; the difference stems from ACL entries that are generated by template loops, which only the snapshot miner renders.

Analysis. Hub concentration (H1a) is tested against 2,000 uniform random digraphs with the same N and E , using degree Gini, maximum in-/out-degree and the share of edges incident to the top 10% of teams, with one-sided empirical p -values. We deliberately do not claim a power law, which 70 nodes cannot support [37]. Growth (H1b) is estimated by ordinary least squares on $\log E = c + a \log N$ over monthly snapshots with $N \geq 10$; because monthly observations are serially dependent, confidence intervals come from a moving-block bootstrap [41] (block length six months, 5,000 resamples).

C. Simulation study (RQ2) and sensitivity analysis (RQ3)

The simulation is described with the elements of the ODD protocol [42] in Section VI. Verification follows Sargent [43]: the simulator must reproduce exact queueing results (Erlang C), the analytical utilizations of Section V, the analytical stability thresholds, and the path-wise dominance of Proposition 3. Face validity is assessed in the spirit of pattern-oriented modeling [44]: the model should reproduce the qualitative patterns reported in the literature (central bottleneck at scale, hub domains as the most loaded, theater no better than centralization).

All four operating models are run on identical random workloads (common random numbers), so comparisons are paired. Each run has a 26-week warm-up, 104 weeks of

measured arrivals and a 104-week drain. RQ2 uses (i) a case study at $N=70$ on the observed NAV graph, on K_{70} , on a uniform random graph with the same number of edges and on a Chung–Lu graph [45] that resamples NAV’s degree sequence (20 replications each); (ii) a sweep over $N \in [10, 160]$ for four graph families; and (iii) decision maps over $N \times \eta$ and $N \times x$ (10 replications per cell). Synthetic graphs at size N follow the fitted growth law $E(N) = E_0(N/N_0)^a$ anchored at NAV’s (N_0, E_0) and keep NAV’s degree heterogeneity. For RQ3 we pre-registered a global sensitivity analysis (Saltelli’s design [46] with 1,024 base samples over eleven factors, i.e., 13,312 runs, and Sobol indices [47] computed with SALib [48], [49]); it is implemented in the artifact but was not executed for this version, so RQ3 is answered from the sweeps above.

D. Pre-registration and deviations

The hypotheses, default parameters, ranges and analysis plan were committed to the repository before any comparative simulation output was produced. Parameter revisions made before that point are documented with reasons (e.g., the governance team was resized so that the mesh is stable at NAV’s observed size, which NAV demonstrably operates). Four deviations occurred afterwards and are reported in Section VIII: (D1) a symmetric treatment of platform tooling became the primary specification after we found that the pre-registered specification unintentionally favored the mesh; the pre-registered specification is reported as a robustness check; (D2) the pre-registered instability criterion proved insensitive, so stability is classified primarily with the exact analytical criterion; (D3) an exploratory analysis of differential reuse was added; (D4) the global sensitivity analysis was not executed for this version.

V. FORMAL MODEL

A. Organization, demand and change

An organization has N domains connected by a directed dependency graph $G = (V, E)$; $u \rightarrow v$ means that domain v consumes data produced by u . We write $d^-(v)$ and $d^+(u)$ for in- and out-degree. Time is measured in weeks and effort in person-weeks (pw).

Definition 1 (Workload). (i) Each domain v requests new analytical use cases as a Poisson process of rate λ . With probability x (if $d^-(v) > 0$) a use case is *cross-domain* and needs data from $K_v = \min(1 + \text{Poisson}(\bar{k}-1), d^-(v))$ distinct providers drawn uniformly from v ’s in-neighbors. (ii) A use case costs consumer-side effort w_c and, per provider, provider-side effort w_p , which is needed only if no existing product can be reused (probability $1 - r$). (iii) Each cross-domain access requires approval: automated with probability g (costing w_{pc} to the builder), otherwise reviewed by a governance team (effort w_g). (iv) Each domain emits changes at rate μ ; a change breaks each outgoing dependency independently with probability $b = b_0(1 - \varphi)$, where φ is *contract discipline* (the share of breaking changes caught before release, e.g.,

by contract tests); a break costs producer-side coordination w_{co} and consumer-side rework w_{rw} . (v) Platform maturity $\pi \in [0, 1]$ multiplies data-engineering effort by $m_\pi = 1 - 0.4\pi$ and cross-team negotiation latency by $1 - 0.5\pi$.

B. Teams

A team of h people suffers Brooks-style communication overhead [14]: each additional colleague adds a fraction η to the time a member needs per unit of productive work, so per-capita efficiency is $1/(1 + \eta(h - 1))$ and the team’s effective capacity is

$$\kappa(h) = \frac{h}{1 + \eta(h - 1)} \text{ pw/week.} \quad (1)$$

Lemma 1. For $0 < \eta < 1$, κ is strictly increasing in $h \geq 1$ and $\kappa(h) < 1/\eta$ for all h .

Proof: $\kappa'(h) = (1 - \eta)/(1 + \eta(h - 1))^2 > 0$, and $\kappa(h) = 1/(\eta + (1 - \eta)/h) < 1/\eta$ because $(1 - \eta)/h > 0$. ■

A team is a work-conserving first-come-first-served (FCFS) multi-server queue whose servers process work at combined rate $\kappa(h)$. If the long-run rate L at which work is offered to a team satisfies $L \geq \kappa(h)$, its backlog cannot converge (it grows linearly when $L > \kappa$), so $L < \kappa$ is *necessary* for stability; under standard regularity conditions (e.g., renewal input) it is also sufficient [50]. Task streams inside the mesh are not exactly renewal, so for the mesh we use sufficiency as an approximation and check it by simulation (Section VIII). We call $\rho = L/\kappa$ the team’s utilization.

C. Operating models

All models have the same people: N_{cd} data engineers, a platform team that builds tooling (and thereby sets π) and a governance team; only their organization differs.

- **Central (C).** The N_{cd} engineers form one pool (scaled by a headcount ratio θ ; $\theta=1$ is budget-neutral). Lacking domain knowledge, it spends an extra s_{ctx} per domain touched. A use case is one job of size $(w_c + \sum w_p + s_{ctx}(1 + K))m_\pi$ plus automated-approval effort; a break is a job of size $(w_{rw} + s_{ctx})m_\pi$.
- **Mesh (M).** Each domain team (c_d people) builds its own use cases. For every provider, the use case waits a negotiation latency (ℓ_c on average), obtains approval, and the provider’s team performs $w_p m_\pi$; the consumer builds $w_c m_\pi$ once all providers are done (fork-join). A break costs w_{co} at the producer and $w_{rw} m_\pi$ at the consumer.
- **Theater (T).** Domains exist in name only: the central pool does all work as in C, plus a hand-off effort w_h per nominal domain owner, followed by a sign-off latency.
- **Hybrid (Y).** As M, but a pooled *core* formed from a share s of every domain’s engineers performs all provider-side work (with context cost s_{ctx} per provider task) and producer-side coordination; s minimizes the peak predicted utilization over all teams.

