

Article

From Collaborative Logistics Theory to Implementation: A Multi-Organizational Hyperledger Fabric Network for Vertical and Horizontal Collaboration

Yousra Chabba , Moulay Ali El Oualidi *  and Mustapha Ahlaqqach 

Laboratory of Advanced Research in Industrial & Logistic Engineering, Ecole Nationale Supérieure d'Electricité et de Mécanique, Hassan II University, Casablanca 20230, Morocco; yousra.chabba@ensem.ac.ma (Y.C.)

* Correspondence: a.eloualidi@ensem.ac.ma

Abstract

Collaborative logistics strategies—both vertical supply-chain coordination and horizontal competitor cooperation—can improve resource utilization, service quality, and supply-chain resilience through cooperation among supply-chain actors. However, their implementation remains constrained by long-standing concerns regarding data transparency, confidentiality, and trust, while many blockchain-based proposals remain largely conceptual and lack empirical validation. This study addresses this gap by presenting the design, implementation, and evaluation of a multi-organizational Hyperledger Fabric network supporting both vertical and horizontal collaborative logistics interactions. The proposed network models independent factories, warehouses, and clients through separate organizations connected across dedicated collaboration channels and protected by Private Data Collections for confidential information exchange. The implementation was deployed on Kubernetes and evaluated through functional validation, privacy and governance tests, and Hyperledger Caliper performance benchmarks comprising more than 3000 transactions. Results demonstrate successful execution of collaborative logistics workflows, enforcement of access-control and confidentiality requirements, and quantification of performance under varying workloads. By providing a reproducible implementation and empirical evaluation, this paper contributes practical evidence on how blockchain technology can support collaborative logistics strategies beyond conceptual frameworks.

Keywords: blockchain; logistics; vertical collaboration; horizontal collaboration; Hyperledger Fabric; multi-organizational networks



Academic Editor: Carlo Blundo

Received: 6 June 2026

Revised: 9 July 2026

Accepted: 20 July 2026

Published: 23 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Modern supply chains increasingly operate as networks of legally independent organizations whose performance depends not only on internal efficiency, but also on their ability to coordinate activities across organizational boundaries. As logistics ecosystems become more interconnected and dynamic, collaboration among manufacturers, warehouses, carriers, distributors, and customers evolves from an operational choice into a strategic necessity for improving resilience, responsiveness, and resource utilization. Yet, achieving such collaboration remains challenging because coordination requires information sharing and collective governance among actors that often possess divergent objectives and asymmetric incentives. This study investigates whether permissioned blockchain technology can provide a digital infrastructure capable of supporting inter-organizational logistics

collaboration while preserving organizational autonomy and selective information disclosure. To this end, the paper designs, implements, and evaluates a multi-organizational Hyperledger Fabric network as a proof-of-concept environment for blockchain-enabled collaborative logistics governance.

1.1. Multi-Organizational Logistics Collaboration: Types and Challenges

Global logistics networks increasingly rely on coordination among autonomous and independent partner organizations pursuing sustainable advantages despite their differences. Across supply chains, from raw materials suppliers to manufacturers, warehouses, carriers and retailers, actors must all synchronize their information, material, and financial flows across each one's organizational boundaries, often without a central authority to govern the interactions. Within this setting, logistics collaboration emerges not only as a central coordination mechanism, but more as a strategic mechanism to enhance organizations' performances, efficiency and resilience collectively in an increasingly competitive market characterized by ever more complex supply networks [1,2]. In this context, two collaboration types are recognized as being the fundamental forms that prevail among others, with each facing distinct governance challenges that traditional digital infrastructures fail to surmount. These two logistic collaboration forms, as complementary as they can be problematic, are the vertical collaboration across supply-chain tiers and the horizontal collaboration among peer competitors: a distinction that is most relevant in large supply networks and multi-partner logistics systems where organizations remain both legally and strategically independent.

In practice, vertical collaboration involves coordination among actors across sequential supply-chain stages: manufacturing to distribution to retail. Factories ship products to warehouses, which fulfill orders for clients, creating interdependencies where each tier's performance affects downstream partners. Despite widespread adoption of Enterprise Resource Planning (ERP) and Electronic Data Interchange (EDI) (which primarily optimize intra-firm processes), organizations typically maintain isolated information systems, generating persistent information asymmetry that requires manual reconciliation [3]. Periodic record matching consumes administrative resources, delays decisions, and creates disputes over quantity, quality, or delivery timing—a burden that grows with the number of partners involved. From a governance standpoint, power imbalances further complicate vertical coordination, where generally dominant manufacturers often dictate terms to smaller distributors. Commonly, information flows asymmetrically: upstream actors withhold strategic data while demanding downstream transparency, undermining trust and collaborative planning. This imbalance is structural rather than incidental. The resulting coordination costs, from periodic reconciliation, dispute resolution, and trust-building mechanisms such as audits and third-party verification, represent inefficiencies that erode supply-chain performance, particularly in large multi-organizational settings.

Horizontal collaboration, by contrast, involves coordination among stakeholders that act at the same level of the supply chain. Such coordination may take the form of: factories sharing production capacity, warehouses pooling inventory, or clients aggregating purchasing power. Research demonstrates substantial potential benefits including logistics cost reductions of 15–30% through capacity optimization, inventory balancing, and joint transportation [4], yet horizontal collaboration remains largely unrealized in practice. This theory–practice gap is persistent. The primary barrier is confidentiality risk, which is not only organizational, but also infrastructural. Logistic actors could pool their resources to realize mutual benefits, but both fear exposing sensitive data (mainly financial), thus revealing competitive intelligence and creating strategic vulnerability. In most current deployments, traditional IT systems offer only binary visibility (all-or-nothing access

control): granting full database access (risking competitive exposure) or maintaining complete information siloing (preventing coordination). Furthermore, in the absence of a shared enforcement mechanism, horizontal collaboration is difficult to exert because competing firms fear opportunistic behavior, where some partners free-ride instead of contributing. Consistent with Pomponi et al. [5], trust emerges as a critical barrier. Without neutral enforcement, cooperation collapses when self-interest dominates. Traditional contracts rely on slow, costly legal dispute resolution, discouraging firms from initiating collaboration, particularly for small and medium-sized enterprises (SMEs).

In their current form, traditional digital systems cannot resolve the core tension in multi-organizational logistics, where vertical collaboration needs transparency to reduce information asymmetry, while horizontal collaboration among competitors requires confidentiality. From an infrastructural standpoint, existing IT architectures force a binary choice between full transparency, thus exposing strategic information, or total opacity, which prevents coordination. ERP, EDI, Internet of Things (IoT), and Artificial Intelligence (AI) tools facilitate data exchange but were not designed to enforce multi-party agreements, prevent unilateral manipulation, or provide selective transparency (i.e., role-dependent data visibility). As a result, horizontal collaboration largely fails to materialize beyond high-trust partnerships with costly legal frameworks, particularly affecting SMEs. The economic consequences are significant, with substantial economic losses arising from redundant inventory, unused capacity, and avoidable stockouts.

These limitations motivate the central question of this study: “Can technology provide a digital environment that enables both transparency and confidentiality simultaneously to truly enable multi-organizational coordination?”

1.2. Blockchain Technology as Enabling Solution

Addressing the aforementioned tension requires a digital infrastructure capable of enforcing coordination rules while preserving organizational autonomy. In this context, blockchain technology has been positioned in the literature as a strong candidate infrastructure for multi-organizational coordination.

Since Nakamoto’s introduction of blockchain as a decentralized consensus protocol for peer-to-peer transactions [6], the technology has evolved into a promising digital infrastructure offering architectural capabilities that potentially address both vertical and horizontal collaboration challenges through protocol-enforced governance mechanisms. Unlike traditional digital infrastructures that primarily facilitate information exchange, blockchain provides cryptographic enforcement of coordination rules [3].

At the infrastructural level, blockchain relies on distributed ledger technology that replicates synchronized transaction records across all authorized participants, thereby reducing information asymmetry and establishing a shared source of truth for inter-firm operations [7,8]. Because transactions are cryptographically time-stamped and immutable, reconciliation, in principle, becomes automatic rather than manual, an essential improvement for vertical supply-chain coordination [9]. At the same time, organizational autonomy and accountability are preserved through independent cryptographic identities. In permissioned blockchain networks such as Hyperledger Fabric, Membership Service Providers (MSPs) manage identity verification and access control without introducing centralized ownership, while multi-party endorsement policies require transaction validation by relevant organizations, effectively preventing unilateral ledger manipulation [7,10,11]. Governance is thus embedded *ex ante* within the protocol rather than enforced *ex post* through audits or dispute resolution [3]. This design shifts coordination logic from organizational discretion to protocol-defined rules. Beyond transparency and integrity, blockchain architectures also enable selective visibility. Modular channel structures restrict transaction access

to defined participant subsets, while Private Data Collections (PDCs) allow organizations to share operational coordination data without disclosing strategically sensitive information. Competing actors observe cryptographic proofs instead of raw values, theoretically enabling horizontal coordination without competitive exposure [7,12].

Taken together, these mechanisms suggest that blockchain can deliver transparency for vertical coordination and confidentiality for horizontal collaboration, shifting coordination from trust-based to protocol-enforced governance [13,14]. Yet, while powerful, these mechanisms introduce configuration complexity and require careful governance definition.

1.3. The Implementation Gap: Related Systems and Remaining Challenges

Blockchain-based logistics collaboration has attracted extensive theoretical attention since 2016, resulting in a large body of conceptual frameworks, architectural proposals, and analytical models outlining potential benefits for inter-organizational coordination [3,9]. In parallel, major logistics actors such as Maersk, DHL, and IBM have announced pilot initiatives promising enhanced visibility and data integrity across supply networks [15,16]. Despite this growing interest, operational and empirically verifiable evidence remains largely absent. To our knowledge, existing studies remain predominantly conceptual [13,14], analytical [17], or focused on adoption barriers and organizational readiness [18] without developing or validating operational blockchain artifacts. Industry pilots are typically reported through announcements rather than technical documentation, while proof-of-concept implementations remain proprietary, preventing independent verification or replication.

To situate this study within the existing landscape, prior blockchain implementations for supply-chain and logistics contexts are reviewed across four categories. Among industry platforms, TradeLens (Maersk/IBM) applied Hyperledger Fabric to global shipping documentation, achieving adoption among major ocean carriers and more than 270 port terminals before its discontinuation [15,16] and IBM Food Trust deployed Fabric for food traceability [19]. Both address vertical transparency with no mechanism for horizontal confidentiality—channel data is visible to all authorized participants. Among conceptual reference architectures, Wang et al. review 62 studies concluding most remain at the framework level [9], Treiblmaier maps blockchain properties to supply-chain outcomes without implementation [14], and Dutta et al. survey 104 papers identifying horizontal collaboration as a gap without providing implementation evidence [3]. Among simulation-based prototypes, Helo & Hao deploy a two-organization Fabric network testing only vertical transactions without PDC, load benchmarking, or adversarial testing—the closest prior work of this earlier generation [8]. Li et al. demonstrate cross-organization logistics coordination without competitor-level privacy [12], and Chang et al. analyze vertical trade finance processes without horizontal collaboration. Among domain-specific systems [20], Choi proposes diamond authentication with no competitive collaboration scenario [17]. Table 1 synthesizes these comparisons.

Table 1. Comparison of supply chain blockchain systems.

System/Study	Scope	Privacy Mechanism	Deployment Model	Evaluation Depth	Public Artifact
TradeLens [15,16]	Global vertical shipping; documentation	Channel-based (Coarse isolation)	Enterprise Cloud (Proprietary HLF)	Anecdotal: Adoption and port efficiency metrics	✗
IBM Food Trust [19]	Vertical farm-to-retail; provenance	Channel-based (Membership only)	IBM Managed Cloud (SaaS)	Anecdotal: Trace-time reduction logs	✗
Helo & Hao [8]	Vertical buyer-supplier operations	None: Single-channel (Full visibility)	Simulated (2-node local prototype)	Functional: System walkthrough/ logic test	✗

Table 1. Cont.

System/Study	Scope	Privacy Mechanism	Deployment Model	Evaluation Depth	Public Artifact
Li et al. [12]	Cross-org logistics resource sharing	Channel-level (Shared visibility)	Unspecified Fabric prototype	Functional: Coordination logic (No stress test)	✗
Chang et al. [20]	Vertical trade finance (Letters of Credit)	Logic-based: ABAC in chaincode	Conceptual (Case study analysis)	Conceptual: Algorithmic validation only	✗
Agrawal et al. [21]	Vertical textile traceability	None: Ledger-wide (Public to members)	Permissioned PoC (Centralized)	Functional: Proof-of-concept transactions	✗
This Study	Vertical + Horizontal (Competitor Coordination)	PDC-based (Field-level isolation)	Multi-Org: 15-namespace/12 MSPs single-cluster K8s	Empirical: 3000+ tx Caliper benchmarks + Adversarial testing	✓ (GitHub (v3.5.6) + Zenodo)

The studies above are foundational but they predate 2021; horizontal-collaboration-capable blockchain implementations may plausibly have emerged more recently. To verify whether the identified gap persists in the current literature and is based on the systematic review methodology in a logistic context [22], a structured search was additionally conducted across four major scientific databases—Scopus (via ScienceDirect), Web of Science (Clarivate), IEEE Xplore, and MDPI—combining the keywords (“blockchain” OR “Hyperledger Fabric” OR “Private Data Collections”) AND (“horizontal collaboration” OR “logistic collaboration” OR “vertical collaboration” OR “supply chain”), restricted to peer-reviewed publications from January 2021 to June 2026. Candidate studies were screened first by title and abstract relevance, then by full-text assessment, and retained if they presented a blockchain application in a logistics or supply-chain context with sufficient implementation detail for architectural comparison. Each retained study was evaluated against three questions:

- (Q1) Does it present an implemented blockchain artifact rather than a conceptual proposal?
- (Q2) Does it support horizontal collaboration among competing organizations rather than only vertical coordination?
- (Q3) Does it implement and evaluate a fine-grained confidentiality mechanism—PDC, Zero-Knowledge Proofs, or Multi-Party Computation?

Table 2 summarizes the 19 retained studies against these criteria.

Table 2. Structured literature review results (2021–2026).

