AI Data Readiness

Moving to a Unified, Real-Time Information Fabric

12/01/2026

Expert Opinions Still Matter

In the era of AI, where knowledge and analysis are readily available at your fingertips, we still believe expertise and experience matters!

This white paper is not written from theory alone, nor from a single technical viewpoint. It is shaped by more than six decades of combined, hands-on experience designing, building, and operating enterprise-scale systems across industries, geographies, and technology generations. Our perspective is grounded in what works in practice, what fails under real-world complexity, and what it truly takes to move from ambition to execution.

The thinking behind this paper brings together three complementary domains that are rarely addressed as one: Enterprise Integration, Data Platforms and Analytics, and AI and Applied Intelligence, all anchored in deep Software Engineering experience. This convergence matters. Many AI initiatives fail not because models are weak, but because integration architectures cannot deliver real-time context, data platforms lack consistent semantics, and governance cannot keep pace with automation. We have encountered these challenges repeatedly, from different angles, and at enterprise scale.

One perspective is rooted in decades of building and scaling data platforms, business intelligence, advanced analytics, and AI solutions for large organizations. It reflects the full evolution from traditional BI to modern lakehouses, feature stores, and production-grade AI, with a consistent focus on how data foundations either enable or constrain transformation. This experience brings a clear understanding of how AI systems are trained, deployed, and governed, and why alignment between historical data and real-time signals is essential for trustworthy AI.

Another perspective comes from long-standing leadership in enterprise integration, spanning multiple generations of integration technologies. From ESBs to APIs, and from message queues to global event meshes, this lens focuses on how systems actually communicate, where latency and coupling arise, and why event-driven architectures are critical for real-time operations. It reflects years of treating integration as a product, not a project, and of building the connective tissue that allows enterprises to act as coherent systems rather than disconnected silos.

The third perspective bridges software engineering, enterprise architecture, and applied AI, with a strong emphasis on moving AI from experimentation into production. It brings experience in building AI systems that must operate under real enterprise constraints: governance, security, reliability, and regulation. This viewpoint is pragmatic and execution-driven, focused on embedding intelligence into operational flows while balancing innovation with control.

What unites these perspectives is realism. We have seen how architectural shortcuts undermine trust, how duplicated logic creates inconsistency, and how fragmented data landscapes become the primary blocker for automation and agentic systems. The Unified Information Fabric described in this paper is not a theoretical ideal. It is a synthesis of lessons learned from large-scale implementations where success must be repeatable and failure is costly.

This paper matters because AI readiness is not an AI problem alone. It is equally an integration challenge, a data semantics challenge, and a governance challenge. Our aim is to help you connect these disciplines, see the full architectural picture, and move forward with an approach grounded in experience rather than hype.

As you read on, you are engaging with a viewpoint shaped by years of designing, fixing, and scaling the systems enterprises depend on every day.

Audience & Outcomes

Audience: CIO, CFO, CDO, Enterprise Architects, Data, AI and Integration leaders.

Keywords: AI Data Readiness, Real-Time Enterprise, Event-Driven Architecture, Active Metadata, GraphQL, Lakehouse, AI Governance, Agentic, Interoperability, Semantic Layer, Digital Resilience, Operational Intelligence

What problems you may be facing?

- Your data lives in disconnected worlds
  Most enterprises still run two separate stacks: an integration world built on Enterprise Service Busses (ESBs), APIs, and message queues, and an analytics world built on warehouses, lakes, and Business Intelligence (BI) tools. Each defines data differently and evolves on its own, making consistent insights hard to deliver.
- AI systems cannot access timely, trusted data
  AI models and Agentic systems need fresh, unified information. In many organizations, data is delayed, inconsistent, and split between operational APIs and analytical pipelines. Perhaps your organization started utilizing Model Context Protocol (MCP), before realizing it doesn't solve the challenges.
- Duplicate logic leads to inconsistent outcomes
  The same business rules are often rebuilt in both integration flows and analytics pipelines. This duplication raises costs and leads to conflicting answers to basic questions.
- Siloed ownership prevents end-to-end accountability
  Integration and analytics teams typically work in separate processes with separate tools. There is no overall view of APIs, events, and datasets that could provide an "end-to-end truth".
- Insights arrive too late to support real-time decisions
  Traditional analytics rely on batch ingestion, so reporting always trails real-world events. When insights arrive hours or days late, they lose any value for automation and operational decision-making.
- No single view across operational and analytical data
  Not all data reaches the Enterprise Data Platform, and operational data locked behind APIs is hard for analytics or AI to consume. This creates fragmented views of the truth and slows any AI initiative that depends on full context.

What you'll gain: A common vocabulary, a reference architecture, and a step-by-step adoption plan to help you align integration and analytics, and put AI into practice on trusted, up-to-date data.

Executive Summary

This paper is written for enterprise data, integration and technology leaders, any other executives responsible for driving AI and data-driven transformation and digital platform modernization. As organizations advance toward intelligent, context-aware systems powered by AI and increased use of the MCP, the pressure on the data foundation for AI has never been greater. BI, Advanced Analytics, and Agentic AI all depend on consistent, real-time, and trustworthy data. Yet, most enterprises still struggle with fragmented integration landscapes, delayed pipelines, and duplicated business logic that slow insight generation and limit AI effectiveness. The path forward lies in reimagining data integration, not just as a technical necessity for Analytics, but as the cornerstone of AI readiness, to ensure that every application, dashboard, and agent can access the same governed, high-quality information at the moment it is needed.

In this new reality, over many years enterprises have built two parallel data worlds:

- **The Integration World:** initially powered by ESBs and message queues, and more recently by micro-services and REST APIs with point-to-point integrations that connect and drive applications.
- **The Analytics World:** built around data warehouses, data lakes, and BI systems that support reporting, planning, and insights.

The result has often been duplicated logic, divergent definitions, and delayed insights. These parallel worlds created a fundamental split between acting on data as it was created ("Data in Motion") and analyzing it after storage ("Data in Rest"). At the same time, in Gartner's Data Management Hype Cycle for 2025, "AI-Ready Data" is identified as a fast mover and at the top of the Peak of Inflated Expectation.

This leaves us with a fundamental question: how do we move forward from here?
This paper explains how to weave integration and analytics into a single, metadata-driven, event-centric fabric that delivers trusted, real-time information for business applications, analytical dashboards, and AI agents alike.

This fabric exposes data from underlying sub-domains and data platforms through a common event broker and provides a unified consumption layer, often via a federated GraphQL Gateway, to ensure data is accessible when needed. The core shift is from a model that primarily values Data in Rest to one that harnesses the power of Data in Motion through a modern integration layer. This paper will also clarify what "graph" means in practice, how Change Data Capture (CDC) and event-driven architecture (EDA) work together, how to handle write-back, and most importantly how to ensure a single set of business logic is implemented once and reused everywhere.

The destination is a unified information landscape: event-driven, graph-aware, governed by active metadata and data contracts, implemented on modern platforms (e.g., Microsoft Azure/Fabric, Databricks, Snowflake, and Open-Source components), and inherently ready for AI.

Where We're Coming From: The Old Split

For most enterprises, today's data challenges have deep historical roots. Over the past two decades, integration and analytics evolved along separate paths, each solving different problems, using different technologies, and governed by different teams.

- Integration focused on connecting applications and enabling transactions in real-time.
- Analytics focused on storing, modeling, and reporting data after the fact.

This dual evolution delivered short-term success but long-term fragmentation. Enterprises now operate two parallel data worlds that rarely align in meaning, timeliness, or ownership. The consequences are visible everywhere: duplicated logic, costly reconciliation, and delayed insights that limit the speed with which AI and automation can act.

Traditional Integration

- Centralized ESB mediated synchronous calls and transformations.
- Rigid orchestration, with governance-heavy change cycles measured in weeks or months.
- Micro-services and REST APIs improved modularity, but led to many synchronous, point-to-point integrations that were brittle and hard to reuse.
- Focus on transactional Data in Motion, but without the context needed for analytics.