D. Loads and stability

Let $\bar{p} = (1-r)\bar{w}_p m_\pi + g w_{pc}$ be the expected provider task, $\beta^+ = \mu b \bar{w}_{co}$ and $\beta^- = \mu b \bar{w}_{rw} m_\pi$ the per-edge maintenance loads on producer and consumer, and $a_0 = \lambda \bar{w}_c m_\pi$ the own-use-case load. Since each cross-domain use case of v reaches a given in-neighbor with probability $\mathbb{E}[K_v]/d^-(v)$, the load on mesh domain i is

$$L_i = a_0 + \sum_{v: i \rightarrow v} \frac{\lambda x \mathbb{E}[K_v]}{d^-(v)} \bar{p} + \beta^+ d^+(i) + \beta^- d^-(i). \quad (2)$$

The central load L_C is the sum over domains of the use-case work of model C plus $|E| \mu b (\bar{w}_{rw} + s_{ctx}) m_\pi$ of rework.

Proposition 1 (Central scale bound). *Let $\eta > 0$. For every headcount, the central model is unstable whenever $N \geq \bar{N}_C := 1/(\eta \lambda \bar{w}_c m_\pi)$.*

Proof: Every use case carries at least $\bar{w}_c m_\pi$ expected work, so $L_C \geq N \lambda \bar{w}_c m_\pi$. By Lemma 1, $\kappa(H) < 1/\eta$ for every H . Hence $N \geq \bar{N}_C$ implies $L_C \geq 1/\eta > \kappa(H)$, violating the necessary condition for stability. ■

Proposition 2 (Mesh stability is a hub condition). *(a) The mesh's domain teams are stable iff $\max_i L_i < \kappa(c_d)$. (b) For every domain, $L_i \leq a_0 + (\lambda x \bar{p} + \beta^+) d^+(i) + \beta^- d^-(i)$. Hence $\max_i [(\lambda x \bar{p} + \beta^+) d^+(i) + \beta^- d^-(i)] < \kappa(c_d) - a_0$ is sufficient for stability: it depends on the degrees of the most connected domains, not on N or $|E|$. (c) On the all-pairs graph K_N , $L_i = a_0 + \lambda x \mathbb{E}[K] \bar{p} + (\beta^+ + \beta^-)(N-1)$ for every i , so the mesh is unstable once $N \geq \bar{N}_M^K := 1 + (\kappa(c_d) - a_0 - \lambda x \mathbb{E}[K] \bar{p})/(\beta^+ + \beta^-)$.*

Proof: (a) follows from the stability condition applied to each team. (b) $K_v \leq d^-(v)$, so each consumer v of i contributes at most $\lambda x \bar{p}$ to the sum in (2); there are $d^+(i)$ such consumers. (c) In K_N every domain has $d^+ = d^- = N-1$, and the provider term sums $N-1$ identical contributions $\lambda x \mathbb{E}[K] \bar{p}/(N-1)$. ■

Corollary 1 (Pooling hubs). *In the hybrid, the core is stable iff $\sum_v \lambda x_v \mathbb{E}[K_v] (\bar{p} + s_{ctx} m_\pi) + \beta^+ |E| < \kappa(s N c_d)$, and domain i iff $a_0 + \beta^- d^-(i) < \kappa((1-s)c_d)$. The provider-side constraint thus depends on $\sum_i d^+(i) = |E|$ rather than on $\max_i d^+(i)$: pooling replaces a hub constraint by an average-degree constraint, at the cost of context and communication overhead in the core.*

Proposition 3 (Theater never beats central). *Under common random numbers, every use case's lead time in model T is at least its lead time in model C.*

Proof: Approvals are served by identical governance teams from identical inputs, so each job joins the central queue at the same epoch in both models, and the queues have the same servers and speed. In T every job's work is that of C plus a non-negative hand-off effort. For an FCFS c -server queue, Kiefer and Wolfowitz [50] give the workload recursion $\mathbf{W}_{n+1} = \mathcal{R}((\mathbf{W}_n + S_n \mathbf{e}_1 - \tau_{n+1} \mathbf{1})^+)$, where $\mathcal{R}$ sorts in ascending order, S_n is the n -th job's service time, τ_{n+1} the

next inter-arrival time, and the n -th waiting time is the first component of $\mathbf{W}_n$. Adding S_n to the smallest component, subtracting a constant, taking positive parts and sorting are all component-wise non-decreasing maps (order statistics are monotone), so by induction $\mathbf{W}'_n \geq \mathbf{W}_n$ whenever $S'_k \geq S_k$ for all $k < n$. Hence every job waits at least as long in T, is served at least as long, and then incurs a non-negative sign-off latency. ■

Remark 1. With the default parameters of Section VI, $\bar{N}_C = 500$ and $\bar{N}_M^K = 151$. The observed NAV graph satisfies the sufficient condition of Proposition 2(b), so its mesh is provably stable at these parameters. Propositions 1–3 concern long-run feasibility. Lead times below saturation depend on queueing delays, fork-join synchronization and negotiation latency, which the simulation quantifies.

VI. PROPOSED APPROACH: SIMULATION DESIGN

We describe the simulator with the elements of the ODD protocol [42].

Purpose. To estimate, for a given dependency graph and parameter vector, the delivery lead time of new analytical use cases under the four operating models of Section V, at equal headcount, and to identify which inputs drive the difference.

Entities and state. Teams (central pool, domain teams, hybrid core, governance team), each a FCFS multi-server resource whose servers work at speed $\kappa(h)/[h]$; use cases (arrival time, consumer, providers, effort draws, approval and latency draws); and breaking changes (time, producer, consumer, effort draws).

Process overview and scheduling. Discrete-event simulation. Use cases and changes are generated in advance for a replication (Definition 1), then replayed against each operating model, so that all models face the same arrivals, efforts and draws (common random numbers). Model-specific routing follows Section V. Lead time is measured from the request to the completion of the consumer-side build.

Stochasticity. Efforts are log-normal with coefficient of variation 1; latencies and approval/coordination efforts are exponential; arrivals and changes are Poisson. Independent random streams are derived from one seed per replication.

Observation. Mean, median and 90th-percentile lead time of use cases arriving during the 104-week measurement window (unfinished ones are censored at the end of the run), separately for local and cross-domain use cases; team utilizations; and the lead-time trend within the window.