Study	Operational Artifact	Vertical Collaboration	Horizontal Collaboration	Confidentiality Mechanism	Empirical Evaluation
Cho et al. (2026) [23]	✓	✓	✗	Organization-specific PDCs	✓
Ravi et al. (2022) [24]	✓	✓	✗	Private channels	✓
Neto et al. (2026) [25]	✓	✓	✗	Dual public/private channels	✓
Rahaman et al. (2024) [26]	✓	✓	✗	Access-control policies	✓
Keo et al. (2025) [27]	✓	✓	✗	No implemented confidentiality mechanism (metadata discussion only)	✓
Ni & Irannezhad (2024) [28]	✓	✓	✗	None	✓
Zhou et al. (2026) [29]	✓	✓	✗	PDC discussed (not competitor-specific)	✓
Duman & Aydoğan (2025) [30]	✓	✓	✗	Role-based access control	✓

Table 2. Cont.

Study	Operational Artifact	Vertical Collaboration	Horizontal Collaboration	Confidentiality Mechanism	Empirical Evaluation
Sayar et al. (2025) [31]	✓	✓	✗	MSP/CA authentication	✓
Ismail et al. (2026) [32]	✓	✓	✗	None	✓
Kutybayeva et al. (2025) [33]	Partial	✓	✗	None	Limited
Naga Sudha et al. (2024) [34]	✓	✓	✗	None	✓
Kamal Idrissi et al. (2024) [35]	✓	✓	✗	None	✓
Qiao et al. (2025) [36]	Partial	✓	✗	None	Limited
Chou et al. (2023) [37]	✓	✓	Partial	Multichain/ channel isolation	✓
This study	✓	✓	✓	Organization-specific PDC	Functional validation + adversarial testing + Caliper benchmarking (>3000 transactions)

The structured review indicates that the gap identified in the foundational literature persists in recent operational implementations: implemented systems from this more recent period remain concentrated on vertical coordination—traceability, provenance, food safety, and document management—with confidentiality, where present at all, typically achieved through access-control policies, private channels, or coarse organization-level partitioning rather than field-level isolation. Two studies merit individual attention as the closest comparators on each axis. Chou et al. [37] is closest in scope, incorporating horizontal coordination within a dynamic partnership, but its privacy boundary operates at the multichain/channel level rather than through organization-specific PDC [37]. Cho et al. [23] is closest in mechanism, implementing genuine per-organization PDC isolation with empirical latency and throughput evaluation, but its two-organization scenario is an explicitly vertical supplier–consumer relationship rather than a horizontal, competing one [23]. No study identified in either Table 1 or Table 2 combines both properties together with empirical governance validation.

Three characteristics distinguish the present study from all systems identified in Tables 1 and 2. First, it integrates both vertical and horizontal collaboration within a single operational Hyperledger Fabric consortium involving competing organizations. Second, it combines this collaboration model with organization-specific PDC that preserve commercially sensitive information through empirically verified field-level isolation. Third, it complements the architectural implementation with functional validation, adversarial confidentiality testing, and automated performance benchmarking comprising more than 3000 executed transactions. Accordingly, to the best of our knowledge, based on the structured review described above, no operational Hyperledger Fabric implementation was identified that simultaneously demonstrates vertical collaboration, horizontal collaboration among competing organizations, and confidentiality-preserving coordination through organization-specific PDC with empirical validation. Finally, unlike the implementation studies identified in the structured review, all artifacts developed in the present study are publicly released through GitHub and Zenodo to enable independent replication and future comparative research.

1.4. Problem Statement and Research Questions

Building on the identified theory–practice gap, this study addresses a twofold problem. First, multi-organizational logistics collaboration demands a digital infrastructure capable of simultaneously enforcing transparency for vertical coordination and confidentiality for horizontal collaboration—requirements that are structurally contradictory under

traditional architectures offering either full data sharing or complete isolation. Second, although blockchain has been proposed as a solution, the absence of operational implementations—as demonstrated in the preceding review—means that blockchain’s governance capabilities remain theoretically asserted but empirically unverified, and no replicable methodology exists for independent evaluation.

Accordingly, this study investigates the technical feasibility of blockchain-enabled logistics collaboration through three interrelated research questions, each addressing a distinct but complementary dimension of implementation and governance.

- RQ1: How can a blockchain network architecture simultaneously support selective transparency and organizational independence for both vertical and horizontal collaborative logistics interactions?
- RQ2: To what extent can organization-specific PDC configurations preserve competitive confidentiality while enabling horizontal logistics collaboration among rival actors?
- RQ3: What performance characteristics, including latency, throughput, and confidentiality-related overhead, emerge under collaborative logistics transaction workloads within a multi-organizational blockchain deployment?

To address these questions, this paper presents the design, implementation, and empirical evaluation of a Hyperledger Fabric-based collaborative logistics network deployed within a controlled Kubernetes environment. The study combines functional validation, governance and confidentiality testing, and performance benchmarking to evaluate the operational behavior of the proposed architecture under representative logistics collaboration scenarios.

1.5. Contributions

This study provides the following contributions:

1. A multi-organizational collaborative logistics network architecture comprising 15 Kubernetes namespaces, 12 independent MSPs, and five dedicated collaboration channels supporting both vertical and horizontal logistics interactions among factories, warehouses, and clients.
2. A confidentiality-preserving collaboration design combining MSP-level organizational separation, channel-based governance, and twelve organization-specific PDC to support selective information sharing while maintaining competitive confidentiality among rival actors.
3. A publicly reproducible implementation artifact including chaincode, Kubernetes deployment manifests, channel and PDC configurations, benchmark scripts, and deployment documentation released through a public repository and archived research artifact (GitHub + Zenodo DOI).
4. A comprehensive empirical evaluation comprising 175 functional validation transactions and 3000 Hyperledger Caliper benchmark transactions executed across 12 controlled workload scenarios to assess governance enforcement, confidentiality preservation, latency, throughput, and resource utilization.
5. A controlled analysis of confidentiality-related performance implications, including comparative benchmark scenarios designed to isolate and evaluate the operational impact of PDC-based information protection mechanisms within collaborative logistics workflows.

1.6. Paper Organization

Guided by the Design Science Research (DSR) methodology [38], this study follows a structured process of artifact design, implementation, demonstration, and evaluation.

Section 2 presents the technical foundation of the research, detailing the system architecture, the deployment phases, and the key configuration and design choices underlying the multi-organizational blockchain network. Section 3 reports the observable outcomes of the operational deployment, including performance metrics, governance-related evidence, and transaction-level verification results. Building on these observations, Section 4 discusses the principal technical insights derived from the implementation, highlighting architectural trade-offs, configuration challenges, and considerations for replication in other contexts. Finally, Section 5 concludes the paper by synthesizing the main findings, outlining the study's contributions, acknowledging its limitations, and identifying future research directions in blockchain-enabled collaborative logistics.

2. Implementation: From Use Case to Operational Network

2.1. Study Case: UK Distribution Network

This implementation follows Design Science Research (DSR) methodology, which addresses practical problems through artifact construction and evaluation [38,39]. DSR is appropriate when existing solutions prove inadequate—as established in Section 1, traditional coordination technologies facilitate but cannot enforce multi-organizational governance. The DSR process for this study follows the six nominal activities of the Design Science Research Methodology (DSRM) proposed by Peffers et al. [38]. Table 3 maps each activity to its corresponding section and evidence. Consistent with Peffers et al.'s characterization of DSRM as iterative rather than strictly linear, the design-and-development activity was itself revisited across four platform iterations before the architecture reported here was reached (Section 3.1), with each iteration's evaluation against Kubernetes networking and Transport Layer Security (TLS) handshake failures feeding back into a revised design—an explicit design–evaluate loop rather than a single build-and-report cycle.

Table 3. DSRM process mapping.

DSRM Activity	Realized in	Evidence
1. Problem identification and motivation	Sections 1.1–1.3	Documented theory–practice gap: vertical transparency vs. horizontal confidentiality tension; structured review of prior implementations (Tables 1 and 2)
2. Definition of objectives for a solution	Section 1.4 (RQ1–RQ3), Section 1.5 (Contributions)	Explicit, falsifiable research questions targeting architecture, confidentiality preservation, and performance characteristics
3. Design and development	Section 2 (entire); iterated per Section 3.1	15-namespace, 12-MSP, 5-channel architecture; four deployment iterations, each evaluated and fed back into redesign before convergence
4. Demonstration	Sections 3.2–3.4 (V1–V4, AV1–AV3)	Artifact executed against six vertical shipment lifecycles and horizontal capacity-sharing, inventory-pooling, and joint-purchasing scenarios
5. Evaluation	Section 3.4.3 (P1a–P1c, G1, INT1), Section 3.5, Section 3.6	Adversarial PDC-boundary testing, governance-violation attempts, Caliper performance benchmarking, and comparison against a centralized RBAC baseline—assessing whether the stated objectives (Activity 2) were actually met, not merely whether the artifact runs
6. Communication	This manuscript; GitHub + Zenodo artifact	Full chaincode, deployment manifests, and benchmark configurations publicly archived for independent replication

2.1.1. Network Topology and Business Context

This study adapts the UK distribution topology from [40] to simulate a multi-organizational network across six regional hubs, as illustrated in Figure 1. While the geographic layout and general tier structure are drawn from an established reference, the organizational entities, transaction parameters, and collaboration scenarios are synthetically defined for this study and do not represent an operational deployment with real logistics firms, live shipment data, or commercial contractual constraints. The manufacturing tier comprises two competing factories—FAC-LIV (Liverpool) and FAC-BRI (Brighton)—each operating with distinct production capacities and overlapping customer markets, making them ideal candidates for horizontal capacity-sharing evaluation. The distribution tier consists of four independent warehouse operators—WH-NEW (Newcastle), WH-BIR (Birmingham), WH-LON (London), and WH-EXE (Exeter)—with varying storage capacities and regional specialization. The retail tier includes six client organizations (CLI-1 through CLI-6) distributed across the network’s regions, generating varied order volumes and delivery schedules. Each of these twelve entities is implemented as a fully independent organization with its own Kubernetes namespace, CA, MSP, peer node, and CouchDB state database, as summarized in Table 4.

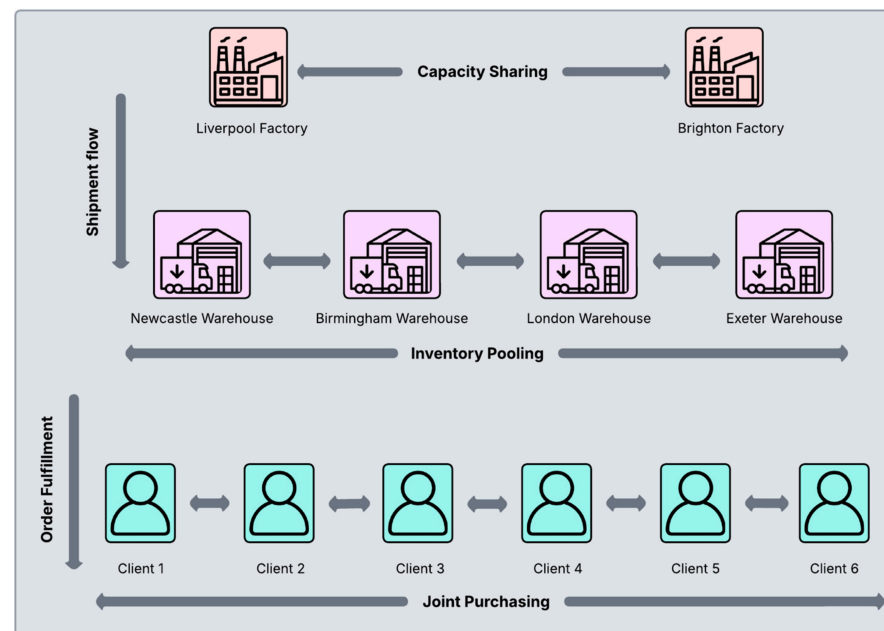


Figure 1. Network design topology.

Table 4. Organizational architecture.

Organization	MSP	Namespace	CA	Role
FAC-LIV	FacLivMSP	org-fac-liv	ca.facliv	Factory
FAC-BRI	FacBriMSP	org-fac-bri	ca.facbri	Factory
WH-NEW	WhNewMSP	org-wh-new	ca.whnew	Warehouse
WH-BIR	WhBirMSP	org-wh-bir	ca.whbir	Warehouse
WH-LON	WhLonMSP	org-wh-lon	ca.whlon	Warehouse
WH-EXE	WhExeMSP	org-wh-exe	ca.whexe	Warehouse
CLI-1 to CLI-6	Cli1MSP–Cli6MSP	org-cli-1 to org-cli-6	ca.cli1–ca.cli6	Client
—	AdminMSP	org-admin	ca.admin	Network Governance
—	OrdererMSP	org-orderer	ca.orderer	Consensus service
—	—	org-chaincode	—	Chaincode execution

This topology supports vertical flows (factory → warehouse → client) and horizontal collaboration (factory ↔ factory, warehouse ↔ warehouse, client ↔ client), creating a suitable environment for evaluating blockchain governance mechanisms across both coordination modes within a single deployment.

2.1.2. Stakeholders, Roles, and Collaboration Objectives

Vertical collaboration involves sequential coordination across tiers. Factory–Warehouse interactions include shipment initiation, receipt confirmation, and inventory synchronization, with the shared objective of maintaining accurate stock records and eliminating reconciliation overhead. Warehouse–Client interactions cover order processing, delivery scheduling, and confirmation, with the goal of timely fulfillment and dispute prevention through verified delivery records.

Horizontal collaboration addresses competitor–competitor interactions within the same tier. Factory–Factory coordination enables capacity sharing during demand surges while preserving confidentiality regarding unit costs. Warehouse–Warehouse coordination supports inventory pooling to prevent regional stockouts. Client–Client coordination enables joint purchasing to secure volume discounts while protecting individual budget structures.

Across all tiers, the core governance requirement is that vertical collaboration demands transparency while horizontal collaboration demands confidentiality. The blockchain architecture must satisfy both requirements simultaneously within a single network.

2.2. Platform Selection: Hyperledger Fabric

2.2.1. Permissioned vs. Public Blockchain

Selecting an appropriate blockchain platform is a foundational architectural decision. Public blockchains such as Bitcoin and Ethereum offer open participation, pseudonymous identities, full transparency, and fee-based transaction processing—characteristics that conflict fundamentally with logistics collaboration requirements where business confidentiality, organizational accountability, controlled membership, and predictable operational costs are non-negotiable [6]. Permissioned blockchains provide controlled membership, where consortium participants are vetted before joining, cryptographic identities are mapped to real-world entities, and there is selective transparency through channel architecture and energy-efficient consensus without transaction fees [7]. The choice of a permissioned network over public blockchain alternatives—such as Ethereum combined with IPFS for off-chain storage—reflects this context: participating organizations are known entities requiring identity-verified accountability and selective data visibility, properties native to permissioned architectures but absent from pseudonymous public networks. Additionally, Fabric imposes no per-transaction gas fees, making on-chain storage of structured logistics records operationally cost-free.