Traditional Analytics

- Data extracted nightly or weekly into warehouses, later into data lakes and data platforms.
- Built entirely around Data at Rest, where value emerged only after data was stored and processed.
- Extract, Transform Load (ETL) and Extract, Load, Transform (ELT) pipelines encoded business rules separately from integration flows.
- BI tools like Power BI overlaid semantic models to present metrics.

Two worlds, two implementations of similar logic, and frequent disagreements about "the numbers." Even the latest version of a Data platform often sees the data aggregated and stored, making it historical rather than real-time and separate from the operational APIs that run the business.

The result is more than just technical debt, it is an organizational and informational divide. Integration teams speak the language of APIs and queues, while analytics teams speak in tables and metrics. Both work with the same data, but interpret and transform it differently.

For AI initiatives, this fragmentation becomes an even bigger blocker. Modern Agentic AI systems, including those using emerging standards like the MCP, can technically connect to tools and systems, but they still depend on the enterprise providing timely, consistent, policy-governed data. MCP solves the interface problem, but not the data problem: agents can call an API, but they can't compensate for inconsistent definitions, stale data, missing events, or the lack of a unified semantic layer. Without a coherent information fabric underneath, agents become unreliable, brittle, and difficult to govern. Closing this gap requires rethinking the architecture itself. The goal is to move from two disconnected data worlds to one unified, real-time information fabric where operational, analytical, and AI workloads all operate on the same trusted foundation.

Why Change Now: Real Time, AI, and Agents

Agentic systems fundamentally redefine how enterprises interact with data. Unlike traditional analytics or static AI models, AI agents can operate continuously, perceiving, reasoning, and acting across dynamic data ecosystems. These agents depend on a constant feedback loop where data is not just consumed but also updated, triggering new events and actions. This creates a bidirectional data fabric where integration, context, and intelligence must co-exist in real time. In this new landscape, the boundary between operational systems, analytical platforms, and AI models dissolves. Each becomes part of a self-adaptive, event-driven network that enables organizations to move from data consumption to data participation, where every application, model, and agent contributes to the collective intelligence of the enterprise. The shift toward Agentic operations therefore demands not just faster data, but smarter, context-rich integration that can evolve alongside the agents it serves.

The historical split between integration and analytics is no longer sustainable. In an era where decisions and customer interactions unfold in seconds, the old model - moving data from systems into warehouses for later reporting - creates unacceptable delays. Enterprises can't afford to wait for insight; they need context and intelligence embedded directly into their operations. This marks a fundamental turning point: from analyzing what happened to responding while it happens.

Drivers for change:

- AI/Agent use cases (assistants, copilots, automation) depend on fresh data and consistent rules, both for reading and writing back to systems of record.
- Customer experience demands instant context (inventory, eligibility, personalization) during the interaction. This requires acting on Data in Motion, not just querying historical Data in Rest.
- Operational resilience requires near real-time signals across supply chain, finance and other relevant divisions. A driver for architectural change is the competitive imperative to act on events as they happen, not hours or days later.

These needs push us from siloed batch pipelines to a streaming-first, event-driven posture that feeds both applications and analytics simultaneously. Still, Analytical Data Platforms are needed as Machine Learning (ML) models, part of the Agentic flow, need training on historical Data in Rest.

What's emerging is a new expectation: data should be continuously available, context-rich, and ready for automated reasoning with AI Agents. This is not only a shift in data architecture but in business tempo. Organizations that can sense, decide, and act in real time will outpace those still bound by nightly batches and asynchronous hand-offs. Real-time integration is no longer optional, it is the foundation of AI-driven competitiveness.

Defining the Building Blocks of the Unified Information Fabric

Before defining the new architecture, establishing a common vocabulary is essential. Many of the technologies shaping modern data integration - events, graphs, semantic layers, and metadata - are often discussed in isolation. Here, we explain them in plain language and show how they interlock to enable real-time, AI-ready data flows across the enterprise.

a. Event-Driven Architecture (EDA)

Systems publish events (e.g., *OrderPlaced*, *InvoicePaid*). Other systems subscribe and react asynchronously. Events are durable, ordered per key, and re-playable for analytics. This is the foundational pattern for creating and consuming Data in Motion at enterprise scale.

It's strongest for real-time updates, decoupling producers/consumers, auditing history, and driving both app workflows and analytics ingestion. EDA is the nervous system of the modern enterprise. By treating every change as an event, organizations can distribute knowledge instantly across systems and keep every part of the business synchronized without direct dependencies.

b. Change Data Capture (CDC)

CDC streams database changes (inserts, updates, deletes) without modifying existing applications. It is especially valuable when legacy systems cannot emit domain events directly. CDC works well for populating lakes and warehouses, and for keeping caches or materialized views synchronized with operational systems.

The Outbox pattern strengthens this approach by ensuring events are captured reliably and without dual-write risks. Instead of having an application update its database and publish an event separately, the application writes the event into a dedicated outbox table within the same database transaction as the business update. A CDC connector then reads from this outbox table and publishes the event to the event mesh. This guarantees that if the business change is committed, the event is published exactly once, even during failures or retries.

Together, CDC and the outbox pattern form a bridge between older systems and modern event-driven architectures. They bring legacy databases into the real-time world without rewriting them, ensuring that every relevant data change becomes visible across both integration and analytics platforms.

c. Three Meanings of Graph in Integration

- **Knowledge Graph (business graph):** a graph of customers, products, orders, relationships, and policies, used for 360° views, recommendations, fraud identification, and lineage of meaning.
- **Topology/Lineage Graph (tech graph):** a live map of systems, APIs, datasets, pipelines, and events, used for impact analysis, dependency management, and change control.
- **Graph-centric APIs:** a unifying read interface over many services and datasets (not a graph database, but a graph query model). For example, a GraphQL Gateway can act as a central point of data consumption, using resolver functions to aggregate and resolve data from various sub-domains and legacy platforms (like a Data Platform) on demand. This avoids having to pre-aggregate all data and allows consumers to fetch exactly what they need.

In practice, "graph" does not mean one specific technology, it describes a way of understanding relationships. Graph concepts connect business meaning, technical lineage, and access patterns, forming the connective tissue of the Unified Information Fabric.

d. Semantic Layers

The semantic layer is the language of trust. When every tool, from dashboards to APIs, speaks the same vocabulary of metrics and definitions, organizations eliminate endless reconciliation and ensure consistency across every consumer of data. It's important to make a distinction between the two levels of semantic layers.

- **Enterprise Conceptual Semantic Model (shared vocabulary):** definitions of entities, metrics, policies, and transformations independent of any tool. This lives in your active metadata and data contracts.
- **Tool-Specific Semantic Model (presentation):** a physical implementation optimized for a BI tool. It should be generated from, or aligned to, the enterprise conceptual model, not invented separately. Examples include the Power BI semantic model and other BI semantic layers.

Enterprise Conceptual Semantic Model should also not be mixed up with the Business Glossary which captures human-readable definitions, while the Semantic Model operationalizes those definitions into entities, relationships, metrics, and policies that can be executed by integration, analytics, and AI systems. The semantic model is the foundation for data contracts, federated queries, shared metrics, and consistent event schemas across the enterprise.

e. Active Metadata & Data Contracts

Active metadata is machine-readable, versioned, and drives automation (code generation, validations, routing). Data contracts specify schemas, semantics, SLAs/SLOs, quality rules, and access policies for a data product or event stream. They enable producers and consumers across both integration and analytics to rely on the same definitions.

Together, active metadata and contracts turn documentation into execution. They provide the shared control layer that enforces consistency and quality automatically, an essential foundation for scaling AI safely and reliably.

These five concepts, events, CDC, graphs, semantics, and active metadata, form the vocabulary of a modern AI-ready data ecosystem. They work together to transform static, siloed systems into a living, connected network of real-time intelligence. In the next chapter, we show how these principles converge in a reference architecture of the Unified Information Fabric.