Parameters. Table I lists defaults and sensitivity ranges. They are assumptions chosen for plausibility, not calibrated to outcomes, with two exceptions marked in the table: the graph exponent is taken from RQ1, and the governance team is sized so that the mesh is stable on NAV's observed graph, a constraint implied by NAV's operation of a mesh. The communication overhead η is the least grounded parameter; its range bounds the per-capita efficiency of a 150-person team between 40% and 100% (at the default, 57%), and results are reported as maps along η .

TABLE I

MODEL PARAMETERS (DEFAULTS AND RANGES USED IN THE SENSITIVITY ANALYSIS). EFFORTS IN PERSON-WEEKS. SOURCE: A = ASSUMPTION, E = ESTIMATED IN RQ1, S = STABILITY CONSTRAINT AT NAV’S SIZE.

Symbol	Meaning	Default	Range	Src.
N	domains	70	10–160	E
a	graph growth exponent	2.03	1.3–2.4	E
λ	use cases per domain per week	0.5	–	A
x	cross-domain share of use cases	0.3	0.05–0.6	A
$\bar{k}$	providers per cross-domain case	1.5	–	A
$\bar{w}_c, \bar{w}_p$	consumer / provider effort	1.0 / 0.5	–	A
r	reuse of existing products	0.4	0–0.8	A
μ, b_0	change rate; break probability	0.2, 0.3	–	A
φ	contract discipline	0.5	0–0.9	A
$\bar{w}_{co}, \bar{w}_{rw}$	coordination / rework per break	0.1 / 0.3	–	A
c_d	engineers per domain team	2	–	A
θ	central headcount ratio	1.0	0.6–1.2	A
η	communication overhead	0.005	0–0.01	A
s_{ctx}	central context effort per domain	0.2	0–0.5	A
π	platform maturity	0.5	0–1	A
ℓ_c	mean negotiation latency (weeks)	1.0	0.25–2	A
g	share of approvals automated	0.5	0–1	A
$\bar{w}_g$	manual approval effort	0.25	–	A
–	governance team size / N	0.06	–	S

Design choices favoring the central model. Contract discipline reduces breakage for central pipelines as much as for data products, although contracts are a mesh construct; the central model pools all engineers into one queue, which reduces waiting through resource pooling; and platform tooling benefits central and domain engineers equally. Any mesh advantage we report is therefore conservative with respect to these choices.

VII. IMPLEMENTATION

The artifact is a Python 3 package (`meshsim`) with four modules: *mining* (GitHub API access with an on-disk cache, Handlebars rendering of NAIS templates, parsing, and history reconstruction with partial clones and `git backfill`), *graphs* (construction, metrics, Chung–Lu, Erdős–Rényi and complete-graph generators, using NetworkX [51]), *analytic* (the loads, thresholds and Erlang-C formulas of Section V) and *sim* (the discrete-event models, built on SimPy [52]). Numbered scripts run the pipeline end to end: *mining*, RQ1 analysis, verification, RQ2 experiments, the Sobol analysis, the statistical analysis and the figures. The analysis script also emits a $\LaTeX$ file of macros from which every number quoted in this paper is typeset, so that text and results cannot diverge.

A test suite (23 tests, `pytest`) covers manifest parsing and edge semantics, template resolution, the analytical formulas (including Lemma 1 and the thresholds of Propositions 1–2), agreement of the simulator with Erlang C, path-wise theater dominance on random configurations, reproducibility under fixed seeds, and agreement of simulated and analytical utilizations. Replications run in parallel; a run at $N=160$ takes about half a second per operating model on a laptop, and all RQ2 experiments complete in under ten minutes on eight cores. The pre-registered Sobol analysis (script `06_rq3_sobol.py`) is included but was not run for this version. The mined data contain team and topic names that NAV publishes openly; no personal data are collected, and all analyses are at team level.

TABLE II

VERIFICATION: MEAN WAITING TIME (WEEKS) OF THE REDUCED CENTRAL MODEL AGAINST ERLANG C (10 REPLICATIONS OF 6,000 WEEKS EACH).

ρ	c	W_q (Erlang C)	W_q (sim.)	95% CI $\pm$
0.3	4	0.013	0.012	0.002
0.5	4	0.087	0.084	0.005
0.7	4	0.357	0.368	0.026
0.9	4	1.969	2.044	0.227

VIII. RESULTS AND EVALUATION

A. Verification of the simulator

Table II compares simulated and exact M/M/4 waiting times (Erlang C) for the central model reduced to a single class without overhead; all four estimates agree within their 95% confidence intervals. Over 200 random configurations the median absolute difference between simulated and analytical utilization is 1.4 percentage points for the central pool and 2.5 points on average across domain teams. Proposition 3 held for all 720,955 use cases in those configurations (0 violations). Beyond the thresholds of Propositions 1 and 2(c), simulated lead times grow without bound: on K_N the mesh’s mean lead time rises from 4.4 weeks at $N=136$ to 20.7 weeks at $N=181$ ($\bar{N}_M^K=151$); a central pool of $N=458$ domains, above its exact asymptotic limit of 417, remains unstable even with four times the budget-neutral headcount.

B. RQ1: structure and growth of cross-domain dependencies

The primary NAV graph has 70 teams and 180 directed cross-team dependencies, i.e., a density of 3.7%: of the 4,830 ordered pairs that an all-pairs assumption would count, only one in 27 carries a declared dependency. The graph is nonetheless well connected (the largest weakly connected component holds 90% of teams) and reciprocity is 0.41. Dependencies concentrate on a few hubs (Fig. 1a): one team consumes data from 31 other teams (in-degree) and another supplies 23 teams (out-degree); the top 10% of teams touch 66% of all edges. Against 2,000 uniform random graphs with the same N and E , the degree Gini (0.53 vs. 0.24, 97.5% quantile 0.28), the maximum in-degree (31 vs. at most 9), and the top-decile edge share (66% vs. 33%) all lie outside the null distribution ($p < 0.001$ for each). **H1a is supported**, and the result is unchanged when the 70 unresolved topics are included ($N=71$, $E=200$).

Fig. 1b shows growth from 2021-05 to September 2026 (65 month-end snapshots with $N \geq 10$). The fitted exponent is $a = 2.03$ (95% block-bootstrap CI [1.99, 2.31]): edges grew *quadratically* in the number of participating teams, with density stable between 3.0% and 4.5%. **H1b is not supported**. The all-pairs assumption thus has the right functional form for this organization but overstates the *number* of dependencies by a factor of about 27 at every size. The subset of topics that teams catalogue as data products (DCAT-annotated; $N=24$, $E=39$) behaves differently: $a = 1.37$ [1.29, 1.50], i.e., sub-quadratic densification.