2.2.2. Hyperledger Fabric Choice

Hyperledger Fabric was selected because its core mechanisms map directly to the dual collaboration requirements established in Section 1. Four capabilities drove this choice. First, MSPs enable each organization to maintain an independent cryptographic identity rooted in its own CA, supporting genuine organizational separation within a shared network. Second, PDC allow field-level data partitioning within a channel, enabling competitors to share coordination data publicly while keeping strategic data disseminated only to authorized MSP peers. Third, multi-party endorsement policies embed governance rules at the protocol layer, requiring cryptographic signatures from specified organizations before any transaction reaches the orderer. Fourth, CouchDB as a state database supports the rich JSON queries required by logistics chaincode—filtering shipments by status, querying

inventory by warehouse, and retrieving orders by client—which LevelDB’s key-only lookup model does not support. HLF 2.4.7 LTS was used as the stable release version at the time of deployment.

2.3. Architectural Choice: Namespace-Based Network

2.3.1. Hardware Constraints and Architecture Design

This study operates under hardware constraints representative of academic research environments and SME pilot projects. The deployment hardware comprises a professional laptop with an AMD Ryzen 7 Pro CPU (2.6 GHz), 32 GB RAM, and a Windows 11 Pro system, with 16 GB RAM and 8 CPU cores allocated to Kubernetes. Typical blockchain deployments assume cloud infrastructure or multi-cluster architectures for organizational isolation, creating barriers for resource-constrained actors. The multi-namespace-based architecture addresses this by achieving genuine organizational independence within a single-cluster resource envelope. The final architecture was reached after four deployment iterations—detailed in Section 3.1—whose lessons informed the Rancher Desktop-based design adopted here.

2.3.2. Technology Decisions

Rancher Desktop (dockerd/moby runtime) was selected for its stable Windows-native networking, avoiding the Domain Name System (DNS) resolution and TLS handshake failures encountered with WSL2-based alternatives. HLF 2.4.7 LTS delivered required capabilities—channels, PDC, endorsement policies—with per-component resource profiles of approximately 1.8 GB per peer, 250 MB per CA, 2.1 GB for the orderer, and 1.5 GB per CouchDB instance. CouchDB was selected over LevelDB because the logistics chaincode requires rich JSON queries that LevelDB’s key-only lookup model cannot support.

2.4. Network Design

2.4.1. Network Architecture and Channel Design

The network is deployed across 15 Kubernetes namespaces within a single cluster: org-admin, org-orderer, org-chaincode, org-fac-liv, org-fac-bri, org-wh-new, org-wh-bir, org-wh-lon, org-wh-exe, and org-cli-1 through org-cli-6, as depicted in Figure 2. Each operational namespace contains an independent CA, MSP configuration, peer pod, and CouchDB instance. Namespace-level NetworkPolicies enforce default-deny traffic isolation, with explicit rules allowing cross-namespace communication required by channel gossip and transaction endorsement flows.

The critical architectural property is MSP independence: because each organization has its own root CA, collection policies of the form OR(‘FacLivMSP.member’) are exclusively satisfiable by peers holding certificates signed by FacLivMSP’s CA. No other organization’s peer is eligible to receive collection data under this policy regardless of chaincode logic—enforcement occurs at Fabric’s gossip-based PDC dissemination layer before any application code executes. This property distinguishes the present architecture from designs where competing entities share a single MSP: in such configurations, PDC policies grant all entities within that MSP access to the same collections, defeating confidentiality between competitors. Cross-namespace communication uses Kubernetes DNS service names throughout, ensuring deterministic peer discovery across all 15 namespaces.

Five dedicated channels structure the collaboration topology, illustrated in Figure 2, and Table 5 maps each channel to its collaboration type and representative functions.

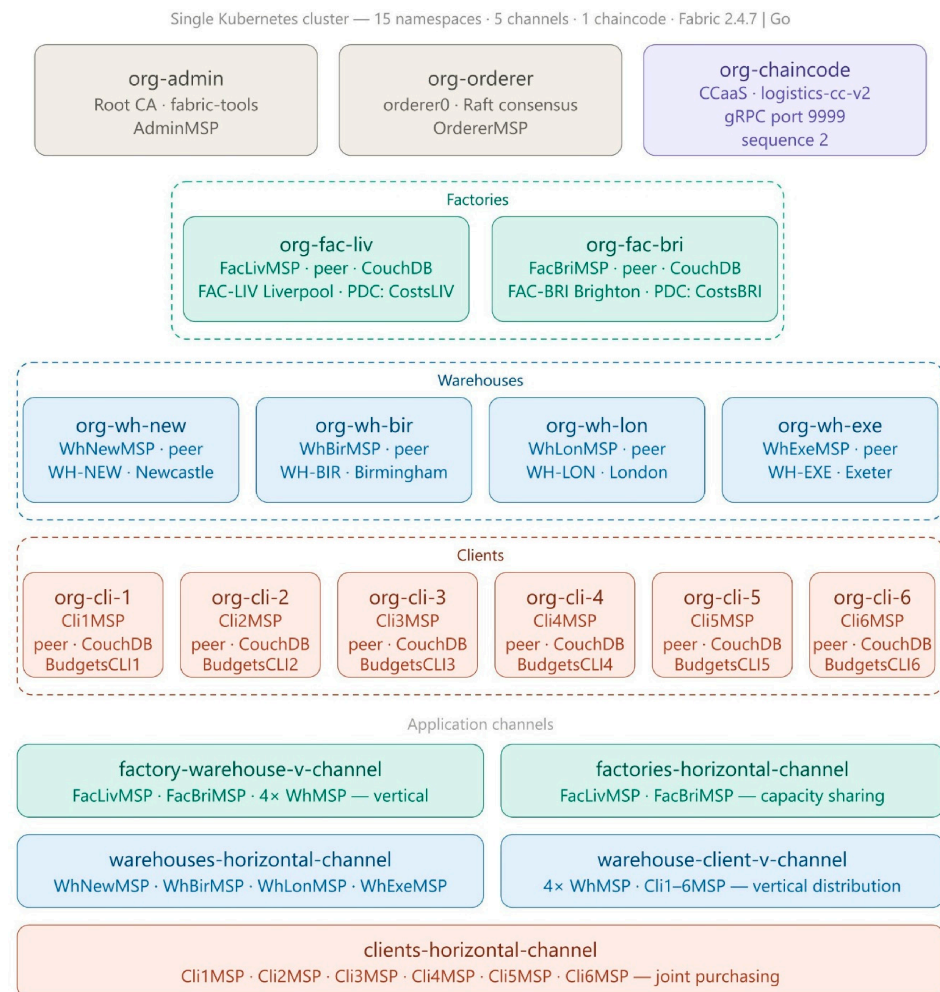


Figure 2. Network architecture.

Table 5. Channel Architecture and Collaboration Mapping.

Channel	Collaboration Type	Member MSPs	Representative Functions
Factory-warehouse-v-channel	Vertical	FacLivMSP, FacBriMSP, 4 × WhMSP	InitiateShipment, ConfirmReceipt
Warehouse-client-v-channel	Vertical	4 × WhMSP, 6 × CliMSP	PlaceOrder, ConfirmDelivery
Factories-horizontal-channel	Horizontal	FacLivMSP, FacBriMSP	ProposeCapacityShare, StorePrivateCosts
Warehouses-horizontal-channel	Horizontal	WhNewMSP, WhBirMSP, WhLonMSP, WhExeMSP	RequestInventoryTransfer, StorePrivatePricing
Clients-horizontal-channel	Horizontal	Cli1MSP–Cli6MSP	ProposeJointOrder, StorePrivateBudget

Governance is enforced through three complementary layers, with the endorsement policy held deliberately uniform across them. First, channel membership defines the outermost boundary: an organization not listed as a member of a channel cannot submit a transaction proposal to it at all, as confirmed experimentally by cross-tier attempts returning a peer-layer “channel not found” response (Section 3.3.3)—a stronger guarantee than endorsement rejection, since the proposal never reaches evaluation. Second, chaincode-level identity checks enforce function-level authorization: each function retrieves the

invoking organization's MSP identity through the client-identity API and rejects callers whose MSP does not match the operation's authorized role, so that, for example, only the organization owning a factory record may modify it. Third, PDC policies bind each confidential collection to a single owning MSP—OR('FacLivMSP.member') for CostsLIV, OR('FacBriMSP.member') for CostsBRI, and so on—restricting private-data dissemination to authorized peers at Fabric's gossip layer (Section 2.4.2). The chaincode itself is committed under a single explicit endorsement policy satisfiable by any one of the twelve organizational MSPs, applied identically to every function on every channel. This uniform policy is an intentional design choice: it isolates the governance variables that the architecture actually relies on—channel membership for submission control and per-MSP collection policies for confidentiality—from endorsement, which is therefore held constant rather than used as a per-function access mechanism. Multi-party horizontal operations such as capacity sharing are consequently implemented as explicit two-step on-chain workflows—a proposal transaction followed by a separate counterparty transaction recorded as ledger state—rather than relying on co-signed endorsement, ensuring each party's approval is independently and immutably attributable.

2.4.2. Confidential Coordination via PDC

The dual collaboration model creates a fundamental data tension: competing organizations must exchange operational data to coordinate (available capacity, stock levels, order status) while protecting strategic data that defines their competitive advantage (unit production costs, profit margins, storage pricing, client budgets). Fabric's PDC mechanism resolves this by partitioning transaction data into two layers within the same channel. Public operational fields are committed to the shared channel ledger, visible to all channel members. Private strategic fields are stored exclusively in the side-databases of authorized peers, with only a SHA-256 hash committed to the public ledger, enabling any channel member to verify data existence and tamper-evidence without accessing the data itself—confirming integrity without zero-knowledge proofs.

Twelve organization-specific collections are configured, each scoped exclusively to one MSP. Table 6 lists the complete collection design.

Table 6. PDC design.

Collection	Policy	Stored Fields
CostsLIV	OR('FacLivMSP.member')	unitCost, margin
CostsBRI	OR('FacBriMSP.member')	unitCost, margin
PricingNEW	OR('WhNewMSP.member')	storageCost, markup
PricingBIR	OR('WhBirMSP.member')	storageCost, markup
PricingLON	OR('WhLonMSP.member')	storageCost, markup
PricingEXE	OR('WhExeMSP.member')	storageCost, markup
BudgetsCLI1–CLI6	OR('Cli[N]MSP.member')	maxPricePerUnit, totalBudget

Because each organization operates under an independent MSP with its own root CA, each collection policy is exclusively scoped to its designated organization. FAC-BRI's peers hold certificates signed by FacBriMSP's CA and cannot satisfy OR('FacLivMSP.member')—this restriction is enforced by Fabric's gossip dissemination protocol, not by chaincode conditional logic. The complete access control logic is formalized in Algorithm A2 (Appendix A.2).

2.4.3. Collaborative Workflow Functions

Vertical collaboration functions implement shipment creation, receipt confirmation, order processing, and delivery validation across the factory–warehouse–client lifecycle, as illustrated in Figure 3. Horizontal collaboration functions support capacity sharing

between competing factories, inventory transfers between warehouses, and joint purchasing coordination among clients. In each horizontal channel, confidentiality-sensitive attributes are managed through organization-specific PDCs while coordination metadata remains visible to all authorized channel participants (see Figure 3). The complete end-to-end lifecycle logic is formalized in Algorithm A1 (Appendix A.1).

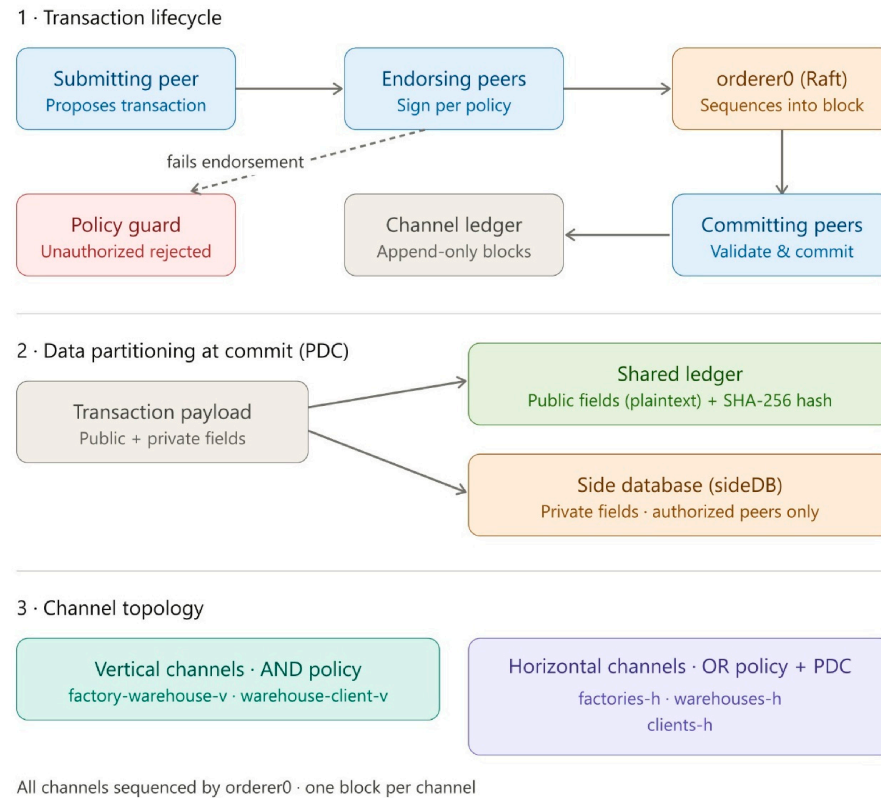


Figure 3. Transaction and data flow.

2.4.4. Threat Model and Isolation Boundaries

This deployment operates under a passive insider adversary model: consortium members are authorized network participants who follow the Fabric protocol but may attempt to access data beyond their PDC entitlement. Three protection layers address this threat class. At the Fabric protocol layer, PDC policies restrict private data dissemination via Fabric’s gossip protocol—access denial occurs before any chaincode execution, returning `privateDataAccessible:false` as a protocol-layer response. Unauthorized state modification is prevented not by endorsement—the chaincode is committed under a uniform policy satisfiable by any single member organization—but by chaincode-level identity checks that reject any caller whose MSP does not own the targeted record (returning `ACCESS_DENIED`), backed by the immutability of committed ledger state. At the transport layer, mutual TLS (mTLS) secures all peer-to-peer and peer-to-orderer communications. At the Kubernetes layer, namespace-level NetworkPolicies enforce default-deny traffic isolation. CA private keys were generated and stored within their respective organizational namespaces and were not shared across organizations during deployment or testing.