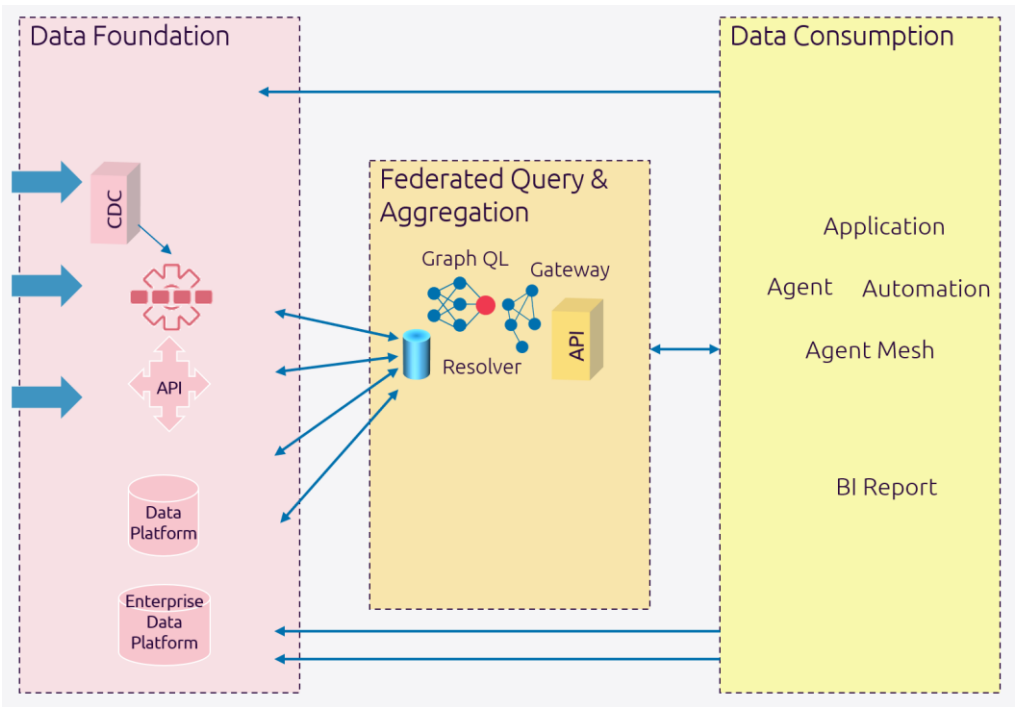
The Unified Information Fabric Reference Architecture

The Unified Information Fabric is not a single product, but an architectural approach. It connects every data-producing and data-consuming system through shared contracts, active metadata, and event-driven integration. You can think of it as the enterprise's real-time information nervous system, one that keeps operational and analytical data flows in sync while applying the same rules and governance everywhere.

At a high level, you can visualize the architecture as three consecutive layers:

- A **Data Foundation Layer** to the left, where data is created and stored in systems, shared via APIs, events, and data platforms.
- A **Federated Query & Aggregation Layer** in the middle, that knows how to find, combine, and reshape data from many places.
- A **Data Consumption Layer** to the right, where applications, analytics tools, and AI agents use the information.

Figure 1 – Illustration of the three Architecture Layers



The Data Consumption Layer can access data in two ways:

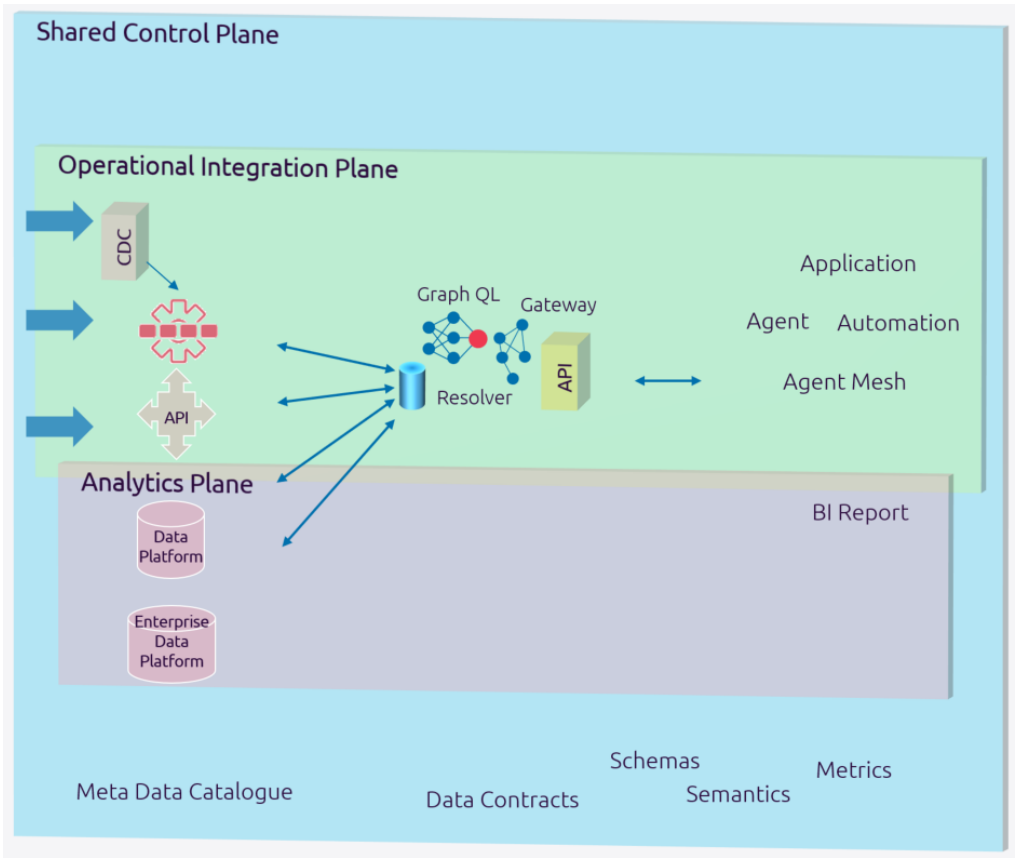
- Through the Federated Query & Aggregation Layer, when a consumer wants a unified, secure, and simplified view across many systems.
- Directly from the Data Foundation Layer, when a specialist system or pipeline needs low-level or high-volume access for a specific purpose.

Cutting across these layers are three important planes:

- A **Shared Control Plane,** that carries contracts, semantics, policies, and lineage.
- An **Operational Integration Plane,** for real-time, transactional Data in Motion.
- An **Analytic Data Plane,** for large-scale analysis, Lakehouse storage, and ML.

Together, these layers and planes turn a fragmented landscape into a single, coherent fabric.

Figure 2 – Illustration of the three Architecture Planes



a. Data Foundation Layer

The Data Foundation Layer is where data is first created, updated, and stored. It contains the business domains and their systems of record: finance, supply chain, sales, customer service, manufacturing, and so on.

In this layer, each domain exposes its information as data products. A data product is a well-defined, reusable set of assets such as:

- An API that returns customer details.
- An event stream that publishes order status changes.
- A Data Platform that holds curated customer and order history.

The Enterprise Data Platform (for example a data lakehouse or warehouse) also lives here. It stores aggregated and historical data that may combine information from many sub-domains.

Key elements in the Data Foundation Layer include:

- **APIs from domain systems:** These expose core business capabilities, such as "create order," "get customer profile," or "calculate price," and provide structured access to operational data.
- **Event Broker / Event Mesh:** This is the engine for Data in Motion. It distributes live updates and business events across the enterprise so that other systems, caches, and analytics can react immediately.
- **Data Platform:** This is the environment where data is aggregated and stored for historical analysis. It is optimized for complex queries, compliance archiving, and large-scale ML model training.

This foundation establishes the sources of operational and analytical truth. Every subsequent layer builds on this, ensuring that live events and historical data remain synchronized and interpretable through a single semantic model.

b. Federated Query & Aggregation Layer

In many organizations, one of the biggest problems is that each consumer must know where data lives and how to call each system. This leads to fragile point-to-point integrations and a lot of duplicated effort. The Federated Query & Aggregation Layer solves this by acting as a single entry point for reading data, even though the data may come from many underlying systems and platforms.