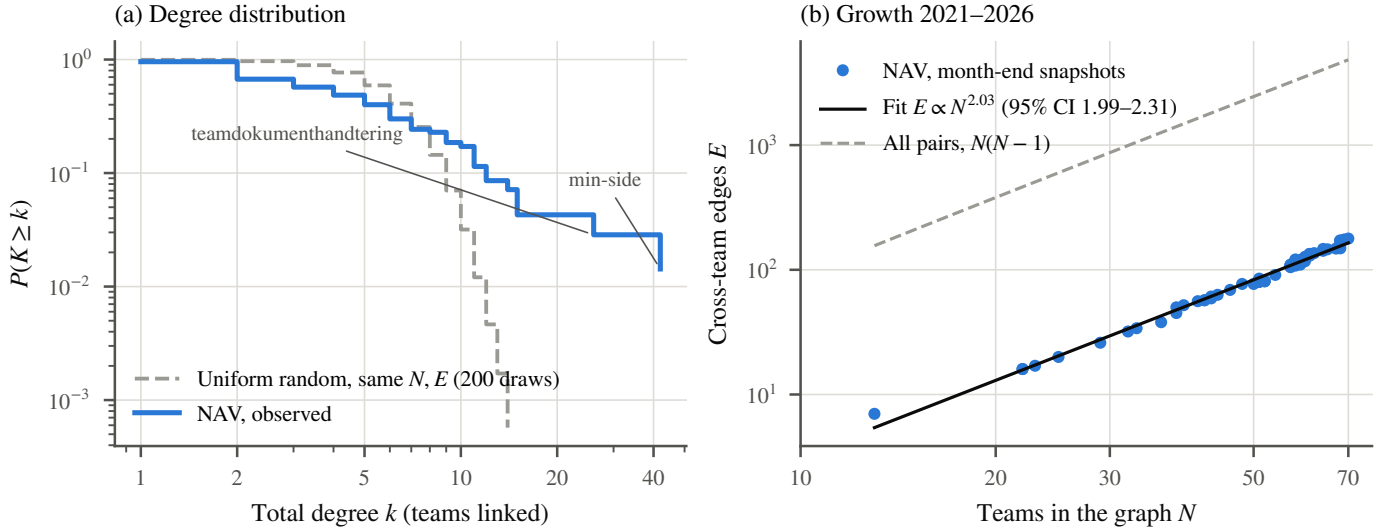


Fig. 1. RQ1. (a) Complementary cumulative degree distribution of the NAV dependency graph against 200 uniform random graphs with the same numbers of teams and edges; the two largest hubs are labeled. (b) Cross-team edges against participating teams for monthly snapshots, 2021–2026, with the fitted growth law and the all-pairs reference $N(N-1)$ (log–log).

C. RQ2: operating models at NAV's size

Table III reports the case study at $N=70$. On the observed NAV graph the mesh has a mean lead time of 1.17 weeks against 1.89 for the central model, a paired difference of -0.72 weeks (95% CI [-0.73, -0.72]); the advantage holds for cross-domain use cases (2.20 vs. 2.75), which pay negotiation latency and fork–join synchronization in the mesh. The hybrid (1.32) lies between the two, and the theater mesh is slowest (2.15). The mesh's busiest team—a hub—runs at a predicted utilization of 0.53, higher than the central pool (0.49): the mesh is faster not because it is less loaded but because its small teams avoid the central pool's communication overhead and context costs. Results on the Chung–Lu and uniform random graphs with the same edge count are nearly identical, so at this size the hubs matter little for mean lead time.

H2b is not supported—in the opposite direction. On K_{70} , the central model becomes unstable (predicted utilization 1.19) with a mean lead time of 16.8 weeks, while the mesh remains stable (1.6 weeks). The difference-in-differences is -14.6 weeks (95% CI [-15.4, -13.7]): assuming all-pairs dependencies does not bias the comparison against the mesh but against the central team, which must absorb all maintenance of all pairs in one pool whose capacity is bounded (Lemma 1). **H3 is supported:** in all 80 paired case-study runs the theater mesh was slower than the central model (minimum difference 0.203 weeks), as Proposition 3 requires.

D. RQ2: scaling and decision maps

Fig. 2 extends the comparison to $N \in [10, 160]$. On graphs that follow the measured growth law (a), a sparser law (b) or have the same density without hubs (c), the mesh is stable throughout, its lead time nearly flat, while the central model's lead time rises with N as the pool grows and its per-capita efficiency falls. On K_N (d) both the central and theater models

TABLE III
CASE STUDY AT $N=70$: MEAN USE-CASE LEAD TIME $\overline{LT}$ (WEEKS, 20 PAIRED REPLICATIONS), MEAN LEAD TIME OF CROSS-DOMAIN USE CASES $\overline{LT}_x$, AND PREDICTED PEAK UTILIZATION $\hat{\rho}_{\max}$ ([†] UNSTABLE).

Graph	Model	$\overline{LT}$	95% CI	$\overline{LT}_x$	$\hat{\rho}_{\max}$
NAV (observed)	central	1.89	[1.88, 1.90]	2.75	0.49
	theater	2.15	[2.14, 2.17]	3.83	0.51
	mesh	1.17	[1.16, 1.18]	2.20	0.53
	hybrid	1.32	[1.30, 1.33]	2.45	0.36
Chung–Lu, $\alpha=2.03$	central	1.89	[1.88, 1.90]	2.74	0.49
	theater	2.14	[2.12, 2.16]	3.83	0.51
	mesh	1.15	[1.14, 1.16]	2.17	0.39
	hybrid	1.32	[1.30, 1.35]	2.46	0.31
Erdős–Rényi	central	1.95	[1.93, 1.96]	2.77	0.50
	theater	2.26	[2.24, 2.27]	3.88	0.52
	mesh	1.22	[1.21, 1.23]	2.19	0.34
	hybrid	1.47	[1.46, 1.48]	2.56	0.34
All-pairs K_N	central	16.83	[15.90, 17.76]	17.66	1.19 [†]
	theater	32.21	[31.07, 33.35]	33.80	1.38 [†]
	mesh	1.55	[1.53, 1.57]	2.67	0.58
	hybrid	1.66	[1.62, 1.69]	2.78	0.63

become unstable from $N=55$ (theater) and $N=70$ (central), while the mesh saturates only near $\bar{N}_M^K=151$. The largest N at which the mesh remains stable on empirically structured graphs (160, the top of the range) exceeds $\bar{N}_M^K$, so **H2a is supported**.