The passive insider model above defines the specific adversary class these mechanisms were evaluated against (Section 3.4.3); it is narrower than the adversary space relevant to an industrial consortium of competing organizations. Table 7 widens the threat catalog accordingly.

Table 7. Expanded adversary and threat catalog.

Adversary Class	Covered by This Study	Mitigation Present	Future Extension
Passive insider (unauthorized PDC/state access attempt)	Yes	PDC isolation, MSP identity checks, mTLS	—
Malicious cluster administrator	No	None in single-cluster deployment	VM or multi-cluster isolation
Compromised Kubernetes node (non-admin)	No	Namespace NetworkPolicies limit lateral movement	Node-level runtime security controls
Compromised certificate authority	No	Independent root CA per organization (Section 2.4.5)	HSMs, key rotation, revocation management (Section 4.5)
Malicious orderer operator	No	None in single-orderer deployment	Multi-organization ordering service (Section 4.5)
Malicious chaincode developer	Partially	Multi-organization chaincode approval required before commitment (Phase 8, Section 2.6.1)	Independent chaincode audit process
Colluding organizations	No	Not preventable through protocol-level access control	Governance and contractual mechanisms
Denial-of-service attacks	No	Not evaluated	Distributed deployment and infrastructure protections
Metadata inference attacks	Partial	PDC hides content, not transaction existence, timing, or size	Traffic shaping or batching—see below

Although PDC mechanisms prevent unauthorized organizations from accessing private field values, they do not conceal the existence, timing, or approximate size of transactions recorded on the shared ledger. Consequently, organizations participating in the same channel may observe transaction frequency or activity patterns associated with a competitor without gaining access to the underlying confidential values. For example, FAC-BRI could observe the frequency and approximate size of FAC-LIV's *InitiateShipment* or *StorePrivateCosts* transactions on the shared channel and treat this as a coarse proxy for FAC-LIV's shipment or order volume—even though the corresponding quantity, cost, and margin fields remain cryptographically inaccessible (Section 3.4.3). The present study does not evaluate how reliably such metadata could support this kind of volume inference, nor does it quantify the resulting information leakage. Mitigation strategies such as transaction batching, fixed publication intervals, or traffic padding may reduce this leakage but were considered outside the scope of the present proof of concept.

Table 8 compares the isolation properties of the namespace-based architecture against VM or multi-cluster alternatives, directly addressing the single-cluster isolation question.

This isolation level is appropriate for the study's scope: a trusted research consortium validating governance mechanisms. All governance claims—PDC confidentiality, endorsement enforcement, and ledger integrity—are enforced at the Fabric protocol layer, which operates independently of the Kubernetes isolation layer. Encryption at rest for CouchDB is not configured. The deployment does not claim protection against adversary classes beyond the passive insider model described above; the additional adversary categories relevant to industrial blockchain consortia and their possible mitigations are summarized in Table 7.

Table 8. Namespace vs. VM isolation comparison.

Property	Namespace Isolation (This Study)	VM/Multi-Cluster Isolation
Boundary type	Logical (NetworkPolicy + RBAC)	Hypervisor or physical
Shared control plane	Yes (single etcd, single API server)	No
Cross-org secret access	Possible for cluster-admin	Not possible
Fabric protocol isolation	Full (independent CA, MSP, TLS per org)	Full
PDC access-control enforcement	Full	Full
Suitable for adversarial production	No	Yes
Suitable for trusted research consortium	Yes	Yes

2.4.5. Key Management and TLS Configuration

The network employs 12 independent root CAs—one per operational organization—implemented with Hyperledger Fabric CA v1.5.5, deployed within each organization’s dedicated namespace. Unlike the prior network design which used a single root CA with per-organization intermediate CAs, the current architecture assigns each organization its own root CA, ensuring certificate chains are entirely independent and no cross-organizational trust is assumed at the CA level. Each CA issues operational certificates exclusively for its organization’s peer node and admin identity. A dedicated TLS CA per organization issues transport-layer certificates separately from identity certificates, maintaining separation between identity trust and transport trust chains. All peer-to-peer and peer-to-orderer gRPC communication uses mutual TLS, with certificates mounted into containers via Kubernetes Secrets. A cross-organization TLS CA bundle enables each peer to validate certificates issued by other organizations’ TLS CAs.

Certificate rotation, Certificate Revocation Lists (CRL), Hardware Security Module (HSM) integration, and encryption at rest were not implemented in this proof of concept and are acknowledged as production hardening requirements.

2.5. Infrastructure Configuration

Table 9 provides the complete infrastructure and network configuration parameters, addressing the configuration detail requirements.

Table 9. Infrastructure and network configuration.

Parameter	Value
Hardware	Single laptop (Lenovo ThinkPad L15)
CPU allocated to Kubernetes	8 cores (AMD Ryzen 7 Pro, 2.6 GHz)
RAM allocated to Kubernetes	16 GB
Kubernetes distribution	Rancher Desktop v1.20.0 (dockerd/moby)
Hyperledger Fabric version	2.4.7 LTS
Fabric CA version	1.5.5
State database	CouchDB 3.2.1 (per organization)
Orderer consensus	Raft (single-node)
Block size (MaxMessageCount)	10 transactions
Batch timeout	2 s
Namespaces	15
Independent MSPs	12
Channels	5
Chaincode deployment	CCaaS (Chaincode as a Service)
PDCs	12
Caliper version	0.6.0
Chaincode language	Go

2.6. Deployment Process

2.6.1. Deployment Phases

Translating the architecture into an operational network required systematic progression through nine sequential phases, each verified before proceeding. The deployment philosophy emphasizes verification at each phase, complete documentation enabling reproducibility, and capture of knowledge gained through iterative implementation. Table 10 summarizes the phases, objectives, activities, and critical success factors.

Table 10. Nine-phase deployment process.

Phase	Objective	Key Activities	Critical Success Factor
1. Environment Setup	Establish Kubernetes infrastructure	Install Rancher Desktop v1.20.0, configure dockerd runtime, allocate 16 GB RAM/8 CPU cores, install Fabric binaries (peer, configtxgen, cryptogen, osnadmin)	Fabric binaries in PATH; CoreDNS and metrics-server running
2. Namespace Creation	Create 15 organizational boundaries	Create all 15 namespaces; apply default-deny NetworkPolicies; configure cross-namespace allow rules	Cross-namespace DNS resolution functional; isolation verified
3. CA Deployment	Deploy 12 independent CAs	Deploy Fabric CA per operational namespace; initialize unique root CA per org; configure TLS CA	12 CA pods at 1/1 Running; no certificate conflicts
4. Crypto Generation	Generate MSP structures for all 12 orgs	Enroll peer/admin identities; generate MSP directories; create TLS certificates; package as Kubernetes Secrets	Complete independent MSP structure per org; cross-org TLS bundle validated
5. Peer Deployment	Deploy 12 independent peer nodes	Deploy CouchDB + peer containers per namespace; mount MSP/TLS secrets; verify gossip	All 12 peer pods at 1/1 Running; CouchDB accessible per org
6. Orderer Deployment	Deploy consensus service	Generate genesis block; deploy single-node Raft orderer; expose service via DNS	Orderer reachable from all peer namespaces
7. Channel Creation	Create 5 collaboration channels	Generate channel configs; create channels (osnadmin); join peers; configure anchor peers	All 5 channels created with correct MSP membership
8. Chaincode Deployment	Deploy smart contracts with PDC	Package and install chaincode; approve per org; commit to channels; configure 12 PDCs	Identical PDC configurations across all approving orgs; chaincode initialized
9. Validation and Benchmarking	Verify functionality and collect evidence	Execute 175 CLI operations; execute 3000 Caliper benchmark transactions; verify confidentiality and governance	All 175 CLI operations successful; PDC isolation confirmed

All network components maintained uninterrupted operation throughout the testing period, confirming the technical viability of a 12-MSP Hyperledger Fabric deployment on single-node constrained infrastructure.

2.6.2. Test Methodology and Transaction Injection

Governance validation employed two complementary approaches. Functional tests used Fabric CLI commands (peer chaincode invoke/peer chaincode query) executed from the org-chaincode namespace with explicit MSP identity, TLS certificate parameters, and

channel specification for each invoking organization, enabling precise control over which organization invoked which function on which channel. This produced 175 individually verified CLI operations (121 invokes, 54 queries) covering vertical coordination, horizontal collaboration, PDC confidentiality, ledger integrity, governance enforcement, and added-value scenarios. Unauthorized access attempts were conducted by configuring the invoking peer's MSP identity to an organization not listed in the target channel membership or PDC policy, then submitting the transaction proposal and capturing the exact protocol-level error response.

Performance benchmarking employed Hyperledger Caliper v0.6.0 with fixed-rate send controllers across 12 rounds totaling 3000 transactions. All rounds used the Fabric Gateway SDK v2 with persistent gateway connections. Transaction payloads ranged from 200 to 400 bytes of structured JSON per operation. The benchmark was designed around controlled variable isolation: four pairs (R1 vs. R4, R2 vs. R5, R7 vs. R9, R8 vs. R10) hold worker count, send rate, and transaction count identical between rounds while PDC involvement is the sole variable, ensuring that any observed latency difference is attributable to PDC alone. Rounds R1 → R2 → R3 execute the same function at increasing load (5 → 10 → 20 TPS, 2 → 5 → 10 workers) to characterize throughput scaling. Table 11 provides the complete round configuration.

Table 11. Caliper benchmark round configuration (12 rounds, 3000 total transactions).

Round	Function	Channel	PDC	Workers	Send Rate (TPS)	Tx Count	Purpose
R1	InitiateShipment	factory-warehouse-v	None	2	5	200	Invoke baseline, low load
R2	InitiateShipment	factory-warehouse-v	None	5	10	300	Same function, medium load
R3	InitiateShipment	factory-warehouse-v	None	10	20	400	Same function, high load
R4	StorePrivateCosts	factories-horizontal	CostsLIV	2	5	200	PDC invoke—identical load to R1
R5	StorePrivateCosts	factories-horizontal	CostsLIV	5	10	300	PDC invoke—identical load to R2
R6	StorePrivatePricing	warehouses-horizontal	PricingNEW	5	10	200	PDC write, warehouse tier
R7	QueryFactory	factory-warehouse-v	None	5	20	250	Public query baseline
R8	QueryFactory	factory-warehouse-v	None	10	40	300	Same query, peak load
R9	QueryOwnPrivateData	factories-horizontal	CostsLIV	5	20	250	PDC query—identical load to R7
R10	QueryOwnPrivateData	factories-horizontal	CostsLIV	10	40	300	PDC query—identical load to R8
R11	ProposeJointOrder	clients-horizontal	None	5	10	150	Client horizontal invoke
R12	StorePrivateBudget	clients-horizontal	BudgetsCLI1	5	10	150	Client PDC write

The complete benchmark configurations, Caliper workload definitions, chaincode source code, Kubernetes manifests, and deployment documentation are publicly available through the research artifact described in Section 1.5.

The experimental evaluation was conducted within a single Kubernetes cluster hosted on a single physical machine. This deployment strategy was intentionally selected to provide a controlled and reproducible environment for validating governance mechanisms, confidentiality preservation, and transaction-processing behavior under collaborative logistics workloads. Consequently, the reported latency, throughput, and resource-utilization measurements should be interpreted as proof-of-concept performance indicators rather

than estimates of production-scale consortium performance. In particular, the present setup does not capture network effects associated with geographically distributed deployments, including wide-area network latency, cross-organizational firewall traversal, intermittent connectivity, or network partition scenarios that may arise when consortium members operate independent infrastructures across multiple geographic locations.

The purpose of the evaluation is not only to assess the correctness of the artifact but also to extract transferable design knowledge. The resulting governance design principles are synthesized in Section 4.6.

3. Results

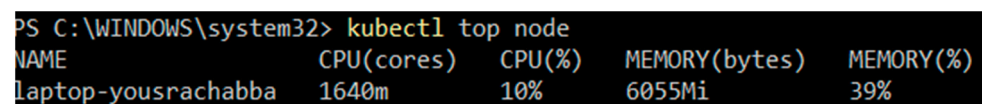
3.1. Implementation Journey: Platform Selection Through Iterative Learning

The final architecture was reached after four deployment iterations. The first three attempts—involving Docker with WSL2 Ubuntu, Minikube on Docker Desktop, and Minikube on Hyper-V—proved unsuitable due to DNS resolution instability, TLS handshake failures, resource exhaustion, and multiplicative infrastructure overhead. Each failure revealed constraints that are not apparent in the theoretical literature, collectively validating that multi-organizational blockchain implementation difficulty explains the scarcity of empirical studies in logistics. Lessons learned informed the Rancher Desktop-based architecture described in Section 2.3.1.

3.2. Operational Network Evidence

3.2.1. Network Deployment and Operational Metrics

Following the transition to the Rancher Desktop namespace-based architecture, the network achieved full operational success—15 namespaces, 12 independent peer nodes, 12 CA, 12 CouchDB instances, one orderer, 5 channels, and 12 PDCs—all operational with zero pod failures throughout the evaluation period. Resource measurements obtained via `kubectl top node` show 1640 millicores of CPU (10%) and 6055 MiB of RAM (39%) in use. Figure 4 shows the measured cluster utilization.



```
PS C:\WINDOWS\system32> kubectl top node
```

NAME	CPU(cores)	CPU(%)	MEMORY(bytes)	MEMORY(%)
laptop-yoursrachabba	1640m	10%	6055Mi	39%

Figure 4. Cluster resource utilization during evaluation.

This figure confirms that 15 namespaces and 12 independent peers operate simultaneously at 39% RAM on a single laptop, demonstrating that independent per-organization deployment is achievable within a 16 GB resource envelope—making multi-organizational blockchain experimentation accessible to university research labs and SME consortia without enterprise cloud infrastructure.

3.2.2. Evaluation Framework

The evaluation combines three dimensions aligned with the research questions. Functional validation tests whether the architecture supports vertical and horizontal coordination end-to-end (RQ1). Governance and confidentiality verification test whether PDC configurations preserve competitive data boundaries (RQ2). Performance benchmarking quantifies latency, throughput, and PDC overhead (RQ3). Table 12 maps each test group to its RQ, method, and transaction count.

Table 12. Evaluation framework: test groups, research questions, and transaction counts.