A common way to implement this, is with GraphQL:

- GraphQL started at Facebook/Meta and has become a widely adopted open standard for building APIs that allow clients to ask for exactly the data they need.
- Instead of calling many REST APIs (or event streams or data sources) and stitching the responses together, a client sends one GraphQL query, and the GraphQL server knows how to fetch and combine the data behind the scenes.

In the context of the Unified Information Fabric, a GraphQL Gateway plays the role of the Federated Query & Aggregation Layer:

- It provides a single, logical endpoint for data consumption.
- It uses resolver functions to fetch, aggregate, and resolve data from the underlying systems.
- It can pull live Data in Motion from domain APIs and events, and historical Data in Rest from the Data Platforms, as needed for a specific query.

For example, a single GraphQL query for a "Customer 360" view may cause the gateway to:

- Call the CRM API for basic customer details.
- Read recent order events from the event mesh.
- Fetch risk scores and lifetime value from the Enterprise Data Platform.

To the consumer, this looks like one simple, consistent data model. Behind the scenes, it is an orchestration across the fabric. This layer acts as the fabric's unified lens. It hides the complexity of the underlying systems and lets traditional business applications, analytical dashboards, AI agents and MCP tools query across systems "as if they were one," while still respecting security, data freshness, and cost boundaries.

c. Data Consumption Layer

At the top of the architecture is the Data Consumption Layer. This is where value is realized, because this is where people and machines actually use the information. It may be for traditional analytics workloads, as well as real-time Agentic AI usage.

Typical consumers include:

- Business applications such as Customer Relationship Management (CRM), Enterprise Resource Planning (ERP) extensions, and customer portals, demanding real-time data access.
- Analytics tools such as BI dashboards, notebooks, and self-service data exploration.
- AI and Agentic systems, including copilots, autonomous agents, and Robotic Process Automation (RPA) bots that need to read and sometimes write data while executing business processes.
- Operational tools, such as monitoring systems or workflow engines.

These consumers can access data in two ways:

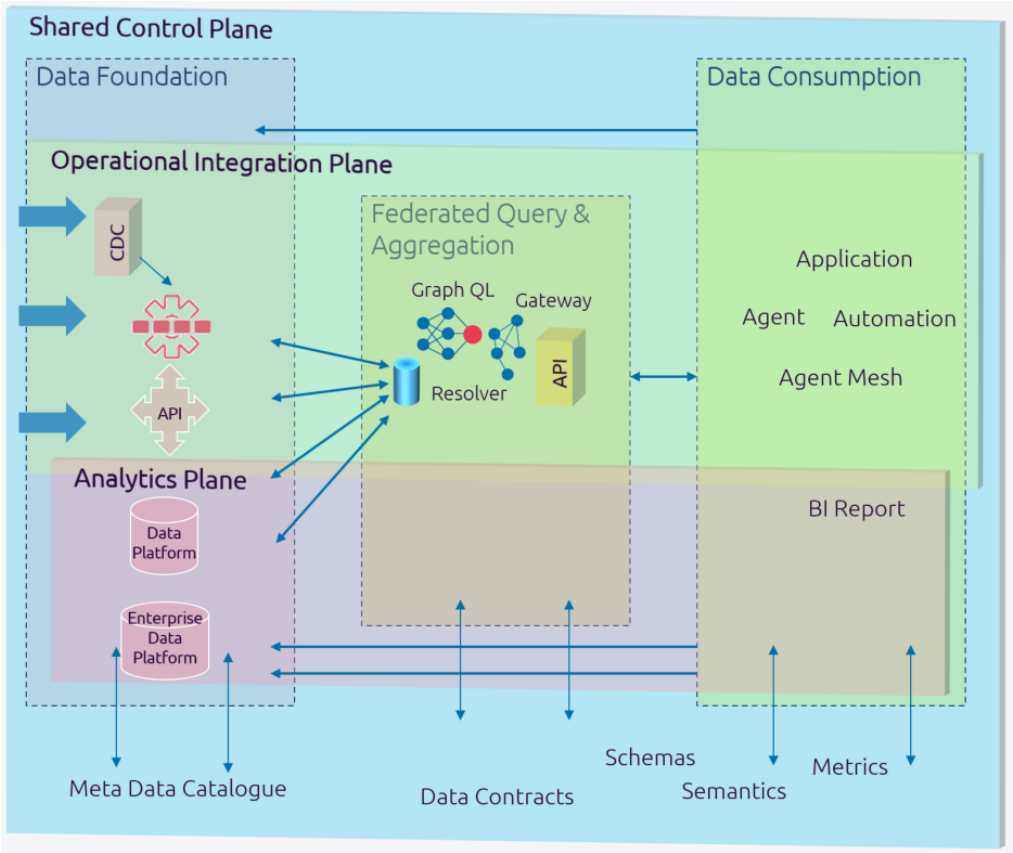
- Through the Federated Query & Aggregation Layer, when they need a unified, governed view that spans multiple domains.
- Directly from the Data Foundation Layer, when they need high-throughput, specialized, or batch access to a specific system or dataset.

For example:

- An AI agent that needs a rich, cross-domain context for a customer will typically use the GraphQL Gateway.
- A nightly financial consolidation job might read directly from curated tables in the Enterprise Data Platform.
- A low-latency trading system may integrate directly with a dedicated event stream.

The key is that everything they access is still governed by the same contracts, semantics, and policies defined in the fabric, this as a result of the Shared Control Plane.

Figure 3 – Illustration of the Layers and Planes coming together



d. Shared Control Plane

The Shared Control Plane is what makes the fabric truly unified. It is the layer of active metadata and policies that sits beside, inside and above all other layers and planes, and it governs how data is described, secured, and monitored. Instead of storing rules in documents and spreadsheets, the control plane makes them machine-readable and executable.

Key components include:

- **Active Metadata Catalog:** A searchable catalog of entities, relationships, lineage, policies, and owners. It is continuously updated from APIs, pipelines, schemas, and code repositories. This allows anyone to see where data comes from, how it is transformed, and who is responsible.
- **Schema Registry & Contract Repository:** A versioned store for schemas and contracts that define events, APIs, and tables. These contracts specify structure, meaning, quality expectations, and access rules.
- **Semantic Definitions & Metric Store:** A central place where business metrics (such as Net Revenue or Active Customer) and key entities are defined once. These definitions are reused by BI tools, APIs, and AI agents to ensure everyone (and everything) calculates numbers the same way.

- **Policy & Access Management:** Encoded rules for classification, masking, and entitlements that apply consistently to APIs, events, and analytic data. This ensures that sensitive data is protected wherever it flows.
- **Observability:** End-to-end lineage, data quality checks, SLOs, and incident routing. This lets teams quickly detect issues, understand impact, and resolve problems.

The Shared Control Plane provides the guardrails that bind all other layers together. It ensures that every data product, API, stream, and pipeline adheres to the same semantic and governance model automatically.

e. Operational Integration Plane

The Operational Integration Plane focuses on keeping the day-to-day business running in real time. It is all about Data in Motion: how changes in one system are communicated to others, and how commands are processed safely. This plane spans the Data Foundation Layer and feeds both the Federated Query Layer and the Analytic Plane.

Key elements include:

- **Domain APIs and Services:** These expose business capabilities such as “create shipment,” “approve claim,” or “update inventory.” They are the canonical way to change state in systems of record.
- **Event Streaming / Event Mesh:** When something important happens (an order is placed, a payment fails, a sensor crosses a threshold), it is published as an event. Other systems subscribe and react. The mesh supports replay and fan-out so multiple consumers can use the same event.
- **CDC Connectors:** Change Data Capture bridges older systems that cannot publish events directly. They stream changes (often via the Outbox pattern) into the event mesh to keep the rest of the enterprise in sync.
- **Orchestrations & Sagas:** Many business processes involve several steps across multiple systems, such as order-to-cash or claims handling. Sagas coordinate these flows, including compensating actions when something goes wrong.