The decision maps (Fig. 3) show which assumption drives this picture. *Without communication overhead in the central team* ($\eta=0$, bottom row of Fig. 3a) the two models deliver within 8% of each other at every size, with the mesh marginally faster up to 40 domains and the central model marginally faster from 70; the hybrid is then up to 18% slower than the central model. For every $\eta > 0$ in our range, the mesh's advantage grows with N and with η . Raising the share

of cross-domain use cases from $x=0.05$ to $x=0.6$ (Fig. 3b) shrinks the advantage, measured as the $\log_2$ ratio, by 30–61% depending on N , but does not reverse it at the default overhead.

E. RQ3: which factors decide

The pre-registered global (Sobol) sensitivity analysis is implemented in the artifact but was not executed for this version (deviation D4), so we answer RQ3 from the one- and two-factor sweeps above, which vary each factor around the defaults and do not quantify interactions.

Central-team overhead η dominates. Moving η from 0 to 0.01 changes the outcome from near parity (within 8% at every N) to a mesh that is several times faster at large N , where the central pool approaches or exceeds its capacity bound (Fig. 3a). *Domain count amplifies* whatever η implies, because the central pool’s per-capita efficiency falls as it grows. *Cross-domain demand* is second-order: from $x=0.05$ to $x=0.6$ the advantage shrinks by 30–61% in $\log_2$ terms without reversing (Fig. 3b). *Dependency structure* matters little in the empirically plausible range—the observed, Chung–Lu and uniform random graphs give nearly identical results at $N=70$ (Table III) and similar curves up to $N=160$ (Fig. 2)—and becomes decisive only toward the all-pairs extreme. *Reuse* has the smallest effect of the factors tested (below).

F. Robustness and exploratory analyses

Pre-registered specification (D1). Under the pre-registered specification, which grants platform-tooling benefits only to domain teams, the mesh’s advantage at $N=70$ is larger (1.17 vs. 2.38 weeks; paired difference -1.21), and H2b and H3 lead to the same verdicts. The symmetric specification we report as primary is therefore the more conservative one.

Differential reuse (D3, exploratory). The only formal model of data-mesh economics predicts that domains under-invest in reusable products [5]. Lowering the mesh’s reuse probability by up to 0.4 (i.e., to zero at the default) changed the $\log_2$ lead-time ratio by at most 0.08 at any N in $\{10, 40, 70, 100, 160\}$ and never reversed its sign: in this model, reuse acts only on provider-side build effort, a small share of total work. Under-investment could matter more through channels we do not model, such as semantic inconsistency (Section X).

IX. DISCUSSION AND LIMITATIONS

Answering the original question. Is data mesh suitable for large organizations? Our results replace a yes-or-no answer with conditions. The empirical part shows that the dependency structure of a real mesh is far from all-pairs and dominated by hubs; the analytical part shows what each operating model’s scaling limit depends on—total load against a bounded pool for the central model, the most connected domains for the mesh; and the simulation shows that, at equal headcount, delivery time favors the mesh to the extent that a large central team loses per-capita efficiency and must acquire context for every domain it serves. Where those diseconomies are small, a central team is not slower, and pooling even gives it an edge

at large N . The “key implementation factors” sought in the original proposal are, in this light, quantities an organization can estimate before deciding.

Why the all-pairs assumption cuts the other way.

Intuition suggests that dense dependencies hurt decentralized organizations most, because they multiply negotiations between teams. In our model, dense dependencies multiply *maintenance*—breaking changes that must be absorbed downstream—and whoever maintains the pipelines absorbs it. A mesh spreads that load over many small teams, each with near-full efficiency; a central model concentrates it in one pool whose capacity cannot exceed $1/\eta$. This does not refute economic analyses that emphasize under-investment in reusable data under decentralization [5]: that mechanism acts on semantic quality and duplicated effort, and in our exploratory test a reuse penalty alone did not change the comparison. It does show that conclusions drawn from an all-pairs assumption should be revisited once the dependency structure is measured.

Hubs, hybrids and theater. Proposition 2 and Corollary 1 explain a practitioner observation: meshes fail locally, at the domains everyone depends on, before they fail globally. At NAV, one team consumes from 31 others; staffing such hubs for their load, or pooling their provider-side work, is the analytical rationale for the hub-and-spoke designs practitioners converge on [13], [30]. In our runs the hybrid rarely beat the mesh on mean lead time—the pooled core pays context costs—but it relieves hub saturation by construction. The theater result is unconditional: renaming teams adds hand-offs to the same central pool and can only slow delivery (Proposition 3), consistent with the anti-pattern reported in practice [13] and with findings that ownership often does not move in adoptions [8].

Limitations. Beyond the threats in Section X, the model abstracts teams as queues with fixed capacity, ignores learning, turnover and change costs of a reorganization, and measures only delivery time. It treats governance as approvals with an automation share, whereas federated governance also shapes semantics and quality [6], [12]. Its value lies in making assumptions explicit and testable, not in point predictions.

X. THREATS TO VALIDITY

We follow the classification of Wohlin et al. [53].

Construct validity. (i) Kafka topic ACLs record *declared*, *operational* data sharing. Only a minority of topics carry catalog annotations, and NAV’s analytical products on other platforms are not visible in public repositories. We therefore speak of declared cross-domain dependencies and report the DCAT-only graph separately; its different growth exponent shows that the answer to H1b depends on what counts as a data product. (ii) NAIS teams (namespaces) are proxies for domains; NAV has moved toward autonomous product teams [9], which fits data mesh’s notion of a domain, but some teams may be platform or enabling teams. (iii) Lead time is one outcome among several that matter; quality, compliance and cost are outside the model.