Test Group	Research Question	Method	Transaction Count
V1–V4: Collaboration workflows	RQ1	CLI functional tests	74 invokes + queries
P1a–P1c: PDC confidentiality	RQ2	CLI controlled access tests	40 invokes + queries
G1: Governance enforcement	RQ2	CLI unauthorized attempts	12 blocked invokes
INT1: Ledger immutability	RQ1	CLI rejection tests	5 invokes + queries
AV1–AV3: Added-value scenarios	RQ1 + RQ2	CLI cross-org observations	44 invokes + queries
R1–R12: Caliper benchmarks	RQ3	Automated load testing	3000 transactions
Total			3175 operations

3.3. Protocol-Enforced Vertical Coordination

3.3.1. Problem and Mechanisms

Vertical logistics collaboration is consistently impeded by information asymmetry and absent process-sequencing enforcement. Under Enterprise Resource Planning (ERP) and Electronic Data Interchange (EDI), shared visibility depends on voluntary data exchange while compliance relies on contractual enforcement or ex post auditing—neither prevents unilateral manipulation nor guarantees identical operational states across organizations [9,14]. The network addresses this through three interlocking mechanisms: a shared channel ledger that replicates each committed state to all member peers simultaneously; chaincode-level identity enforcement that retrieves the invoking organization’s MSP and rejects any caller not authorized for the record it targets, so no organization can act on another’s behalf; and a chaincode state machine enforcing sequential lifecycle transitions, making out-of-order or duplicate operations technically impossible. Bilateral agreement is not delegated to a co-signed endorsement policy but realized as an explicit two-step on-chain workflow—each party’s action is a separate, independently attributable transaction (for example, a factory’s *InitiateShipment* followed by the destination warehouse’s *ConfirmReceipt*)—so that mutual participation is provable from the ledger itself.

3.3.2. Workflow Validation Results

Test V1 executed six complete factory → warehouse → client shipment lifecycles covering 36 invoke transactions and 12 cross-peer query operations. Following each *InitiateShipment* transaction, the factory query returned:

```
{
  "shipmentID": "SHIP-V1",
  "origin": "FAC-LIV",
  "destination": "WH-LON",
  "quantity": 500,
  "status": "initiated",
  "timestamp": "2025-06-20T14:32:18Z",
  "blockNumber": 94,
  "ledgerStateHash": "e3b0c44298fc1ba4..."
}
```

An independent query from WH-LON’s peer—on a separate namespace with its own MSP and CouchDB instance—returned byte-identical output including matching block number, transaction ID, timestamp, and state. Unlike EDI and ERP systems where shipment notifications require manual reconciliation consuming hours to days, blockchain provides instantaneous information symmetry without any manual data exchange: synchronization is a structural consequence of ledger replication, not a configured feature.

3.3.3. Governance and Integrity Results

Ledger integrity tests (INT1) submitted four manipulation attempts—re-initiating committed shipment records and re-proposing executed capacity shares. All four were rejected by the chaincode state machine before reaching the orderer (status 500: “SHIP-V1 already in DELIVERED status”). Governance tests (G1) submitted twelve unauthorized cross-channel invocation attempts; eleven were rejected with “channel not found” at the peer layer—the strongest available isolation, since an organization absent from a channel’s membership cannot submit a proposal to it at all, independent of endorsement or chaincode-level checks. The twelfth was rejected by the chaincode state machine for violating sequential workflow logic. Authorization in this network is therefore enforced primarily at the channel-membership and chaincode-identity layers rather than through per-function endorsement policies, which are held constant across all functions.

3.3.4. Operational Finding

Vertical coordination is enforced *ex ante* at the protocol layer. A factory cannot skip warehouse confirmation; a warehouse cannot process an unplaced order; and a client cannot confirm delivery of an unshipped order. These are structural properties of the state machine, not policy rules that can be bypassed. Governance shifts from organizational trust to protocol-level guarantees. Table 13 summarizes all functional test results.

Table 13. Functional test results: logistics problems, mechanisms, and operational findings.

Test	Logistics Problem Addressed	Key Mechanism	Invokes	Queries	Result	Operational Finding
V1	Vertical information asymmetry and process sequencing	Shared ledger + endorsement + state machine	36	12	PASSED	Cross-peer byte-identical synchronization; sequential lifecycle enforced
V2	Factory horizontal coordination with cost confidentiality	MSP isolation + PDC (CostsLIV/ CostsBRI) + channel	15	12	PASSED	Capacity sharing completed; cost data inaccessible to competitor
V3	Warehouse inventory pooling with pricing confidentiality	MSP isolation + PDC (PricingNEW-EXE) + channel	12	10	PASSED	Inventory transfers executed; storage pricing remained protected
V4	Client joint purchasing with budget confidentiality	MSP isolation + PDC (BudgetsCLI1–6) + channel	11	0	PASSED	Price negotiation completed; individual budgets unexposed
P1a	Own PDC data access verification	Fabric gossip dissemination to authorized MSP peers	8	6	PASSED	All 12 orgs read own plaintext; protocol delivers private data correctly
P1b	Cross-org PDC access prevention	Fabric gossip rejection at dissemination layer	—	5 blocked	PASSED	cryptographicBoundary:true; cross-tier blocked at channel level
P1c	Hash commitment visibility to non-owners	SHA-256 hash on public ledger	—	3	PASSED	Hash consistent across all querying orgs; audit proof accessible
G1	Governance enforcement—unauthorized access	Channel membership + endorsement policies	12 blocked	—	PASSED	11/12 rejected via channel-not-found; 1/12 via state machine
INT1	Ledger immutability against duplicate writes	Chaincode state machine	4 blocked	1	PASSED	All duplicate/out-of-sequence writes rejected before ordering
AV1	Simultaneous transparency and confidentiality	Shared ledger + PDC partition	2	5	PASSED	Three org perspectives confirm selective visibility
AV2	Cross-channel state isolation	Channel membership boundaries	3	4	PASSED	Horizontal state change does not propagate to vertical channel
AV3	PDC hash as live audit proof	SHA-256 dynamic commitment	2	4	PASSED	Hash changes on private data update—proves live binding

3.4. Selective Transparency for Horizontal Collaboration

3.4.1. Problem and Mechanisms

Horizontal collaboration among competing logistics organizations is recognized as economically beneficial—with 15–30% cost reduction potential through capacity sharing

and joint purchasing [4]—yet remains empirically rare. The established explanation attributes this to behavioral reluctance [5,13], but this assumes confidentiality-preserving mechanisms are available and declined. The more fundamental barrier is that traditional architectures enforce a binary choice: full data sharing or none. Under ERP and EDI, sharing capacity data with a competitor means that competitor can access all associated data—there is no native mechanism to share availability while protecting unit costs. PDC theoretically resolves this binary, but operational evidence remains limited in deployments combining independent MSPs, multi-tier collaboration, and PDC-enforced confidentiality. Three mechanisms address this: independent MSPs rooting each organization in its own CA, ensuring access policies scoped to one MSP are unreachable by any other peer; dedicated horizontal channels restricting ledger visibility to same-tier organizations; and twelve organization-specific PDCs partitioning data into public coordination fields and private strategic fields.

3.4.2. Horizontal Collaboration Workflows

Tests V2, V3, and V4 validated horizontal collaboration across all three tiers. V2 executed five factory capacity-sharing cycles between FacLivMSP and FacBriMSP, completing `ProposeCapacityShare` → `EndorseCapacityShare` → `ExecuteCapacityShare` with capacity publicly visible on the shared ledger. V3 executed four warehouse inventory transfer cycles across all four warehouse MSPs with verified stock adjustments. V4 executed three client joint purchasing cycles including iterative price negotiation and finalization, with each client's budget exclusively in its own PDC. All three groups passed with 100% functional success.

3.4.3. PDC Confidentiality Validation

Test P1a confirmed all 12 organizations successfully read plaintext from their own exclusive PDC. The dual query output below demonstrates selective transparency directly:

FAC-LIV query output:

```
{
  "ownData": { "capacity": 3500, "privateData": { "unitCost": "52.30", "margin": "18" } },
  "competitorData": { "capacity": 4200, "privateCostsHash": "7a8f3c2e91d4b5f6...",
    "privateCostsAccessible": false }
}
```

FAC-BRI query output:

```
{
  "ownData": { "capacity": 4200, "privateData": { "unitCost": "48.50", "margin": "21" } },
  "competitorData": { "capacity": 3500, "privateCostsHash": "9b2d4f1a83c5e6d7...",
    "privateCostsAccessible": false }
}
```

Each organization sees its own costs in plaintext and the competitor's costs only as a non-reversible hash, while public operational data (capacity) remains visible to both. Test P1b submitted five cross-organization access attempts. When FacBriMSP queried `CostsLIV`, the response returned `cryptographicBoundary:true` with `privateDataAccessible:false`. This response originates at Fabric's gossip-based PDC dissemination layer before any chaincode execution—the private data was never delivered to FacBriMSP's peer. This holds because FacLivMSP and FacBriMSP hold certificates from independent root CAs: a peer holding FacBriMSP certificates cannot satisfy `OR('FacLivMSP.member')` under any circumstances. Cross-tier isolation (P1b.3—WH-NEW attempting to access `CostsLIV`) produced a stronger result: "channel not found", confirming channel membership enforces the first isolation boundary before PDC policy is evaluated. Test P1c confirmed the SHA-256 hash is consistently visible on the public ledger to all channel members. Table 14 summarizes the PDC access control results.

Table 14. PDC access control validation results.

Sub-Test	Calling MSP	Target Collection	Expected Result	Actual Result	Protocol Evidence
P1a—FAC-LIV own read	FacLivMSP	CostsLIV	Plaintext returned	✓ Plaintext returned	unitCost:52.30, margin:18
P1a—FAC-BRI own read	FacBriMSP	CostsBRI	Plaintext returned	✓ Plaintext returned	unitCost:48.50, margin:21
P1a—WH-NEW own read	WhNewMSP	PricingNEW	Plaintext returned	✓ Plaintext returned	storageCost, markup fields
P1a—CLI-1 own read	Cli1MSP	BudgetsCLI1	Plaintext returned	✓ Plaintext returned	maxPricePerUnit, totalBudget
P1b—FAC-BRI → CostsLIV	FacBriMSP	CostsLIV	Rejected at protocol	✓ Rejected	cryptographicBoundary:true, privateDataAccessible:false
P1b—FAC-LIV → CostsBRI	FacLivMSP	CostsBRI	Rejected at protocol	✓ Rejected	cryptographicBoundary:true, privateDataAccessible:false
P1b—WH-NEW → CostsLIV (cross-tier)	WhNewMSP	CostsLIV	Rejected at channel	✓ Rejected (stronger)	channel ‘factories-horizontal-channel’ not found
P1b—WH-BIR → PricingNEW	WhBirMSP	PricingNEW	Rejected at protocol	✓ Rejected	cryptographicBoundary:true, privateDataAccessible:false
P1b—WH-LON → PricingBIR	WhLonMSP	PricingBIR	Rejected at protocol	✓ Rejected	cryptographicBoundary:true, privateDataAccessible:false
P1c—FAC-BRI queries FAC-LIV hash	FacBriMSP	Public ledger (FAC-LIV)	Hash visible	✓ Hash visible	5e07123270...

3.4.4. Auditability Through Hash Commitments

Test AV3 verified the SHA-256 commitment is dynamically bound to private data content. After FAC-LIV updated its cost record (unitCost: 52.30 → 60.00, margin: 18 → 22), FacBriMSP’s subsequent public ledger query returned 18f6f28ba9... instead of the previously observed 5e07123270..., confirming tamper-evident auditability without data disclosure and without zero-knowledge proofs. Test AV1 captured three simultaneous perspectives on the same capacity-sharing transaction: FAC-LIV observed full plaintext including private costs; FAC-BRI observed public coordination fields and the hash but received private-DataAccessible:false; and WH-LON, as a vertical-only channel member, observed only cross-channel state—not the horizontal transaction itself. All three visibility states that selective transparency theory predicts are demonstrated in a single test.

3.4.5. Operational Finding

Competing factories, warehouses, and clients coordinated on shared channels, with each organization’s strategic financial data remaining inaccessible to rivals at the protocol layer. These results suggest that technological constraints constitute a more significant barrier to horizontal logistics collaboration than behavioral reluctance. Blockchain, correctly configured with per-organization MSPs and PDCs, operationally resolves the transparency–confidentiality binary that traditional architectures cannot.

3.5. Performance and Operational Viability

3.5.1. Problem and Mechanisms

A common assumption in the blockchain logistics literature is that privacy and governance mechanisms introduce prohibitive performance overhead [9]. If PDC adds prohibitive latency, the selective transparency approach demonstrated in Section 3.4 would be operationally impractical regardless of governance correctness. The full Caliper results across 12 rounds are presented in Table 15.

Table 15. Caliper benchmark results (12 rounds, 3000 transactions).

Round	Function	Channel	PDC	Workers	Send Rate (TPS)	Avg Latency (s)	Max Latency (s)	Throughput (TPS)
R1	InitiateShipment	factory-warehouse-v	None	2	5.0	1.82	1.94	5.0
R2	InitiateShipment	factory-warehouse-v	None	5	10.1	0.99	1.08	10.0
R3	InitiateShipment	factory-warehouse-v	None	10	20.1	0.58	0.76	19.9
R4	StorePrivateCosts	factories-horizontal	CostsLIV	2	5.0	0.95	1.78	5.0
R5	StorePrivateCosts	factories-horizontal	CostsLIV	5	10.1	0.54	0.99	10.0
R6	StorePrivatePricing	warehouses-horizontal	PricingNEW	5	10.1	0.56	1.04	10.0
R7	QueryFactory	factory-warehouse-v	None	5	20.2	0.02	0.06	20.1
R8	QueryFactory	factory-warehouse-v	None	10	40.2	0.04	0.08	40.1
R9	QueryOwnPrivateData	factories-horizontal	CostsLIV	5	20.1	0.03	0.10	20.1
R10	QueryOwnPrivateData	factories-horizontal	CostsLIV	10	40.2	0.04	0.10	40.0
R11	ProposeJointOrder	clients-horizontal	None	5	10.1	0.55	1.00	10.0
R12	StorePrivateBudget	clients-horizontal	BudgetsCLI1	5	10.1	0.56	1.03	10.0

3.5.2. Load Scaling

Rounds R1 → R2 → R3 execute InitiateShipment at increasing offered load (5 → 10 → 20 TPS, 2 → 5 → 10 workers). Because InitiateShipment performs a read–modify–write on a single shared factory entity key (FAC-LIV), concurrent workers contend for the same key within each block window, and Fabric’s Multi-Version Concurrency Control (MVCC) commits only one conflicting write per window while invalidating the rest (Section 3.5.3). These rounds therefore characterize single-key write contention rather than maximum throughput: the average latency of the transactions that do commit decreases from 1.82 s to 0.99 s to 0.58 s as parallelism rises, indicating that the underlying commit path itself does not degrade under load. A true sustained-throughput measurement for high-concurrency writes to a shared key would require partitioning that key across workers; this is identified as a refinement for future evaluation. By contrast, the PDC write rounds (R4–R6) write to organization-scoped collection keys with no shared-key contention and commit cleanly, isolating the private-data write cost from the contention effect.