This Operational Integration Plane enables the business to operate in real time. It is where systems and APIs continuously exchange events, ensuring that operational processes, from transactions to customer interactions, always reflect the latest state.

f. Analytic Data Plane

The Analytic Data Plane focuses on turning raw data into insight and models. It builds on the Operational Integration Plan with the APIs and events in the Data Foundation Layer and prepares them for analytics, BI, and ML.

The data platform architecture is lakehouse-style because it blends the strengths of data warehouses (reliable tables, strong schema, governance) with the flexibility of data lakes (scalability, support for many data types, batch and streaming).

Key responsibilities include:

- **Streaming and Batch Ingestion:** The same events and CDC streams that drive operations are ingested into bronze/silver/gold zones in the lakehouse. Bronze holds raw data, silver standardizes and cleans it, and gold provides curated, analytics-ready datasets.
- **Transformations as Code:** Data transformations are expressed as code (SQL, ELT pipelines, or streaming jobs) that reuse logic from data contracts in the Shared Control Plane. This means the same business rules used in APIs and events can also be applied in analytics and ML.
- **Materialized Views and Features:** The plane creates optimized tables, views, and ML features for BI dashboards, data science notebooks, and online serving systems.

Here, the same events and contracts that drive operations also feed Analytics and AI. Treating definitions as code for both the Operational Integration Plane and the Analytic Data Plane is a "shift-left" move that eliminates semantic drift between them. It ensures that when a KPI or model result is shown in a dashboard, it matches what applications and agents see in real time.

In summary, the Unified Information Fabric replaces a world of isolated systems and one-off integrations with a layered, governed, and event-aware architecture. Data is created once, described once, and then safely reused everywhere: in applications, in analytics, and by AI.

Making Logic Truly Shared and Online

Architecture alone doesn't eliminate duplication. Consistency must be enforced through how logic is defined, generated, and reused. In traditional models, each system or team re-implemented similar business rules in slightly different ways, leading to drift and costly rework. The Unified Information Fabric, and especially the Shared Control Plane, changes that by treating logic as code, metadata, and contract, defined once and applied everywhere. This section describes how that principle becomes operational.

a. Define Once, Generate Many

To make logic truly shared, the first step is to describe it once in a form that both humans and machines can use. From there, the same definitions can be compiled into the different technical artifacts that applications, analytics, and AI need.

- Describe entities, metrics, and rules in the contract/metadata layer.
- Generate: (a) BI models, (b) streaming transformations, (c) API schemas, (d) policy configurations.
- Keep definitions versioned; pipeline code imports these packages at build time.

This is the foundation of agility. By moving business definitions into machine-readable contracts, teams can regenerate models, pipelines, and APIs automatically, ensuring every consumer of data, whether human or AI, operates on the same definitions without manual reconciliation.

b. Patterns for Reuse

Once logic is defined centrally, it needs to be reused consistently across different types of workloads. Instead of each team re-coding the same rules, they draw from a common library of transformations, metrics, and policies.

- **Transformation libraries:** common User-Defined Functions (UDFs) / rules packaged as libraries for both batch and streaming jobs.
- **Metric layer ("headless BI"):** central metric definitions used by dashboards, notebooks, and APIs.
- **Policy-as-code:** access control and classification checked consistently in both Operational Integration and Analytic Data planes.

Reusable logic turns governance from a constraint into an accelerator. Teams no longer have to interpret definitions or re-implement calculations, they simply import and extend shared libraries. This shortens delivery cycles, lowers maintenance costs, and ensures every insight and transaction is aligned to the same truth.

c. Avoiding Divergence

Even with shared definitions and reusable patterns, things can drift over time if changes are not controlled. The fabric therefore needs guardrails that automatically check whether new code still respects the agreed contracts and semantics.

- **Contract tests** on both the integration and analytics sides enforce schema/metric consistency.
- **Lineage + impact analysis** from the graph highlights ripple effects before deployment.
- **Approval gates**: CI/CD requires contract compliance before merging changes.

Preventing drift is where automation matters most. By embedding checks and lineage validation directly into the CI/CD process, the organization ensures that every data product, model, or API remains compliant with its contract, before it reaches production. This transforms governance from after-the-fact inspection into real-time assurance.

How CDC + Events + APIs Work Together

The Unified Information Fabric depends on three complementary integration patterns working together: APIs for commands and queries, events for real-time notifications, and CDC for capturing changes from systems that can't yet publish events natively. When combined, they create a fully synchronized, bidirectional data flow between operational systems, analytic platforms and AI platforms, eliminating latency and duplication while keeping the enterprise consistent in real time.

a. Read Paths

A read path ensures every consumer, applications, dashboards, and AI agents, has immediate, consistent access to the latest state of the business. Events push data where it's needed; CDC backfills or synchronizes from systems that were never designed for streaming. The GraphQL gateway ties it together as the single point of truth for reading across both Data in Motion and Data in Rest.

- Domain events: subscribers update caches, search indexes, and analytic stores.
- CDC fills gaps where systems can't emit events; outbox ensures transactional consistency.
- A GraphQL Gateway offers unified query access for apps and agents, pulling live data from operational sub-domains or historical data from the Data Platform through a single interface.

Together, these mechanisms form a real-time "read fabric" across the enterprise. Operational systems, caches, and analytics stay continuously aligned, and the same data is available everywhere, with full lineage and version control.

b. Write-Back Patterns

Reading data consistently across systems is only half of the equation. A real-time enterprise also needs to write changes back into operational systems in a safe, governed, and auditable way. This becomes especially important with AI agents, automated decisions, and analytical insights that need to influence business processes.
Write-back must follow clear patterns because uncontrolled writes can easily break data integrity, introduce conflicting states, or bypass validation rules. The Unified Information Fabric ensures that all updates, regardless of whether they originate from humans, applications, or AI, flow through well-defined and traceable mechanisms.

Below are the primary write-back patterns in the fabric and the role each plays:

- Command via API (synchronous)
  This is the canonical and safest way to change state in systems of record. A request goes through a domain API, is validated, persisted, and emits an event afterward. It is used when immediate confirmation and strong consistency are required.
- CQRS + Events (asynchronous)
  Commands are submitted to domain services, validated, and then expressed as events. These events update read models and analytic stores in real time. This pattern supports high-volume, distributed workloads where write and read paths need to scale independently.
- Reverse ETL / Operational Analytics
  Curated, governed insights such as risk scores or recommendations are fed back into applications through events or APIs. This allows analytical outcomes to directly augment operations, but still within a controlled and governed interface.
- Anti-pattern: Dual Writes
  A critical warning: analytics tools should never write directly into operational databases. This bypasses business logic and validation, risks data corruption, and creates systems of record that cannot be trusted. All write-backs must go through APIs or event-driven upserts with idempotency and auditability.

Together, these patterns close the loop between insight and action. They allow the enterprise to benefit from automation and AI-driven decisions, while maintaining transactional safety, lineage, and governance. Whether the change originates from a user, a business rule, or an AI agent, it always flows through controlled pathways that preserve integrity and trust.

Graph in Practice

The concept of "graph" can sound abstract until you see it in action. In a unified information fabric, graphs aren't just data structures; they are living maps of how the enterprise operates. They show how customers, products, and systems connect, how data flows across boundaries, and how every change propagates through the organization. These graphs bring both business meaning and technical control into one navigable network.

a. The Business Knowledge Graph

- Nodes: customers, accounts, orders, products, suppliers, locations.
- Edges: owns, purchases, shipsTo, partOf, relatedTo.
- Use: 360° views, churn prediction, fraud rings, recommendation, and policy inference.