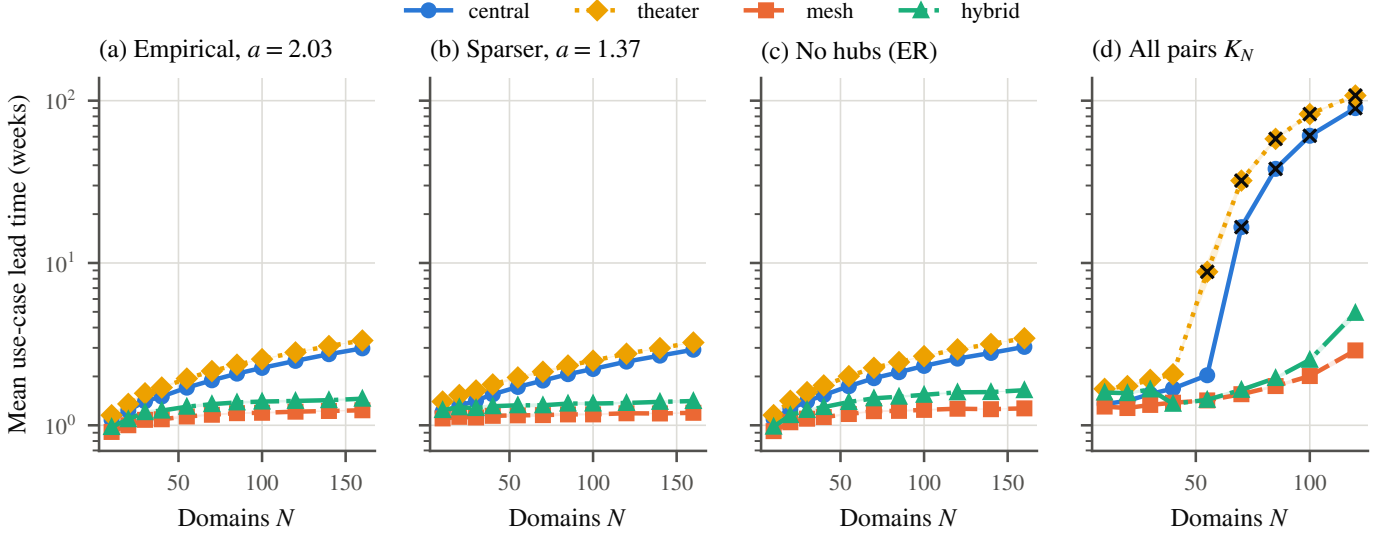


Fig. 2. RQ2 scaling: mean use-case lead time (log scale; means of 10 paired replications, shaded 95% CIs) against the number of domains for four dependency-graph families. Crosses mark configurations that are analytically unstable (a queue grows without bound).

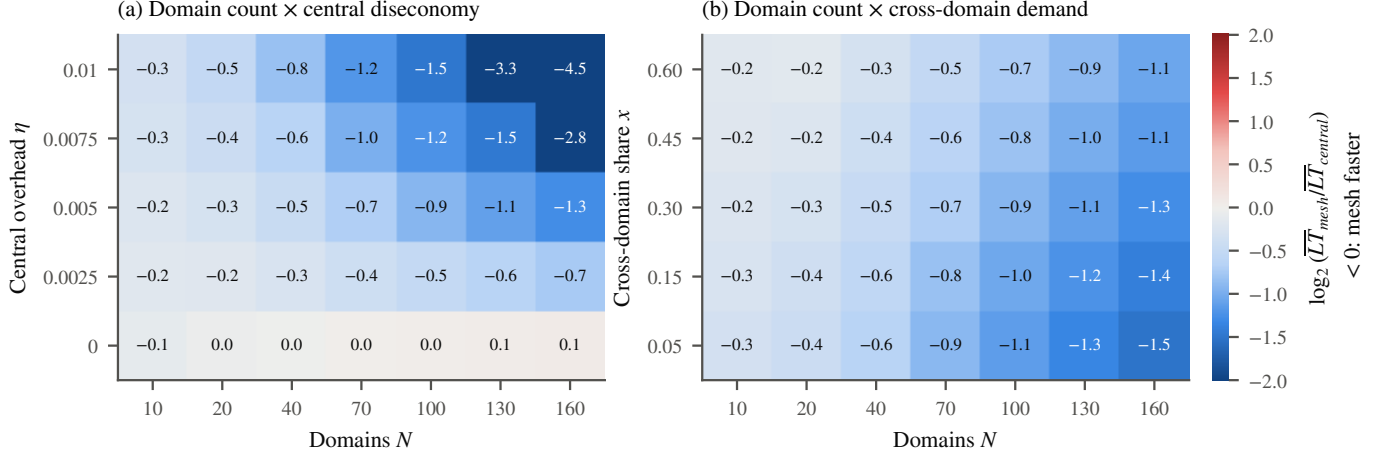


Fig. 3. RQ2 decision maps: $\log_2$ of the ratio of mean lead times, mesh over central (blue: mesh faster; -1 means twice as fast), on graphs following the measured growth law. (a) Domains versus the central team’s communication overhead η ; (b) domains versus the share of cross-domain use cases x (at $\eta=0.005$). Cell values are means of 10 paired replications.

Internal validity. (i) The graph is a lower bound: private repositories are invisible, code search indexes only default branches, and repository data carry the usual perils of mining GitHub [54]. The unchanged hub results on the larger variant with inferred topics, and the robustness of RQ2 across graph families including the all-pairs extreme, limit the consequences. (ii) Template resolution is heuristic; 70 topics could not be assigned to an environment, and historical templated versions inherit their path’s current classification. (iii) Growth over time conflates organizational growth with platform adoption: N counts teams participating in declared sharing, so part of its increase may reflect teams adopting the Topic-resource mechanism rather than new domains. (iv) The block bootstrap assumes serial dependence decays within six months.

Conclusion validity. Simulation estimates are precise (paired common random numbers; narrow intervals) but precision is not accuracy: conclusions are conditional on the model. We therefore report results as maps and sensitivity indices, verified the simulator against exact results, and pre-registered hypotheses. Four deviations from the pre-registration (D1–D4) are disclosed; the pre-registered specification yields the same verdicts. RQ3 rests on one- and two-factor sweeps around the defaults (D4), which cannot reveal interactions between factors or regions of the parameter space far from the defaults; the pre-registered global analysis would address this.

External validity. One public-sector organization in one country is studied; other organizations may have denser or sparser dependency structures, different team sizes, or analytical products that are not event streams [55]. The model’s

parameters are assumptions, and the single most influential one—communication overhead in large central teams—is not measured in our data. The simplified capacity model (1) treats a central team as one coupled pool; real central teams are subdivided, which lowers effective overhead, and our range for η is meant to span that case down to zero. Finally, the model omits organizational change costs, learning, and the semantic-interoperability costs of decentralization emphasized by economic analyses [5].

XI. CONCLUSION AND FUTURE WORK

We set out to make an old question—is data mesh suitable for large organizations, and what determines success?—falsifiable. Mining 566 infrastructure manifests from a large public organization, we found that declared cross-domain dependencies are sparse (3.7% of pairs) and concentrated on a few hub teams, and that they grew quadratically at a stable low density rather than sub-quadratically as we had hypothesized; catalogued data products alone densified sub-quadratically. We proved that a central team’s capacity is bounded under communication overhead whatever its size, that mesh stability is governed by its most connected domains, and that relabeling without decentralization can never outperform centralization, and we verified a simulator against these results. In the simulation, the all-pairs assumption penalized the central model rather than the mesh, and the mesh’s delivery advantage depended above all on whether a large central team loses efficiency as it grows: without such overhead the two models were within a few percent of each other.