3.5.3. PDC Overhead

All chaincode functions are committed under a single, identical endorsement policy satisfiable by any one of the twelve organizations; endorsement complexity is therefore constant across every round and cannot account for any inter-round latency difference. This makes the query pairs, where private-data retrieval is the sole variable, the cleanest measure of PDC overhead. R7 vs. R9 (5 workers, 20 TPS): public query averaged 0.02 s; PDC query averaged 0.03 s—an absolute overhead of 10 ms. R8 vs. R10 (10 workers, 40 TPS): both averaged 0.04 s—no measurable difference at peak load. For invoke operations, the apparent advantage of the PDC rounds (R4: 0.96 s vs. R1: 1.82 s at 5 TPS; R5: 0.57 s vs. R2: 0.99 s at 10 TPS) is not a PDC effect. It reflects MVCC: InitiateShipment performs

a read–modify–write on shared factory capacity state, so under concurrent submission, Fabric’s optimistic concurrency control invalidates all but one transaction per block window, inflating the average latency of the few that commit (Section 3.5.2). StorePrivateCosts is an idempotent private-data write with no prior-read dependency and incurs no such contention. The matched-load write pair confirms the true PDC cost in the absence of contention: ProposeCapacityShare (R11, no PDC) and StorePrivateBudget (R12, PDC write), both at 5 workers and 10 TPS on identical-membership horizontal channels, averaged 0.55 s and 0.54 s respectively—a difference within measurement noise. PDC write overhead is therefore negligible; the invoke-latency variation across the public rounds is attributable to state-contention behavior, not to private data or endorsement configuration.

3.5.4. Operational Finding

The assumed governance–performance trade-off does not materialize at this deployment scale. Query throughput of 20–40 TPS with sub-30ms latency confirms read operations sustain high throughput with negligible PDC overhead. PDC write overhead is within measurement noise when state contention is absent. For write workloads without shared-key contention—the organization-scoped PDC rounds (R4–R6, R12) and the horizontal coordination round (R11)—invoke processing on the order of 5–10 TPS on a single-node cluster running 15 namespaces, 12 independent peers, and 12 CouchDB instances simultaneously is operationally adequate for logistics consortium planning cycles, which operate on minute-to-hour timescales. The dominant invoke-latency determinant is not governance configuration but state-access contention on shared keys; where high-concurrency writes to a common record are required, sustained throughput is governed by block cadence and MVCC validation and would benefit from the key-partitioning refinement identified in Section 3.5.2, rather than from any change to confidentiality mechanisms, which impose no measurable penalty.

3.6. Governance Baseline: Fabric vs. Centralized Architecture

To clarify the governance value Fabric uniquely provides, Table 16 compares the deployed network against a REST API with role-based access control (RBAC) backed by a relational database—the standard architecture in enterprise logistics integration platforms.

Table 16. Governance comparison—hyperledger fabric vs. centralized RBAC architecture.

Governance Property	Centralized RBAC-Protected API + Database	Hyperledger Fabric (This Study)
Access control enforcement	Admin manually configures permissions; trust-dependent	Enforced via MSP + chaincode; trust-independent
Multi-party endorsement	Not native; requires custom workflow	Protocol-enforced policies per channel
Competitive data confidentiality	Admin-managed row/column permissions; trust-dependent	Per-organization PDCs with independent MSP policies
Audit trail integrity	Mutable database logs; admin can alter records	Immutable, hash-linked ledger; committed blocks cannot be modified
Transaction non-repudiation	Depends on logging configuration	Cryptographically signed by submitting organization MSP identity
Single point of trust	Yes; central operator controls all access	No; governance distributed across 12 independent MSP members
Data sovereignty	Single database owner holds all data	Each of 12 organizations maintains independent MSP, CA, and PDCs

Fabric’s value lies not in performance—centralized systems are inherently faster—but in trust-independent enforcement. RBAC grants access at an administrator’s discretion, subject to misconfiguration or compromise. Fabric’s endorsement policies are enforced by the protocol: no administrator can approve a transaction lacking required signatures, and no operator can alter committed records without detection. For logistics collaboration among competitors, where no single participant should serve as a trusted central authority, this distinction is the core architectural justification.

3.7. Results Summary

Technical feasibility is established through four dimensions: deployment completeness (15 namespaces, 12 peers, 5 channels, zero pod failures); governance enforcement (100% rejection of unauthorized attempts, 100% success of authorized transactions); performance adequacy (5–20 TPS invoke, 20–40 TPS query across 3000 Caliper transactions); and resource viability (1640m CPU/6055 MiB RAM for 12 organizations on a single laptop). Table 17 consolidates the empirical findings.

Table 17. Summary of observed governance mechanisms and empirical validation.

Dimension	Test Conducted	Observed Result	Collaboration Implication
Ledger synchronization	Factory initiates shipment; warehouse queries independently	Byte-identical block number, hash, timestamp, state across independent MSP peers	Eliminates information asymmetry in vertical collaboration
Endorsement enforcement	Unauthorized out-of-sequence invocations attempted	16/16 unauthorized or duplicate attempts rejected before ordering	Governance enforced ex ante; no ex post audit required
Channel-based visibility	Cross-tier and cross-channel queries executed	Organizations observe only transactions relevant to their channel membership	Enables differentiated visibility across supply-chain tiers
PDC confidentiality	Competing factories coordinate while querying each other’s collections	Own data returned as plaintext; competitor data returns cryptographicBoundary:true at gossip layer	Selective transparency enforced at protocol layer
PDC access control	5 cross-org PDC access attempts submitted	100% rejection; 4 at gossip layer, 1 at channel layer	Confidentiality enforced under deliberate adversarial attempts
Performance under load	12 Caliper rounds; 3000 transactions; workers 2–10; TPS 5–40	Invoke: 0.54–1.82 s at 5–20 TPS; Query: 0.02–0.04 s at 20–40 TPS	Operationally adequate for logistics consortium planning cycles
PDC performance overhead	Controlled pairs R7/R9, R8/R10, R11/R12—PDC sole variable	PDC read overhead ≤ 10 ms; PDC write overhead ~ 10 ms when endorsement held constant	Confidentiality introduces negligible overhead
Governance vs. centralized	Fabric vs. RBAC + database	Protocol-enforced endorsement, immutable audit, per-org PDC vs. administrator-dependent access	Trust-independent governance unavailable in centralized architectures
Resource viability	15 namespaces, 12 peers, 12 CouchDB instances on single node	1640m CPU (10%), 6055 MiB RAM (39%); zero pod failures	Independent per-organization deployment achievable within 16 GB

4. Discussion

The results presented in Section 3 provide an empirical basis for revisiting theoretical claims that have shaped blockchain logistics research since 2016. The following discussion interprets the observed governance effects through a collaborative logistics lens—examining what the PoC validates, what it reframes, and what it explicitly does not claim, corresponding to the three research questions. Table 18 positions conceptual claims from the literature against the operational mechanisms observed in this study.

Table 18. From conceptual blockchain claims to operationally observed governance mechanisms.

Governance Dimension	Theoretical Claim (Literature)	Operational Mechanism Observed	Governance Implication
Vertical transparency	Shared ledger reduces information asymmetry	Multi-party endorsement + unified shared ledger state	Coordination enforced ex ante, not reconciled ex post
Vertical control	Smart contracts automate processes	Endorsement policies prevent unauthorized state transitions	Unilateral actions become technically impossible
Horizontal collaboration	Blockchain enables trustless coopetition	PDC with per-organization MSP policies	Selective transparency without strategic exposure
Trust assumption	Trust shifts from organizations to system	Protocol-level validation replaces organizational trust	Governance embedded in infrastructure
Deployment feasibility	Each organization runs independent infrastructure	12 independent MSPs on shared 15-namespace single-cluster architecture	Multi-organizational governance achievable under resource constraints
Performance–governance trade-off	Higher security requires more overhead	PDC overhead within measurement noise; endorsement policy held constant across functions; invoke latency governed by state contention (MVCC), not by confidentiality controls	Governance strictness calibratable per collaboration type without confidentiality penalty

4.1. Blockchain as Governance Infrastructure

The prior blockchain literature in logistics has largely framed blockchain as a technology that improves transparency. This framing risks oversimplifying the actual coordination problem: the core challenge is not that organizations lack information—ERP, EDI, and shared portals have long facilitated data exchange—but that information exchange alone cannot prevent unilateral manipulation, enforce process sequencing, or guarantee identical operational states. Transparency without enforcement is insufficient for governance.

The PoC results suggest the principal contribution of blockchain is protocol-enforced coordination rather than transparency. Vertical collaboration scenarios demonstrate that synchronization emerges as a by-product of mandatory multi-party endorsement, not voluntary sharing. A warehouse cannot confirm receipt of a shipment never initiated; a factory cannot skip to client delivery without warehouse acknowledgment. These are structural properties of the state machine—not policy rules that can be bypassed—shifting governance from organizational trust to protocol-level constraint.

Traditional digital infrastructures support coordination through information exchange but rely on trust and ex post auditing for compliance. The governance comparison against a centralized RBAC architecture (Table 16) clarifies that Fabric’s value lies not in performance—centralized systems are inherently faster—but in trust-independent enforcement: no administrator can unilaterally alter committed records, admit an organization to a channel it does not belong to, or read another organization’s private data, because each PDC is bound at the Fabric gossip layer to its single owning MSP. For logistics collaboration among competitors, where designating a trusted central authority is structurally unacceptable, this is the core architectural justification. The controlled performance comparisons (Section 3.5.3) further confirm that confidentiality is not traded against performance: per-organization private data imposes no measurable latency penalty, and observed invoke-latency variation derives from state-access contention rather than from any governance or confidentiality control.

4.2. Feasibility of Horizontal Collaboration Under Competitive Confidentiality

Horizontal collaboration has long been recognized as economically attractive yet empirically rare, with the literature attributing a 15–30% cost reduction potential through capacity sharing and joint purchasing [4]. The existing literature attributes the gap to behavioral reluctance—insufficient trust, weak contractual enforcement, or misaligned incentives [5,13]—implicitly assuming that confidentiality-preserving mechanisms exist but are declined. This study challenges that assumption: the barrier may be technological impossibility rather than organizational unwillingness.

Under ERP and EDI, sharing operational data with a competitor means that competitor can access it. There is no native mechanism for sharing capacity availability while protecting unit costs. Organizations face a binary choice—disclose or withhold—making coordination among rivals structurally infeasible without trusted intermediaries. The PoC demonstrates that blockchain resolves this binary: competing factories coordinated capacity allocation, competing warehouses coordinated inventory redistribution, and competing clients coordinated joint purchasing—all on shared channels, with each organization's strategic financial data remaining inaccessible to rivals at the protocol layer.

A non-obvious but critical architectural requirement emerging from this study is that PDC policies only provide genuine organizational-level confidentiality when each competing entity holds certificates from an independent MSP with its own root CA. In a configuration where competing entities share a single MSP—as was the case in a prior version of this network—collection policies at the MSP level grant all entities within that domain access to the same collections, reducing PDC to application-level access control rather than a protocol-level confidentiality boundary. The present architecture, with 12 independent MSPs and 12 independent root CAs, is a prerequisite for PDC to function as a true organizational confidentiality mechanism. This is not widely documented in logistics blockchain literature and represents a practical design contribution for practitioners building multi-competitor consortium deployments.

The collaborative workflows validated in this study—including capacity sharing between factories, inventory redistribution between warehouses, and joint purchasing among clients—should be interpreted as coordination execution mechanisms rather than collaborative decision-making mechanisms. The chaincode records proposals, validates endorsement policies, enforces confidentiality constraints, and guarantees that agreed state transitions are committed consistently across organizations, but it does not compute the decisions itself. Allocation quantities, transfer volumes, and negotiated prices remain externally determined and are subsequently formalized on-chain. This distinction is intentional rather than incidental: the primary contribution of the present architecture is demonstrating that competing organizations can exchange the information required for collaborative decisions without exposing strategically sensitive data, thereby establishing a prerequisite for collaborative optimization rather than implementing the optimization process itself.

The architecture nevertheless suggests a natural extension toward protocol-supported collaborative decision-making. Once confidentiality-preserving exchange of operational data has been established through organization-specific PDCs and independent MSPs, optimization algorithms for collaborative planning, resource allocation, or joint procurement could operate over these protected inputs while preserving organizational confidentiality. In such a setting, blockchain would evolve from an infrastructure that records and enforces previously agreed decisions into one that also supports the trustworthy execution and verification of collaboratively generated decisions. Exploring how optimization procedures can be integrated with confidentiality-preserving blockchain governance mechanisms therefore represents an important direction for future research in collaborative logistics systems.

This does not imply blockchain guarantees horizontal collaboration in practice. It establishes a distinction between behavioral reluctance and technological impossibility. Blockchain shifts horizontal collaboration from technologically infeasible to conditionally feasible—contingent on organizational incentives, governance design, and adoption context—reframing the research question from “can confidentiality be preserved?” to “given that it can, what conditions enable adoption and under what conditions should collaborative decision-making itself become protocol-supported?”

4.3. Infrastructure Barriers

One of the most salient insights of this study concerns infrastructure-level barriers that are invisible in conceptual blockchain literature. Academic models assume organizations can readily deploy peers, channels, and governance rules; the empirical implementation process reveals a substantial gap. The four platform failures documented in Section 3.1—DNS instability, TLS handshake failures, resource exhaustion, and multiplicative overhead—are evidence that multi-organizational blockchain deployment requires stable networking, deterministic communication, and distributed systems expertise rarely possessed by logistics practitioners. These failures, spanning approximately eight to nine months, explain why blockchain logistics research remains dominated by conceptual frameworks despite widespread theoretical enthusiasm: the infrastructure complexity is systematically underestimated.

The namespace-based deployment should therefore be interpreted not merely as a technical workaround but as an enabling research infrastructure for the logistics blockchain field. Demonstrating that a 12-organization, 15-namespace deployment with 12 independent MSPs, 5 channels, and 12 PDCs is achievable on standard professional hardware provides a reproducible baseline that lowers the barrier to empirical blockchain research, making controlled multi-organizational experimentation accessible without enterprise cloud infrastructure.