The business knowledge graph provides the context layer for decision-making. It reveals relationships that are hidden in traditional tables, allowing teams to ask richer questions such as: "Which suppliers are linked to delayed shipments?" or "Which customers are likely to churn based on network influence?" For AI models and agents, this graph becomes the foundation for reasoning about real-world relationships.

b. The Topology & Lineage Graph

- Nodes: systems, APIs, topics/queues, pipelines, datasets, reports.
- Edges: produces, consumes, transforms, dependsOn.
- Use: change impact, SLO ownership, regulatory traceability, platform capacity planning.

While the business knowledge graph captures meaning, the topology and lineage graph captures mechanics. It shows how data moves across systems and who depends on what. For IT and governance teams, it becomes the control center for understanding impact before change. Together, these two graphs connect "what the business means" with "how the technology works."

c. Keeping the Graph Fresh

A graph is only useful if it stays current. Continuous harvesting from pipelines, APIs, and repositories ensures that the map always reflects reality. This allows every stakeholder, from developers to compliance officers, to navigate a live representation of how data and systems actually behave. To accomplish this, harvesters scrape schemas, repositories, CI/CD, runtime telemetry, streaming platforms, and BI catalogs.

- Events from platform components update the graph automatically; no manual diagrams.
- The same graph powers search, approvals, incident response, and agent navigation.

d. Graph & Agents

Agents can traverse the knowledge graph to resolve an identity, enrich context, or find the authoritative API to write back, with policies attached.

When AI agents operate on the fabric, they use the graph as their compass. It guides them to the right data source, the correct definition, and the authorized way to act. This ensures that automation happens safely, respecting lineage, policies, and data contracts.

Governance Without Friction

In most organizations, governance has historically been seen as a blocker, a necessary control that often slows change. In the Unified Information Fabric, governance becomes a built-in capability of the platform itself. It's no longer about reviews and committees, but about codified policies, automated validation, and transparent ownership. The goal is simple: to keep innovation fast while keeping data trustworthy and compliant. Governance accelerates change instead of blocking it.

- **Data product ownership:** every dataset/event/API has an accountable owner with SLOs.
- **Quality as code:** expectations/validation rules attached to contracts; failures trigger alerts.
- **Privacy & security:** classification, masking, and purpose-based access encoded once and enforced across planes.
- **Lifecycle management:** deprecation rules, schema evolution policies, and consumer notification via contracts.

Implementing governance as code requires a mindset change: from centralized control to embedded responsibility. Domain teams become accountable for data quality, SLOs, and compliance; the platform team provides the automation to make that responsibility practical.

This shift moves governance from review boards and spreadsheets to runtime validation and continuous assurance. It lays the foundation for trusted AI and transparent decision-making, ensuring that innovation and compliance can move at the same speed. In short, the Unified Information Fabric evolves both Data Mesh and Data Governance into a single, automated, event-aware model. Governance is no longer the gate at the end of the process. It is the framework that keeps data consistent, secure, and AI-ready from the moment it's created.

a. Understanding the Unified Information Fabric in the Light of Traditional Data Mesh

Over the past several years, Data Mesh has emerged as a response to the limitations of centralized data lakes and monolithic analytics teams. It introduced a powerful set of ideas: decentralizing data ownership to business domains, treating data as a product, and implementing federated governance through a self-serve platform.

The Unified Information Fabric builds directly on these principles, but extends them beyond analytics into operational, event-driven data flows and real-time AI enablement.

This section relates familiar concepts from Data Mesh to the broader scope of the Unified Information Fabric. It shows that we're not replacing Mesh thinking; we're expanding it to cover both real-time and analytical domains under one active governance model.

From Data Mesh to Information Fabric

Traditional Data Mesh focused primarily on Data in Rest - how data is produced, shared, and governed once it lands in a lakehouse or data platform. The Unified Information Fabric broadens this scope to include Data in Motion, ensuring the same semantic and governance rules apply to streaming events, APIs, and operational systems as to analytic datasets.

Data Mesh Concept	Unified Information Fabric Equivalent	Key Evolution
Domain data products	Sub-domain data products (APIs, events, tables)	Includes both operational and analytic domains
Federated governance	Shared control plane (active metadata, policies, contracts)	Automated, policy-as-code enforcement
Self-serve platform	Platform enablement layer (streaming, catalog, CI/CD)	Real-time, event-driven infrastructure
Data contracts	Schema + metric + policy definitions reused across planes	Executable metadata, not documentation
Data as a product	APIs, events, and tables all governed as products	Cross-domain reuse by apps, BI, and AI agents

A Broader Scope, Same DNA

The Unified Information Fabric can be seen as "Data Mesh plus Integration Mesh plus Active Metadata Fabric." It maintains Mesh's core idea: business domains owning and serving their data products - but connects those domains via an event-centric backbone, ensuring that operational systems, analytics, and AI share one semantic layer.

This extension brings real-time data and AI into the same governance model, creating what's effectively a "living Data Mesh" where data doesn't just sit waiting to be analyzed, it moves, interacts, and stays compliant along the way. This convergence means that the same Order-Placed event triggers a workflow in the ERP, populates a real-time dashboard, and updates an AI model's feature store, without separate pipelines or duplicated logic.

Federated, Not Fragmented

Just like Data Mesh, the fabric is federated by design: domains maintain autonomy, but within guardrails defined by contracts and active metadata. The platform team's role remains one of enablement, providing reusable templates, lineage observability, and compliance automation. The result is a living, real-time Data Mesh, where the boundaries between integration and analytics dissolve, and data truly becomes a shared, governed enterprise fabric.

b. Understanding the Unified Information Fabric in the Light of Traditional Data Governance

Traditional Data Governance emerged from a compliance and control mindset. It ensured that data was cataloged, classified, and protected, but often as a separate, manual discipline disconnected from engineering workflows. In this model, governance acted as a gatekeeper, slowing down innovation while providing assurance. The Unified Information Fabric redefines governance as an active, automated control layer, a system that both governs and accelerates change.

From Passive to Active Governance

In a traditional set-up, governance was about *knowing* where data lived and *who* could access it. In the fabric model, governance is about *enforcing* correctness, quality, and policy automatically across every data plane: operational, analytical, and AI.

Traditional Governance	Unified Information Fabric Governance	Key Difference
Central catalog of datasets	Active metadata graph continuously harvested from APIs, schemas, pipelines	Always live and connected
Manual stewardship	Policy-as-code, contract validation, automated SLO enforcement	Embedded in CI/CD workflows
Focus on compliance and audit	Focus on operational trust, reliability, and reuse	Governance accelerates delivery
Data quality checks post-hoc	Quality rules attached to data contracts	Prevents issues before deployment
Security via access reviews	Entitlements, masking, and lineage enforced across both planes	Consistent across APIs and analytics

Operating Model, People & Process

Technology alone doesn't deliver transformation. The real differentiator is how teams are structured, governed, and empowered to use it. A unified information fabric succeeds when it is owned by the business, enabled by the platform team, and reinforced by automated controls. The goal is a working model that encourages autonomy without sacrificing consistency.

A key principle is organizing work through product-oriented domain teams. These teams are aligned to business areas such as Sales, Supply, or Finance, and each team owns the APIs, events, and data products that represent their domain. This includes tables, streams, features, and the service levels that accompany them. By placing ownership where the expertise lives, each domain treats its data and integration assets as products with customers, clear outcomes, and measurable SLOs. This structure improves accountability, reduces dependency on centralized bottlenecks, and aligns technology decisions more closely with business value.

Supporting these domain teams is the platform team, whose role is enablement rather than control. The platform team provides shared capabilities such as streaming infrastructure, catalog and metadata services, security foundations, CI/CD templates, and testing harnesses. They maintain the active metadata graph and the self-service tooling that ensures consistency across domains. Instead of approving every change, the platform team supplies automated templates, reusable patterns, and policy enforcement mechanisms that make good practices easy and scalable. This model replaces manual control with embedded governance.