For practitioners, this shifts the adoption question from “mesh or not” to measurable conditions: how much a central team’s per-capita throughput falls as it grows, how much context it must acquire per domain, how concentrated the organization’s data dependencies are, and whether hub domains are staffed for their load. For researchers, it offers a dataset, a verified open model and a pre-registered design that others can re-run on their own organizations.

Future work should (i) measure communication overhead and context costs in real data teams, which would turn our maps into predictions; (ii) replicate the dependency analysis in organizations whose analytical products live in warehouses or lakehouses rather than event streams; (iii) model the semantic-integration costs that economic analyses emphasize [5]; and (iv) extend the model to AI agents as data-product consumers, whose access is increasingly mediated by automated governance [27], [28] and whose security properties under adversarial inputs remain largely unexamined [56].

ACKNOWLEDGMENTS

The author thanks Prof. Guenther Ruhe and Prof. Tim Reimer (University of Calgary), who supervised the original research proposal from which this work grew. The present study was carried out independently; any errors are the author’s own. The author also thanks NAV for publishing its platform configuration openly.

REFERENCES

- [1] Z. Dehghani, “How to move beyond a monolithic data lake to a distributed data mesh,” *martinfowler.com*, May 2019. [Online]. Available: <https://martinfowler.com/articles/data-monolith-to-mesh.html>
- [2] —, “Data mesh principles and logical architecture,” *martinfowler.com*, Dec. 2020. [Online]. Available: <https://martinfowler.com/articles/data-mesh-principles.html>
- [3] —, *Data Mesh: Delivering Data-Driven Value at Scale*. O’Reilly Media, 2022.
- [4] A. Goedegebuure, I. Kumara, S. Driessen, W.-J. van den Heuvel, G. Monsieur, D. A. Tamburri, and D. D. Nucci, “Data mesh: A systematic gray literature review,” *ACM Computing Surveys*, vol. 57, no. 1, pp. 1–36, 2024.
- [5] G. Besanson, “The data hydration gap: A formal model of underinvestment in general-purpose data products under decentralized governance,” *arXiv preprint arXiv:2604.00218*, 2026. [Online]. Available: <https://arxiv.org/abs/2604.00218>
- [6] J. Wasser, I. Kumara, G. Monsieur, W.-J. van den Heuvel, and D. A. Tamburri, “Data contracts in data mesh: A systematic gray literature review,” in *Lecture Notes in Business Information Processing*. Springer, 2025, pp. 21–38.
- [7] J. Bode, N. Kühl, D. Kreuzberger, and C. Holtmann, “Toward avoiding the data mess: Industry insights from data mesh implementations,” *IEEE Access*, vol. 12, pp. 95 402–95 416, 2024.
- [8] R. S. C. Winter and T. Hackl, “Exploring data mesh adoption in large organizations,” *Issues in Informing Science and Information Technology*, vol. 22, p. 012, 2025.
- [9] K. Vestues, G. K. Hanssen, M. Mikalsen, T. A. Buan, and K. Conboy, “Agile data management in NAV: A case study,” in *Agile Processes in Software Engineering and Extreme Programming (XP 2022)*, ser. *Lecture Notes in Business Information Processing*. Springer, 2022, pp. 220–235.
- [10] A. Barbala, G. K. Hanssen, and T. Sporse, “Towards a common data-driven culture: A longitudinal study of the tensions and emerging solutions involved in becoming data-driven in a large public sector organization,” *Journal of Systems and Software*, vol. 218, p. 112185, 2024.
- [11] T. van Eijk, I. Kumara, D. D. Nucci, D. A. Tamburri, and W.-J. van den Heuvel, “Architectural design decisions for self-serve data platforms in data meshes,” in *Proc. IEEE 21st International Conference on Software Architecture (ICSA)*, 2024, pp. 135–145.
- [12] M. Tonnarelli, T. van Eijk, I. Kumara, D. D. Nucci, D. A. Tamburri, and W.-J. van den Heuvel, “Architectural design decisions for federated computational governance in data meshes,” in *Proc. IEEE 23rd International Conference on Software Architecture (ICSA)*, 2026, pp. 143–154.
- [13] S. Werner, K. Prakash, L. Aliyeva, D. Sato, and J. Spens, “The state of data mesh in 2026: From hype to hard-won maturity,” *Thoughtworks Insights (blog)*, Jan. 2026. [Online]. Available: <https://www.thoughtworks.com/en-us/insights/blog/data-strategy/the-state-of-data-mesh-in-2026-from-hype-to-hard-won-maturity>
- [14] F. P. Brooks, Jr., *The Mythical Man-Month: Essays on Software Engineering*. Addison-Wesley, 1975.
- [15] I. A. Machado, C. Costa, and M. Y. Santos, “Data mesh: Concepts and principles of a paradigm shift in data architectures,” *Procedia Computer Science*, vol. 196, pp. 263–271, 2022.
- [16] J.-G. Perrin and E. Broda, *Implementing Data Mesh: Design, Build, and Implement Data Contracts, Data Products, and Data Mesh*. O’Reilly Media, 2024.
- [17] Bitol Project, Linux Foundation AI & Data, “Open data contract standard (ODCS),” Specification, 2026. [Online]. Available: <https://bitol.io/>
- [18] I. Ramos, M. Y. Santos, D. Joshi, and S. Pratik, “Data mesh adoption: A multi-case and multi-method readiness approach,” in *Lecture Notes in Business Information Processing*. Springer, 2024, pp. 16–29.
- [19] I. Kumara, S. Driessen, T. van Eijk, D. D. Nucci, D. A. Tamburri, and W.-J. van den Heuvel, “Data mesh architecture: From theory to practice,” in *Companion Proc. IEEE 21st International Conference on Software Architecture (ICSA-C)*, 2024, pp. 375–376.
- [20] L. Schuiki, C. Giebler, E. Hoos, and H. Schwarz, “Unraveling data mesh: Current state, challenges and research gaps,” in *Communications in Computer and Information Science*. Springer, 2025, pp. 59–79.
- [21] I. Blohm, F. Wortmann, C. Legner, and F. Köbler, “Data products, data mesh, and data fabric,” *Business & Information Systems Engineering*, vol. 66, no. 5, pp. 643–652, 2024.