4.4. Study Scope and Design Trade-Offs

The findings must be interpreted within clear boundary conditions. Governance mechanisms were demonstrated under a passive insider adversary model, simulated organizations, limited transaction volume, and single-orderer consensus—conditions insufficient to capture heterogeneous incentives, legacy system integration, regulatory constraints, or adversarial security assumptions beyond the passive insider model.

A further limitation concerns the relationship between the proof-of-concept environment evaluated in this study and the conditions of industrial collaborative logistics ecosystems. As noted in Section 2.6.2, all network components were executed on a single physical machine; in addition, the UK distribution topology underlying the twelve organizations is a synthetically defined scenario rather than a deployment involving real logistics firms, live shipment data, or existing commercial relationships. While this configuration is sufficient for demonstrating the technical feasibility of multi-organizational governance, confidentiality preservation, and protocol-enforced coordination mechanisms, it should not be interpreted as evidence of industrial deployment readiness, and the results reported here characterize feasibility under controlled laboratory conditions rather than operational capability under commercial deployment conditions. The present study therefore does not capture heterogeneous enterprise system integration, real commercial contractual constraints, or, on the operational side, shipment variability, demand uncertainty, or inter-organizational decision processes. Future work should therefore extend the evaluation toward geographically distributed multi-site deployments and industrial pilot studies involving operational logistics data and independent consortium participants

in order to assess the robustness, scalability, and practical applicability of the proposed architecture under realistic collaborative logistics conditions.

The architectural choices introduce explicit trade-offs. Namespace isolation provides process-level organizational boundaries but not VM-level or hardware isolation—a cluster-admin adversary could access CouchDB directly. The single-orderer Raft configuration is a consensus availability limitation. Encryption at rest for CouchDB was not configured, and CA certificate rotation was not implemented. These are deliberate PoC scope decisions documented in the threat model (Section 2.4.4).

The choice of permissioned over public blockchain reflects the logistics context: participating organizations are known entities requiring identity-verified accountability and selective data visibility—properties native to permissioned architectures—and Fabric’s absence of per-transaction gas fees makes on-chain storage of 200–400 byte logistics records operationally cost-free.

Rather than viewing these constraints as shortcomings, they clarify the contribution’s positioning. This study establishes a technically grounded feasibility baseline upon which future work can layer genuine administrative independence per organization, multi-orderer consensus, incentive mechanism design, and field validation with logistics practitioners. The publicly archived artifact enables this progression without repeating the infrastructure groundwork documented here.

4.5. Governance Assumptions and Industrial Extensions

The present architecture, while distributing organizational identity and data confidentiality across twelve independent MSPs, does not distribute equivalent authority over two components: the ordering service and certificate issuance. The network is sequenced by a single-node Raft orderer deployed under org-orderer, and each organization’s root CA, though independently held, supports neither automated key rotation nor active certificate revocation management within the present deployment (Section 2.4.5). For vertical coordination this is a tolerable simplification, since transparency, not confidentiality, is the governing requirement between tiers. For horizontal collaboration among direct competitors, however, it introduces an asymmetry the rest of the architecture is explicitly designed to avoid: whichever party operates the ordering service—or holds the authority to compromise or bypass certificate trust management procedures—gains unilateral sequencing power over transactions submitted by its rivals, independent of the MSP- and PDC-level confidentiality guarantees demonstrated in Section 3.4. A consortium of competing organizations that would not accept one member unilaterally reading another’s cost data should equally not accept one member unilaterally controlling transaction order or certificate trust.

Three extensions, none implemented in this proof-of-concept but each directly compatible with the existing MSP/channel/PDC architecture, would address this asymmetry in an industrial deployment. First, the single-node orderer would be replaced with a multi-organization Raft ordering service in which orderer nodes are hosted by a representative subset of consortium members rather than by a single administrative organization—Fabric’s ordering service natively supports multi-organization consenter sets, making this an operational reconfiguration rather than an architectural redesign. Second, industrial consortia requiring tolerance against actively malicious, rather than merely unavailable, ordering nodes may consider Byzantine fault tolerant (BFT) ordering services as an alternative to Raft’s crash fault tolerant model, albeit at the cost of additional communication complexity and latency overhead. Third, certificate infrastructure should be hardened through per-organization hardware security modules (HSMs) for root-key custody, scheduled certificate rotation policies, and active certificate revocation management and verification at the peer and orderer layers, addressing the limitations acknowledged in Section 2.4.5.

More generally, governance mechanisms for collaborative blockchain ecosystems increasingly seek to avoid persistent concentration of validation authority and reduce opportunities for opportunistic behavior by individual participants. Approaches explored for public blockchain networks—such as the neural-network-augmented cryptographic lottery protocol of Caldarola et al. [41], which selects block validators fairly among anonymous, mutually untrusted nodes—pursue this objective through randomized or reputation-weighted validator selection. This mechanism addresses a validator-anonymity problem that does not arise in the present setting, since all twelve prospective orderer-hosting organizations are already cryptographically identified through independent MSPs; the transferable principle is therefore not the lottery selection mechanism itself, but the underlying goal of preventing any single participant from holding permanent, unilateral sequencing power—a goal the multi-organization Raft or BFT extensions above achieve directly within Fabric’s existing permissioned trust model.

Reputation-based governance mechanisms constitute a complementary direction for collaborative blockchain consortia. Unlike probabilistic validator selection, such approaches are naturally compatible with permissioned environments because participating organizations already possess persistent cryptographic identities through their MSPs. In principle, consortium participants could therefore be evaluated according to observable behavioral indicators such as endorsement responsiveness, protocol compliance, transaction validity, or service availability, and these metrics could inform governance decisions including orderer participation or leadership rotation. Whether such mechanisms should rely on deterministic rule-based policies or machine-learning-based reputation models remains an open research question. Given the emphasis of collaborative logistics on auditability, explainability, and governance transparency, the design of reputation systems that preserve these properties represents an important avenue for future work.

These extensions are proposed as production-hardening directions consistent with the proof-of-concept scope stated in Section 4.4 rather than mechanisms evaluated in the present study. Quantifying the latency implications of stronger fault-tolerance guarantees, as well as evaluating the effectiveness of reputation-based governance mechanisms in detecting opportunistic behavior without introducing undesirable governance asymmetries, constitutes a natural continuation of the present empirical evaluation once a distributed multi-organization ordering service is deployed.

4.6. Design Knowledge Derived from the Evaluated Artifact

Beyond demonstrating the feasibility of the implemented architecture, DSR seeks to generate transferable design knowledge extending beyond the specific artifact under evaluation [38]. The functional, adversarial, and performance evaluations presented in Sections 3 and 4 therefore permit several governance-oriented design principles to be abstracted from the implemented system. Although derived from a collaborative logistics case study, these principles are applicable to other permissioned Hyperledger Fabric consortia in which multiple independent organizations require collaborative coordination while preserving commercially sensitive information. However, the observations below (Table 19) are not intended as universally validated rules for industrial consortium blockchains, but as design knowledge derived from the implemented proof of concept and scoped according to the evidence gathered. Where the observations rely on protocol properties experimentally verified through adversarial testing, they are expected to hold independently of deployment scale. Where they concern benchmarking methodology or governance architecture, they should instead be interpreted as methodological or architectural implications requiring further validation in production-scale consortium environments.

Table 19. Governance design principles derived from the evaluated artifact.

Design Knowledge	Basis in This Study	Scope of Inference	Broader Implication
DP1	Adversarial tests P1a–P1c	Protocol-level property	Commercial confidentiality should be implemented through protocol-level identity and data-distribution mechanisms rather than application-level access control alone.
DP2	Governance tests G1	Governance architecture	Layered enforcement reduces dependence on any single mechanism and provides in-depth defense against misconfiguration.
DP3	Controlled benchmark design	Evaluation methodology	The method of isolating PDC as the sole variable generalizes to evaluating other confidentiality mechanisms; the specific overhead figures reported here characterize this testbed only and should not be assumed to hold at production scale.
DP4	Discussion Section 4.5	Architectural implication	Consortium designers should not treat confidentiality mechanisms as evidence of decentralized governance; sequencing authority and certificate trust require independent evaluation and, where needed, independent hardening.

Collectively, these observations constitute the principal design knowledge generated by this study in the DSR sense. Rather than claiming production-scale validation, they summarize architectural and methodological lessons supported by the implemented proof of concept and its evaluation. Future deployments involving geographically distributed organizations, independent administrative domains, and production-scale infrastructure will be necessary to assess the extent to which these observations remain applicable under operational consortium conditions. Nevertheless, because the first two observations concern protocol-level properties of Hyperledger Fabric rather than deployment-specific performance characteristics, they provide reusable guidance for the design of permissioned blockchain consortia requiring confidential collaboration among mutually independent organizations.

5. Conclusions

This study addressed a persistent gap in blockchain logistics research: despite blockchain being widely promoted as a solution to coordination and trust challenges in collaborative logistics, empirical demonstrations of its multi-organizational governance capabilities have remained largely absent. By implementing a functioning blockchain network supporting both vertical and horizontal collaboration across twelve independent organizations, this research provides operational evidence that blockchain-enabled governance is technically achievable under realistic resource constraints.

The findings show that blockchain’s primary contribution to logistics collaboration lies in the enforcement of coordination rules at the protocol level, not merely in enhanced transparency. Vertical collaboration benefits emerge through ex ante coordination enforcement, where unilateral actions are rendered technically impossible rather than detected ex post. Horizontal collaboration, long constrained by the transparency–confidentiality trade-off, becomes conditionally feasible through PDC enabling selective information sharing without strategic exposure and without a measurable performance penalty. A critical architectural finding also emerges: MSP independence per competing organization is a prerequisite for PDC to function as a protocol-level rather than application-level confidentiality boundary—a practical design contribution not previously documented in the logistics blockchain literature. Together, these results reposition blockchain from

an informational tool to a governance-enforcing infrastructure for inter-organizational logistics coordination.

Beyond validating governance mechanisms, the study offers methodological and architectural contributions. The documented nine-phase implementation process and four platform failure iterations reveal that the gap between conceptual blockchain architectures and operational reality is substantial—explaining the scarcity of empirical studies. The namespace-based deployment architecture demonstrates that independent per-organization deployment is achievable on standard professional hardware, lowering the infrastructure barrier for university research labs and SME logistics consortia. All implementation artifacts are publicly released to enable independent replication.

These contributions should be interpreted within the study's scope. The network constitutes a proof of concept with simulated organizations, controlled evaluation conditions, and a single-orderer consensus configuration. The study does not claim organizational adoption, long-term operational performance, or production-scale viability. It establishes a feasibility baseline upon which future research can build, where blockchain is perceived as a governance infrastructure rather than merely a transparency-enhancing technology.

Finally, future research should extend this proof of concept toward industry validation through practitioner interviews, expert evaluation, and usability assessment within logistics consortia. Beyond validation, a key research direction lies in extending horizontal collaboration from coordination execution toward decentralized planning and decision-making. Integrating strategic planning, tactical resource sharing, and operational execution within chaincode logic would allow investigation of blockchain not only as a coordination infrastructure, but as a distributed decision-making mechanism in collaborative logistics networks.

Author Contributions: Conceptualization, Y.C.; methodology, Y.C.; software, Y.C.; validation, Y.C., M.A.E.O. and M.A.; formal analysis, Y.C.; investigation, Y.C.; resources, Y.C.; data curation, Y.C.; writing—original draft preparation, Y.C.; writing—review and editing, M.A.E.O. and M.A.; visualization, Y.C.; supervision, M.A.E.O. and M.A.; project administration, M.A.E.O.; funding acquisition, Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Center for Scientific and Technical Research (CNRST), Morocco, through an excellence scholarship awarded to the first author, grant number “7 UH22022”.

Data Availability Statement: The complete implementation artifact supporting this study is publicly available at <https://github.com/yousrachabba/hlf-collaborative-logistics-blockchain> accessed on 21 May 2026, and archived with a persistent identifier at Zenodo (DOI: <https://doi.org/10.5281/zenodo.20325561>). The repository contains chaincode source code (Go), channel and PDC configuration files, Kubernetes deployment manifests for all 15 organizational namespaces, network policies, Hyperledger Caliper benchmark configurations and workload modules, a full cluster state backup, and phase-by-phase documentation covering the complete deployment process from environment setup through performance benchmarking. All materials are released under the Apache License 2.0 to enable independent replication and extension of the results reported in this study.

Acknowledgments: The authors gratefully acknowledge Bouragba and Ziyati of Ecole Supérieure de Technologie at University Hassan II Casablanca for their technical assistance in the early phases of the Hyperledger Fabric network architecture and deployment. During the preparation of this manuscript, the authors disclose that AI writing assistance tools, namely OpenAI's ChatGPT (GPT-5) and Anthropic's Claude (Sonnet 4.5), were used for language editing, structural refinement and formatting purposes. The research design, implementation, data collection, analysis, and intellectual contributions remain entirely the work of the authors, who take full responsibility for the accuracy, integrity, and originality of the final manuscript content.