Finally, a well-defined change and compliance process ensures that agility and governance reinforce rather than contradict each other. Work begins with contracts: schemas, metrics, and rules are defined before implementation. CI/CD pipelines include automated checks like schema validation, data quality tests, lineage comparisons, and policy evaluation, so that compliance becomes a continuous part of delivery. When issues occur, lineage and contract metadata guide incident response and enable rapid rollback with clear accountability.

When the operating model combines empowered domain teams with an enablement-focused platform function and automated compliance, the enterprise achieves both speed and safety. Governance becomes part of everyday work, not a final checkpoint. People, process, and platform evolve together around shared definitions and contracts, forming the human foundation of the Unified Information Fabric.

Data Quality & Trust - What Leaders Should Require

No matter how advanced an architecture becomes, it succeeds or fails on trust. Data quality is not only a technical concern but a leadership issue, because it defines how confidently decisions, whether human or AI-driven, can be made. The Unified Information Fabric embeds quality controls directly into the data lifecycle so that assurance is continuous, not reactive. This is done though:

- **Contracted SLAs/SLOs:** for freshness, completeness, accuracy, and availability.
- **Quality checks at boundaries:** for schema conformance, referential integrity, and metric reconciliation vs. system of record.
- **Reconciliation jobs:** for continuous comparisons between operational and analytic truth, and alerts on divergence.
- **Business acceptance tests:** for finance/operations sign off on KPI definitions in the central metric store.

These controls ensure that data remains reliable at every stage, from ingestion to analytics to AI consumption. Service-level objectives and active monitoring replace after-the-fact inspection with real-time validation. Executives can see not only whether data is available, but whether it is trustworthy and why. By uniting operational data, analytics, and AI under shared contracts, the organization gains a single definition of truth. Quality issues can be traced instantly through lineage graphs, and automated alerts make problems visible before they impact business outcomes. This transparency turns data trust into a measurable, reportable capability, which is something boards and regulators increasingly expect.

For leaders, data trust should now be treated as a managed asset, not an assumption. By embedding quality rules, reconciliations, and ownership into the Unified Information Fabric, organizations create conditions for confident decision-making, auditable AI, and resilient operations. In practice, trust becomes the most valuable deliverable of the data strategy.

Platform Landscape - How the Pieces Fit

A unified information fabric isn't a single product or vendor solution. It's an architectural approach built from interoperable components that work together to support shared semantics, event streaming, and active governance. The goal is to combine best-of-breed technologies in a way that feels cohesive to users and consistent to AI systems. What matters most is not the tool itself, but the contracts, metadata, and integration patterns that connect them. The Unified Information Fabric consists of the following components:

- **Streaming/Event Mesh:** managed cloud services or Open Source (for publish/subscribe, replay, ordering, schema registry).
- **CDC & Connectors:** tools that support outbox as well as direct log-based capture.
- **Lakehouse & Storage:** scalable tables with ACID semantics and time travel; batch + streaming ingest.
- **Semantic/Metric Layers:** central definitions reused by BI and APIs.
- **Active Metadata & Catalog:** discovery, lineage, contracts, and policy propagation.
- **API Management & GraphQL Gateway:** consistent gateways for read/write with security and throttling.

These components can be assembled in many ways, depending on enterprise strategy and existing investments. The key is that each element participates in a governed ecosystem: events travel through the mesh, CDC connects legacy systems, the lakehouse stores and enriches data, and the control plane maintains the single source of semantic truth.

This open approach protects the organization from vendor lock-in while enabling best-fit adoption. Each layer remains replaceable as long as it conforms to shared contracts and metadata standards. The result is a platform landscape that supports real-time operations, analytics, and AI within a single governed framework.

In practice, the Unified Information Fabric becomes the connective layer across a heterogeneous stack. It ensures that data produced in one domain, stored in another, and consumed by an AI model somewhere else all speak the same language and follow the same rules. This interoperability is what turns the technology landscape into a true business capability.

AI & Agents on the Fabric

Artificial intelligence delivers value only when it operates on trustworthy, well-governed data. The Unified Information Fabric provides exactly that foundation. By exposing real-time, contextual, and policy-aware information through consistent APIs, events, and graphs, it enables AI models and agents to act as first-class citizens inside the enterprise, safely, transparently, and with full traceability. Instead of interacting with fragmented sources or ad-hoc integrations, agents navigate a governed environment where every dataset, event, and API follows shared semantics and contracts.

AI agents typically rely on a few recurring patterns when working on top of the fabric. Retrieval-Augmented Generation (RAG) becomes grounded in governance because retrieval is guided by the enterprise knowledge graph and policy-aware indices, ensuring that agents only pull from trusted and authorized sources. When agents need to perform actions, they use contracts and lineage to choose the correct API, event stream, or table, guided by the topology graph that maps the authoritative systems. All write-backs occur through validated, controlled commands, not uncontrolled data writes. And when ML models support these agents, feature consistency is preserved through shared definitions: the same semantic metrics and feature logic appear in both training (offline) and inference (online), ensuring that model behavior is stable and explainable.

This link between ML models and the data fabric is especially important. Agents gain intelligence from models trained on historical Data at Rest, typically stored in a lakehouse or analytical platform. If those datasets do not follow the same definitions, semantics, and contracts as real-time Data in Motion, agents will behave unpredictably. A model trained on one definition of "customer," "risk," or "order value" but asked to operate on a different real-time definition will produce inconsistent or incorrect outcomes. By enforcing a single semantic layer across both planes, the Unified Information Fabric ensures that what a model learns during training is identical to what it encounters during real-time decision-making. This alignment is essential for safe, reliable, and auditable AI.

These patterns connect intelligence with infrastructure. RAG becomes explainable because it is sourced from governed datasets. Agents act through approved interfaces that respect lineage, policies, and access rules. ML models remain trustworthy because the features they use are managed through shared contracts and validated at every step. The result is an AI ecosystem that is both agile and accountable. Every model, prompt, or agent interaction can be traced through lineage graphs and governed semantics, preventing errors and ensuring compliance, while building trust in AI outputs that reflect authoritative enterprise data.

In the Unified Information Fabric, AI is no longer an external consumer of data. It becomes an integrated operational actor, reading, reasoning, and writing through the same controlled pathways used by humans and applications. This alignment transforms AI from an experimental capability into a dependable operational partner that operates with the same rules, context, and truth as the rest of the enterprise.

Potential Risks & How to Mitigate Them

Every transformation introduces an element of risk. The shift to a Unified Information Fabric changes technology, ownership, and culture at once. The most common pitfalls are not technical failures but organizational and design imbalances, losing consistency, over-centralizing control, or locking into rigid tools. Understanding these risks early allows leaders to steer confidently while keeping flexibility and trust intact. Below, are the most common risks we have identified:

- **Dual systems of truth:** Mitigate via data contracts, metric store, and automated reconciliation. Without a single semantic definition, integration and analytics can drift apart.
- **Over-centralization:** Keep a small platform team; push ownership to domains with templates and guardrails. By giving domain teams autonomy within clearly defined templates, organizations maintain innovation speed while preserving consistency.
- **Vendor lock-in:** Favor open interfaces (contracts, schemas, metric definitions) and portable table formats. The fabric relies on interoperability, not allegiance to one stack. Open standards keep options open and make cloud, on-premises, and hybrid environments equally viable.
- **Performance vs. cost:** Use tiered storage, compaction, and lifecycle rules; route to hot paths only the data that's needed. Real-time data adds infrastructure demands. Managing data lifecycles intelligently balances responsiveness with efficiency.
- **Cultural change:** Invest in enablement; celebrate early wins; appoint domain data owners. Technology succeeds when people embrace it. Early pilots that demonstrate visible business impact build momentum and make cultural adoption natural.

These risks are not red flags, they are the predictable growing pains of modernization. The Unified Information Fabric mitigates them by design: shared contracts prevent divergence, automation enforces policy, and active metadata keeps the system observable. With disciplined rollout and strong communication, the benefits of real-time, governed intelligence far outweigh the transition challenges.