- [22] A. Gieß and A. Hutterer, “The future of data management: A delimitation of data platforms, data spaces, data meshes, and data fabrics,” *Information Systems and e-Business Management*, vol. 23, no. 4, pp. 971–997, 2025.
- [23] S. Hoseini, C. Quix, and S. Decker, “Data architectures and their technical requirements (DATER),” arXiv preprint arXiv:2606.08811, 2026. [Online]. Available: <https://arxiv.org/abs/2606.08811>
- [24] A. Wider, K. Jarmul, and A. Akhtar, “Towards automating federated data governance,” in *Proc. IEEE International Conference on Web Services (ICWS)*, 2024, pp. 10–19.
- [25] A. Dolhopolov, A. Castelltort, and A. Laurent, “Implementing federated governance in data mesh architecture,” *Future Internet*, vol. 16, no. 4, p. 115, 2024.
- [26] M. Brambilla and P. Plebani, “Scalable policy-as-code decision points for data products,” in *Proc. IEEE International Conference on Web Services (ICWS)*, 2025, pp. 924–930.
- [27] A. Wider, S. Harter, and L. W. Dietz, “AI-assisted data governance with data mesh manager,” in *Proc. IEEE International Conference on Web Services (ICWS)*, 2025, pp. 963–965.
- [28] M. Tonnarelli, F. Scaramuzza, S. Harter, and L. W. Dietz, “Data product MCP: Chat with your enterprise data,” arXiv preprint arXiv:2601.08687, 2026. [Online]. Available: <https://arxiv.org/abs/2601.08687>
- [29] A. Dolhopolov, A. Castelltort, and A. Laurent, “Trick or treat: Centralized data lake vs decentralized data mesh,” in *Management of Digital EcoSystems (MEDES 2023)*, ser. Communications in Computer and Information Science. Springer, 2024, pp. 303–316.
- [30] O. Angéilil and J. Migon, “Beyond the data mesh illusion: Designing modern AI-augmented lakehouses to bridge the gap between theory and practice,” arXiv preprint arXiv:2605.27131, 2026. [Online]. Available: <https://arxiv.org/abs/2605.27131>
- [31] M. E. Conway, “How do committees invent?” *Datamation*, vol. 14, no. 4, pp. 28–31, 1968.
- [32] A. MacCormack, C. Y. Baldwin, and J. Rusnak, “Exploring the duality between product and organizational architectures: A test of the “mirroring” hypothesis,” *Research Policy*, vol. 41, no. 8, pp. 1309–1324, 2012.
- [33] T. W. Malone and K. Crowston, “The interdisciplinary study of coordination,” *ACM Computing Surveys*, vol. 26, no. 1, pp. 87–119, 1994.
- [34] M. Cataldo, J. D. Herbsleb, and K. M. Carley, “Socio-technical congruence: A framework for assessing the impact of technical and work dependencies on software development productivity,” in *Proc. 2nd ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2008, pp. 2–11.
- [35] J. Leskovec, J. Kleinberg, and C. Faloutsos, “Graphs over time: Densification laws, shrinking diameters and possible explanations,” in *Proc. 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2005, pp. 177–187.
- [36] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol. 286, no. 5439, pp. 509–512, 1999.
- [37] A. Clauset, C. R. Shalizi, and M. E. J. Newman, “Power-law distributions in empirical data,” *SIAM Review*, vol. 51, no. 4, pp. 661–703, 2009.
- [38] NAIS / NAV IT, “Kafka on NAIS: Topic resources and access control,” NAIS platform documentation, 2026. [Online]. Available: <https://doc.nais.io/persistence/kafka/>
- [39] —, “Kafkarator: Operator for Aiven Kafka topic and user management,” GitHub repository, 2026. [Online]. Available: <https://github.com/nais/kafkarator>
- [40] W3C, “Data catalog vocabulary (DCAT) – version 3,” W3C Recommendation, Aug. 2024. [Online]. Available: <https://www.w3.org/TR/vocab-dcat-3/>
- [41] H. R. Künsch, “The jackknife and the bootstrap for general stationary observations,” *The Annals of Statistics*, vol. 17, no. 3, 1989.
- [42] V. Grimm, S. F. Railsback, C. E. Vincenot, U. Berger *et al.*, “The ODD protocol for describing agent-based and other simulation models: A second update to improve clarity, replication, and structural realism,” *Journal of Artificial Societies and Social Simulation*, vol. 23, no. 2, p. 7, 2020.
- [43] R. G. Sargent, “Verification and validation of simulation models,” *Journal of Simulation*, vol. 7, no. 1, pp. 12–24, 2013.
- [44] V. Grimm, E. Revilla, U. Berger, F. Jeltsch, W. M. Mooij, S. F. Railsback, H.-H. Thulke, J. Weiner, T. Wiegand, and D. L. DeAngelis, “Pattern-oriented modeling of agent-based complex systems: Lessons from ecology,” *Science*, vol. 310, no. 5750, pp. 987–991, 2005.
- [45] F. Chung and L. Lu, “The average distances in random graphs with given expected degrees,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 25, pp. 15 879–15 882, 2002.
- [46] A. Saltelli, P. Annoni, I. Azzini, F. Campolongo, M. Ratto, and S. Tarantola, “Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index,” *Computer Physics Communications*, vol. 181, no. 2, pp. 259–270, 2010.
- [47] I. M. Sobol’, “Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates,” *Mathematics and Computers in Simulation*, vol. 55, no. 1–3, pp. 271–280, 2001.
- [48] J. Herman and W. Usher, “SALib: An open-source python library for sensitivity analysis,” *Journal of Open Source Software*, vol. 2, no. 9, p. 97, 2017.
- [49] T. Iwanaga, W. Usher, and J. Herman, “Toward SALib 2.0: Advancing the accessibility and interpretability of global sensitivity analyses,” *Socio-Environmental Systems Modelling*, vol. 4, p. 18155, 2022.
- [50] J. Kiefer and J. Wolfowitz, “On the theory of queues with many servers,” *Transactions of the American Mathematical Society*, vol. 78, no. 1, pp. 1–18, 1955.
- [51] A. A. Hagberg, D. A. Schult, and P. J. Swart, “Exploring network structure, dynamics, and function using NetworkX,” in *Proc. 7th Python in Science Conference (SciPy 2008)*, 2008, pp. 11–15.
- [52] Team SimPy, “SimPy: Discrete event simulation for python, version 4.1,” 2024. [Online]. Available: <https://simpy.readthedocs.io>
- [53] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer, 2012.
- [54] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian, “The promises and perils of mining GitHub,” in *Proc. 11th Working Conference on Mining Software Repositories (MSR)*, 2014, pp. 92–101.
- [55] P. Runeson and M. Höst, “Guidelines for conducting and reporting case study research in software engineering,” *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009.
- [56] A. Naik, J. Culligan, Y. Gal, P. Torr, R. Aljundi, A. Paren, and A. Bibi, “OMNI-LEAK: Orchestrator multi-agent network induced data leakage,” arXiv preprint arXiv:2602.13477, 2026. [Online]. Available: <https://arxiv.org/abs/2602.13477>