Conflicts of Interest: The authors declare no conflicts of interest. Also, the funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

SME	Small and Medium Enterprise
PDC	Private Data Collections
MSP	Membership Service Provider
DSR	Design Science Research
DSRM	Design Science Research Methodology
CA	Certificate Authority
HLF	Hyperledger Fabric
LTS	Long-Term Support
IPFS	InterPlanetary File System
DNS	Domain Name System
TLS	Transport Layer Security
RBAC	Role-Based Access Control
TPS	Transactions Per Second
SDK	Software Development Kit
CCaaS	Chaincode as a Service
ERP	Enterprise Resource Planning
EDI	Electronic Data Interchange
MVCC	Multi-Version Concurrency Control
CLI	Command Line Interface
PoC	Proof of Concept
gRPC	Google Remote Procedure Call
CRL	Certificate Revocation List
HSM	Hardware Security Module
mTLS	Mutual Transport Layer Security
PVC	Persistent Volume Claim
DP	Design Principle

Appendix A

Appendix A.1. Algorithm A1: Vertical Shipment Lifecycle with State-Machine Enforcement

Algorithm A1: InitiateShipment/ConfirmReceipt (vertical factory → warehouse)

Input: shipmentID, originFactoryID, destWarehouseID, productID, quantity

Context: ctx (transaction context with client identity and ledger stub)

```

1: callerMSP ← cid.GetMSPID(ctx)
2: if callerMSP does not own originFactoryID then
3:   return error "ACCESS_DENIED: only <ownerMSP> can endorse this proposal"
4: factory ← GetState(originFactoryID)
5: if factory = nil then return error "entity <originFactoryID> not found"
6: if factory.availableCapacity < quantity then
7:   return error "proposing factory no longer has sufficient capacity"
8: warehouse ← GetState(destWarehouseID)
9: if warehouse = nil then return error "requesting warehouse <destWarehouseID>
   does not exist"

```

Algorithm A1 Cont.

```

10: if GetState(shipmentID)  $\neq$  nil then return error "shipment <shipmentID> already exists"
11: shipment  $\leftarrow$  { ID, origin, destination, product, quantity,
    status: "INITIATED", initiatedAt: txTimestamp }
12: factory.availableCapacity  $\leftarrow$  factory.availableCapacity – quantity
13: PutState(originFactoryID, factory) // read-modify-write on shared key
14: PutState(shipmentID, shipment)
15: return success(shipment)

--- ConfirmReceipt (invoked later by destination warehouse) ---
16: callerMSP  $\leftarrow$  cid.GetMSPID(ctx)
17: shipment  $\leftarrow$  GetState(shipmentID)
18: if shipment = nil then return error "shipment <shipmentID> not found"
19: if callerMSP does not own shipment.destination then
20:     return error "ACCESS_DENIED: order <id> belongs to <ownerMSP>"
21: if shipment.status  $\neq$  "INITIATED" then
22:     return error "shipment <id> is not in initiated state (current: <status>)"
23: warehouse.currentStock  $\leftarrow$  warehouse.currentStock + shipment.quantity
24: shipment.status  $\leftarrow$  "DELIVERED"; shipment.confirmedAt  $\leftarrow$  txTimestamp
25: PutState(destWarehouseID, warehouse); PutState(shipmentID, shipment)
26: return success(shipment)

```

Appendix A.2. Algorithm A2: PDC Confidential Write and Cross-MSP Access Control with SHA-256 Commitment

Algorithm A2: StorePrivateCosts/QueryOwnPrivateData/QueryCompetitorPublicData

Context: ctx (transaction context); transient map carries private payload

```

--- StorePrivateCosts (confidential write) ---
1: callerMSP  $\leftarrow$  cid.GetMSPID(ctx)
2: collection  $\leftarrow$  collectionForMSP(callerMSP) // e.g. FacLivMSP  $\rightarrow$  "CostsLIV"
3: if collection = nil then
4:     return error "ACCESS_DENIED: unknown MSP type (caller: <callerMSP>)"
5: privatePayload  $\leftarrow$  GetTransient(ctx)["costs"] // {unitCost, margin}, never on public ledger
6: if privatePayload = nil then
7:     return error "no private data found in <collection> for key <entityID>"
8: PutPrivateData(collection, entityID, privatePayload) // side-DB of owning MSP only
9: if PutPrivateData failed then
10:    return error "failed to store private costs in <collection>"
11: hash  $\leftarrow$  GetPrivateDataHash(collection, entityID) // SHA-256, computed by Fabric
12: publicRecord  $\leftarrow$  GetState(entityID)
13: publicRecord.privateCostsHash  $\leftarrow$  hash // commitment on public ledger
14: PutState(entityID, publicRecord)
15: return success(hash)

```

Algorithm A2 Cont.

```

--- QueryOwnPrivateData (authorized read) ---
16: callerMSP ← cid.GetMSPID(ctx)
17: collection ← collectionForMSP(callerMSP)
18: data ← GetPrivateData(collection, entityID)
19: if data = nil then return error "no private data found in <collection> for key
    <entityID>"
20: return success(data)                                     // plaintext to owning MSP

--- QueryCompetitorPublicData / VerifyPrivacyBoundary (cross-MSP attempt) ---
21: callerMSP ← cid.GetMSPID(ctx)
22: targetCollection ← collectionForEntity(targetEntityID) // e.g. "CostsBRI"
23: // PDC policy OR("<owner>MSP.member") is enforced by Fabric's gossip layer:
24: // a non-owning peer never received the private data, independent of this code.
25: hashOnChain ← GetPrivateDataHash(targetCollection, targetEntityID) // visible
    to all
26: attempt ← GetPrivateData(targetCollection, targetEntityID)
27: if attempt = nil then // caller's peer holds no copy → boundary holds
28:     return { cryptographicBoundary: true, privateDataAccessible: false,
                publicHash: hashOnChain }
29: else
30:     return { cryptographicBoundary: false, privateDataAccessible: true } //
    owner only

```

References

1. Cao, M.; Zhang, Q. Supply Chain Collaboration: Impact on Collaborative Advantage and Firm Performance. *J. Oper. Manag.* **2011**, *29*, 163–180. [\[CrossRef\]](#)
2. Christopher, M.; Holweg, M. "Supply Chain 2.0": Managing Supply Chains in the Era of Turbulence. *Int. J. Phys. Distrib. Logist. Manag.* **2011**, *41*, 63–82. [\[CrossRef\]](#)
3. Dutta, P.; Choi, T.M.; Somani, S.; Butala, R. Blockchain Technology in Supply Chain Operations: Applications, Challenges and Research Opportunities. *Transp. Res. E Logist. Transp. Rev.* **2020**, *142*, 102067. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Cruijssen, F.; Cools, M.; Dullaert, W. Horizontal Cooperation in Logistics: Opportunities and Impediments. *Transp. Res. E Logist. Transp. Rev.* **2007**, *43*, 129–142. [\[CrossRef\]](#)
5. Pomponi, F.; Fratocchi, L.; Tafuri, S.R. Trust Development and Horizontal Collaboration in Logistics: A Theory Based Evolutionary Framework. *Supply Chain Manag. Int. J.* **2015**, *20*, 83–97. [\[CrossRef\]](#)
6. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System; White Paper. 2008. Available online: <https://bitcoin.org/bitcoin.pdf> (accessed on 19 July 2026).
7. Androulaki, E.; Barger, A.; Bortnikov, V.; Muralidharan, S.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Murthy, C.; Ferris, C.; et al. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *Proceedings of the 13th EuroSys Conference, EuroSys 2018*; Association for Computing Machinery, Inc.: New York, NY, USA, 2018. [\[CrossRef\]](#)
8. Helo, P.; Hao, Y. Blockchains in Operations and Supply Chains: A Model and Reference Implementation. *Comput. Ind. Eng.* **2019**, *136*, 242–251. [\[CrossRef\]](#)
9. Wang, Y.; Han, J.H.; Beynon-Davies, P. Understanding Blockchain Technology for Future Supply Chains: A Systematic Literature Review and Research Agenda. *Supply Chain Manag.* **2019**, *24*, 62–84. [\[CrossRef\]](#)
10. Cachin, C.; Schubert, S.; Vukolić, M. Non-Determinism in Byzantine Fault-Tolerant Replication. In *Proceedings of the Leibniz International Proceedings in Informatics, LIPIcs*; Schloss Dagstuhl–Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing: Wadern, Germany, 2017; Volume 70, pp. 24.1–24.16.
11. Christidis, K.; Devetsikiotis, M. Blockchains and Smart Contracts for the Internet of Things. *IEEE Access* **2016**, *4*, 2292–2303. [\[CrossRef\]](#)

12. Li, M.; Shen, L.; Huang, G.Q. Blockchain-Enabled Workflow Operating System for Logistics Resources Sharing in E-Commerce Logistics Real Estate Service. *Comput. Ind. Eng.* **2019**, *135*, 950–969. [\[CrossRef\]](#)
13. Choi, T.M.; Siqin, T. Blockchain in Logistics and Production from Blockchain 1.0 to Blockchain 5.0: An Intra-Inter-Organizational Framework. *Transp. Res. E Logist. Transp. Rev.* **2022**, *160*, 102653. [\[CrossRef\]](#)
14. Treiblmaier, H. The Impact of the Blockchain on the Supply Chain: A Theory-Based Research Framework and a Call for Action. *Supply Chain Manag.* **2018**, *23*, 545–559. [\[CrossRef\]](#)
15. Jensen, T.; Hedman, J.; Henningsson, S. How TradeLens Delivers Business Value with Blockchain Technology. *MIS Q. Exec.* **2019**, *18*, 221–243. [\[CrossRef\]](#)
16. Jovanovic, M.; Kostić, N.; Sebastian, I.M.; Sedej, T. Managing a Blockchain-Based Platform Ecosystem for Industry-Wide Adoption: The Case of TradeLens. *Technol. Forecast. Soc. Change* **2022**, *184*, 121981. [\[CrossRef\]](#)
17. Choi, T.M. Blockchain-Technology-Supported Platforms for Diamond Authentication and Certification in Luxury Supply Chains. *Transp. Res. E Logist. Transp. Rev.* **2019**, *128*, 17–29. [\[CrossRef\]](#)
18. Balci, G.; Surucu-Balci, E. Blockchain Adoption in the Maritime Supply Chain: Examining Barriers and Salient Stakeholders in Containerized International Trade. *Transp. Res. E Logist. Transp. Rev.* **2021**, *156*, 102539. [\[CrossRef\]](#)
19. IBM. *IBM Food Trust: Transforming Food Transparency*; IBM: Armonk, NY, USA, 2022.
20. Chang, S.E.; Chen, Y.C.; Wu, T.C. Exploring Blockchain Technology in International Trade: Business Process Re-Engineering for Letter of Credit. *Ind. Manag. Data Syst.* **2019**, *119*, 1712–1733. [\[CrossRef\]](#)
21. Agrawal, T.K.; Kumar, V.; Pal, R.; Wang, L.; Chen, Y. Blockchain-Based Framework for Supply Chain Traceability: A Case Example of Textile and Clothing Industry. *Comput. Ind. Eng.* **2021**, *154*, 107130. [\[CrossRef\]](#)
22. Durach, C.; Kembro, J.; Wieland, A. A New Paradigm for Systematic Literature Reviews in Supply Chain Management. *J. Supply Chain Manag.* **2017**, *53*, 67–85. [\[CrossRef\]](#)
23. Cho, G.; Go, Y.; Kim, M. A Hyperledger Fabric-Based SBOM Management System for Secure Software Supply Chain Integrity. *Electronics* **2026**, *15*, 2573. [\[CrossRef\]](#)
24. Ravi, D.; Ramachandran, S.; Vignesh, R.; Falmari, V.R.; Brindha, M. Privacy Preserving Transparent Supply Chain Management through Hyperledger Fabric. *Blockchain Res. Appl.* **2022**, *3*, 100072. [\[CrossRef\]](#)
25. Neto, B.; Febra, J.; Durães, P.; Domingues, P.; Antunes, C.M.; Maximiano, M.; Gomes, R.; Távora, V.; Remédios, O. Supply Chain Traceability by Recording EPCIS Data on Hyperledger Fabric. *Procedia Comput. Sci.* **2026**, *278*, 709–716. [\[CrossRef\]](#)
26. Rahaman, M.; Tabassum, F.; Arya, V.; Bansal, R. Secure and Sustainable Food Processing Supply Chain Framework Based on Hyperledger Fabric Technology. *Cyber Secur. Appl.* **2024**, *2*, 100045. [\[CrossRef\]](#)
27. Keo, R.; Firdaus, M.; Rhee, K.-H. A Secure Rubber Supply Chain Management System Based on Hyperledger Fabric Blockchain: A Use Case in Cambodia. *Front. Blockchain* **2025**, *8*, 1474329. [\[CrossRef\]](#)
28. Ni, L.; Irannezhad, E. Performance Analysis of LogisticChain: A Blockchain platform for Maritime Logistics. *Comput. Ind.* **2024**, *154*, 104038. [\[CrossRef\]](#)
29. Zhou, Z.; Feng, Y.; Sakurai, K. Design and Implementation of a Trusted Food Supply Chain Traceability System with Incentive Using Hyperledger Fabric. *Computers* **2026**, *15*, 108. [\[CrossRef\]](#)
30. Duman, E.; Aydoğan, E. Enhancing Traceability and Reliability in Cold Chain Logistics Through Hyperledger Fabric and IoT. *Appl. Sci.* **2025**, *15*, 12149. [\[CrossRef\]](#)
31. Sayar, A.; Kurtbaş, M.C.; Sancaktar, Ç.; Kabak, R.D.; Çakmak, Ş. Application of Hyperledger Blockchain Technology to Logistics Supply Chain with IoT. *Procedia Comput. Sci.* **2025**, *252*, 814–823. [\[CrossRef\]](#)
32. Ismail, S.; Othman, B.; Reza, H.; Teshome Hunde, E. Towards an Agentic AI-Enabled Blockchain-Based Fish Supply Chain Using Hyperledger Fabric. *Electronics* **2026**, *15*, 1916. [\[CrossRef\]](#)
33. Kutubayeva, K.; Razaque, A.; Rai, H.M. Enhancing Pharmaceutical Supply Chain Transparency and Security with Blockchain and Big Data Integration. *Procedia Comput. Sci.* **2025**, *259*, 1511–1522. [\[CrossRef\]](#)
34. Naga Sudha, C.M.; Nayahi J, J.V. TrackChain: Hyperledger Based Pharmaceutical Supply Chain–Resource Utilization Perspective. *Heliyon* **2024**, *10*, e23250. [\[CrossRef\]](#) [\[PubMed\]](#)
35. Zineb, K.I.; Mohamed, L.; Hamid, H.; Meryem, Y.; Ghita, L.; Hanae, H. BlockSupply: Blockchain-Based Logistics Traceability Solution. *Softw. Impacts* **2024**, *21*, 100666. [\[CrossRef\]](#)
36. Qiao, M.; Chen, X.; Zhou, Y.; Mok, P.Y. Blockchain-Driven Innovation in Fashion Supply Chain Contractual Party Evaluations as an Emerging Collaboration Model. *Blockchain Res. Appl.* **2025**, *6*, 100266. [\[CrossRef\]](#)
37. Chou, C.-C.; Richard Hwang, N.-C.; Li, C.-W.; Wang, T.; Wang, Y.-Y. Implementing a Multichain Framework Using Hyperledger for Supply Chain Transparency in a Dynamic Partnership: A Feasibility Study. *Comput. Ind. Eng.* **2023**, *175*, 108906. [\[CrossRef\]](#)
38. Peffers, K.; Tuunanen, T.; Rothenberger, M.A.; Chatterjee, S. A Design Science Research Methodology for Information Systems Research. *J. Manag. Inf. Syst.* **2007**, *24*, 45–77. [\[CrossRef\]](#)
39. Hevner, A.R.; March, S.T.; Park, J.; Ram, S. Design Science in Information Systems Research. *MIS Q.* **2004**, *28*, 75–106. [\[CrossRef\]](#)

40. Williams, H.P. *Model Building in Mathematical Programming*, 5th ed.; John Wiley & Sons: Chichester, UK, 2013; ISBN 9781118443330.
41. Caldarola, F.; d'Atri, G.; Zanardo, E. Neural Fairness Blockchain Protocol Using an Elliptic Curves Lottery. *Mathematics* **2022**, *10*, 3040. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.