A Pragmatic Implementation Roadmap

A successful shift to a Unified Information Fabric doesn't happen through a single project. It's a journey of progressive alignment, starting small, proving value, and scaling through repeatable patterns and micro transformation. The following phased approach is designed to deliver measurable business impact early while laying the foundation for enterprise-wide adoption.

a. Phase 0 – Preparation (2–4 weeks)

- Pick one business journey (e.g., order-to-cash) with visible value.
- Establish data contract format and minimal active metadata model.
- Stand up streaming and catalog foundations; integrate identity/SSO.

The objective of this phase is clarity and alignment. By choosing a single, high-value use case and defining the minimal set of metadata and contracts, teams avoid over-engineering while ensuring all stakeholders share the same vocabulary.

b. Phase 1 – Pilot (60–90 days)

- Implement outbox + CDC for 1-2 systems; publish canonical domain events, creating a live stream of Data in Motion.
- Land the same events in the lakehouse/Data platform to create a corresponding source of Data in Rest; build a metric layer for 3–5 KPIs.
- Provide a GraphQL Gateway for read access, with resolver functions connecting to the pilot sub-domains; expose a write-back command path.
- Wire lineage harvesting to populate the topology graph end-to-end.

Success criteria will be a single source of definitions; one KPI seen consistently in an app, a dashboard, and a simple agent - all in real time and accessible through a unified query interface.

The pilot proves the concept. It demonstrates that integration and analytics can share one semantic layer, and that AI agents or dashboards can consume consistent, governed data in real time. The emphasis here is on visible, cross-functional value rather than technical perfection.

c. Phase 2 – Scale (3–9 months)

- Add domains; build contract templates, pipelines, and tests; expand policy-as-code.
- Introduce reverse ETL patterns to operationalize insights.
- Build feature store (online/offline) for ML with shared contracts.
- Expand the business knowledge graph for 360° use cases.

This phase turns repeatability into scale. Templates, automated tests, and policies make every new domain easier to onboard. At this stage, operational analytics and ML begin to converge, sharing the same definitions and governance model.

d. Phase 3 – Enterprise Rollout (12+ months)

- Integrate finance and regulatory reporting; harden SLOs and cost governance.
- Automate semantic model generation for BI from central definitions.
- Enable agentic workloads to traverse the graph and invoke write-backs safely.

In the final phase, the Unified Information Fabric becomes the enterprise standard. Governance and automation mature into full CI/CD integration, and AI workloads begin to operate directly on governed, real-time data. At this point, the model is self-sustaining and continuously improving.

What Good Looks Like: Signals for the Board

Success in building a Unified Information Fabric isn't measured by the number of pipelines or tools deployed. It's measured by business agility, consistency, and confidence in decision-making. The following signals help executive leaders and boards recognize when the transformation is truly taking hold across technology, operations, and culture.

- Time-to-market for a new KPI or API measured in days, not months. Teams can respond quickly to new business questions or regulatory requirements because definitions, pipelines, and APIs are generated from shared contracts instead of built from scratch.
- A KPI (e.g., Net Revenue) yields the same value in the app, the dashboard, and finance reports. This consistency shows that the enterprise has achieved a single semantic model, one version of truth used across all domains.
- Lineage answers "where did this number come from?" in minutes. Governance and observability are fully operationalized. When executives can trace a metric to its source instantly, trust in data-driven decisions grows exponentially.
- Incidents are resolved faster with graph-driven impact analysis. The platform's topology and lineage graphs allow teams to identify dependencies quickly, shortening the time to detect and recover from data incidents.
- Agents can safely read and write with auditable trails. AI systems operate under the same governance as humans. Every decision, retrieval, or update has in-built traceable lineage and policy enforcement.

Together, these signals indicate a living, governed ecosystem where data flows seamlessly between operations, analytics, and AI. Business and technology teams speak the same language, change cycles accelerate, and decision-makers can act with confidence.

Appendix A: Glossary

- **Active Metadata:** Machine-readable information about data (schemas, lineage, policies) used to automate checks and code generation.
- **Application Programming Interface (API):** A managed doorway to read or write data from a system.
- **Change Data Capture (CDC):** Technology that streams database changes as events.
- **Command Query Responsibility Segregation (CQRS):** A pattern separating read and write models to improve scalability and clarity in event-driven architectures.
- **Data Contract:** A formal agreement describing a dataset, event, or API, including its schema, meaning, SLAs, and policies. Ensures producers and consumers share consistent definitions.
- **Data Contract Registry / Schema Registry:** A version-controlled repository that stores and validates the schemas, metrics, and policies for all data products, ensuring stability and traceability.
- **Data in Motion:** Data that is actively moving between systems, typically as real-time events. It is consumed and acted upon as it is generated.
- **Data in Rest:** Data that is stored in a persistent system such as a database, data warehouse, or lakehouse. Typically analyzed after it has been collected and processed
- **Data Platform:** A centralized or federated environment for aggregating and storing data, often for analytics and business intelligence. In a modern fabric, it acts as one of several governed data sources.
- **Data Product:** A reusable, trusted dataset, stream, or feature owned by a business domain and managed with explicit quality and performance objectives.
- **Event:** A record that something happened, used to notify other systems.
- **Event Broker / Event Mesh:** The infrastructure that distributes event messages across the enterprise.
- **Feature Store:** A governed repository that stores and manages the features used by ML models, ensuring consistency between training and production.
- **Graph (Business):** A map of entities and how they relate (customers, orders, products).
- **Graph (Topology/Lineage):** A map of systems, datasets, APIs, and pipelines showing how data flows, transforms, and depends on other elements.
- **GraphQL / GraphQL Gateway:** GraphQL is a query language and API style to fetch exactly the data you need from many sources. A GraphQL Gateway uses this to provide a single access point over multiple underlying sub-domains and data systems.
- **Lakehouse:** A hybrid data architecture that combines the reliability and structure of a warehouse with the scalability and flexibility of a data lake, supporting both batch and streaming use cases.
- **Metric Layer / Semantic Layer:** Central place where business metrics are defined once and reused everywhere.
- **Outbox Pattern:** A design pattern that records business events in a transactional table ("outbox") within the same database change, guaranteeing consistency before streaming events to other systems.
- **Retrieval-Augmented Generation (RAG):** An AI technique that enriches LLM responses by retrieving relevant, governed information from trusted data sources before generating output.
- **Reverse ETL:** Feeding curated analytics data and insights back into operational systems.
- **Saga:** An orchestrated, long-running transaction that coordinates multiple steps across services with compensating actions for rollback.
- **User-Defined Functions**: Reusable logic modules that help ensure consistency across all data processing layers.

Closing Thoughts

Unifying integration and analytics is not a tooling fad, it is the operating system of a data-driven enterprise. By adopting an event-driven, graph-aware, metadata-first architecture with shared contracts and semantics, organizations can unlock trustworthy real-time operations, analytics, and AI, all from the same source of truth.

The path to a unified information fabric does not begin with a massive rebuild but with a focused, high-value pilot that proves the end-to-end model. Choose one business journey and apply the core principles of the fabric: define shared contracts, stream real-time events through an outbox or CDC pattern, land the same data in the lakehouse, and expose it all through a single query interface such as GraphQL. Demonstrate that one KPI or entity definition can appear consistently in an application, a dashboard, and a simple AI agent, all grounded in the same semantic model. This first 90-day cycle creates the evidence your organization needs: that integration and analytics can finally operate from one definition of truth, and that AI can participate safely and effectively in real-time operations. Once this is shown, scaling becomes a process of templating and onboarding domains, expanding shared libraries, and automating governance. The journey continues, but the breakthrough happens the moment the first business process runs on the unified fabric, proving that the future is practical, attainable, and already in motion.

Credits

Special thanks to Marcus Norrgren and Johan Burman at Sogeti Sweden for their insightful perspectives and feedback.

Authors

Joakim Wahlqvist, Sogeti, Global Head of Data & AI

Avinash Jha, Volvo AB, Head of API & Integration

Arun Sahu, Alliant, Head of Data, AI & Applied Intelligence