KAI WAEHNER

2026

KAI WAEHNER LANDSCAPE 2026

DATA INTEGRATION LANDSCAPE

Event Streaming, API, and Batch in the Era of Agentic AI

Published: July 2026

Kai Waehner
ADVISORY FIELD CTO

kai-waehner.de

INSIDE THIS REPORT

Contents

Data integration is not a new problem. Enterprises have been connecting systems for decades. What is new is the pressure. AI models and autonomous agents need data that is current, governed, and continuously flowing. Batch pipelines that run at midnight and API calls that return yesterday's state are no longer good enough for applications that make decisions in real time. An agentic AI system that queries stale data does not just return a slow answer. It acts on the wrong one.

The result is a market under significant pressure and significant consolidation. The past two years have brought a wave of major acquisitions that signal a fundamental shift in how the industry views integration infrastructure. IBM acquired Confluent for $11 billion. Salesforce acquired Informatica for $8 billion, adding it to a portfolio that already includes MuleSoft and Tableau. TIBCO merged into Cloud Software Group alongside Citrix. Qlik absorbed Talend. Fivetran merged with dbt Labs to combine data movement and transformation. These are not routine acquisitions. They signal that the integration layer has become a strategic asset, not a commodity.

This document maps the data integration landscape across three communication paradigms, analyzes the leading vendors and emerging players, and draws out the trends that will shape enterprise integration architecture through 2027.

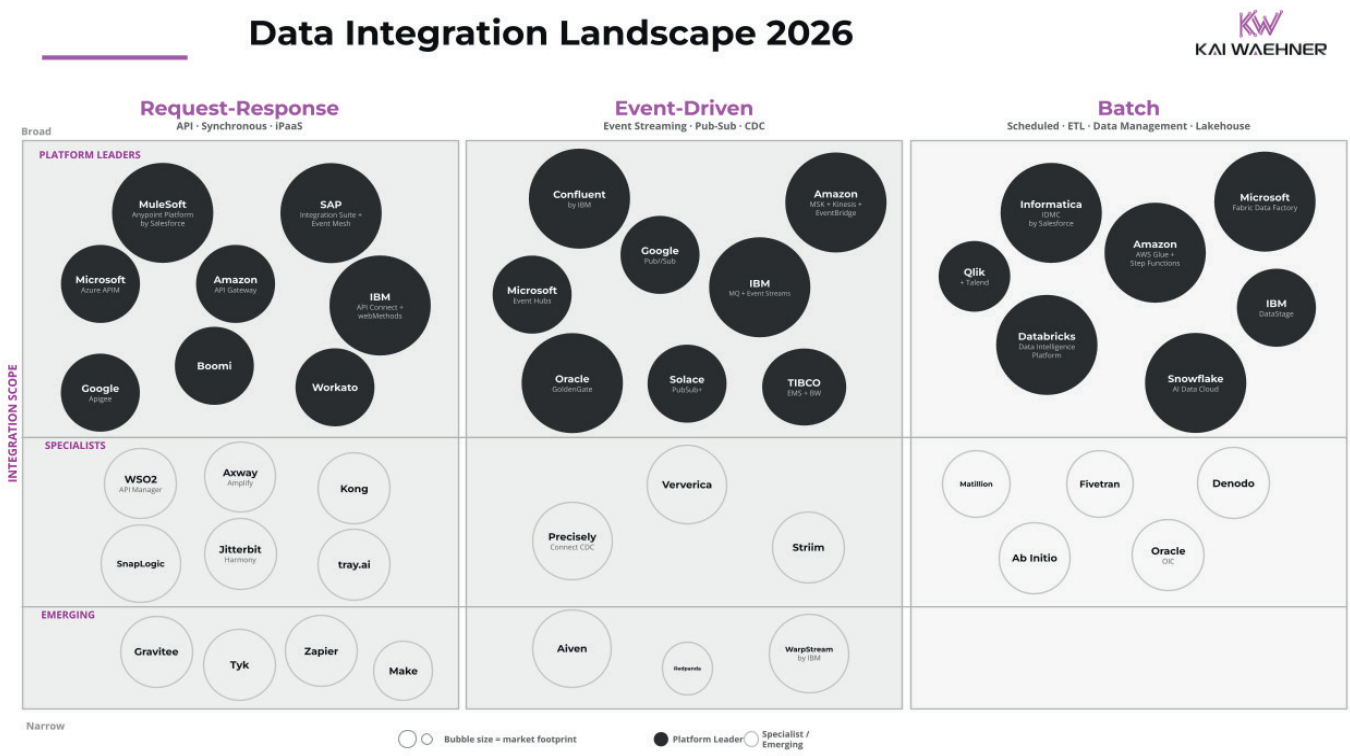

Figure 01. The Data Integration Landscape 2026: vendors plotted across three communication paradigms (Request-Response, Event-Driven, Batch) against integration scope. Bubble size reflects market footprint; filled bubbles are platform leaders, outlined bubbles are specialists and emerging players.

01

CHAPTER 01

How to Read This Landscape

This landscape is defined by two dimensions: broadness of integration scope and supported communication paradigms. By understanding both dimensions companies can save significant time and avoid selecting a platform that serves only one use case instead of many.

Two dimensions define every vendor position in this landscape: **how broad their integration scope is, and which communication paradigms they support.** Understanding both dimensions before evaluating vendors saves significant time and avoids the common mistake of selecting a platform for one use case that cannot serve the next three.

Integration Scope: From Specialized Tools to Universal Platforms

The vertical axis of this landscape measures integration scope. A narrow, specialized tool does one thing well. Precisely Connect excels at mainframe CDC. Kong dominates API gateway functionality. These are deliberate choices, and they serve specific needs well.

A universal platform connects to almost anything: cloud services, on-premises systems, SaaS applications, and legacy middleware. Microsoft, IBM, AWS, MuleSoft and Informatica under Salesforce, SAP, and Google operate at this end of the spectrum, each covering dozens or hundreds of source and target systems.

Scope matters because enterprise architecture is rarely clean. Most large organizations run hundreds of systems across multiple clouds and on-premises environments. A narrow tool solves today's specific problem efficiently. A universal platform absorbs future requirements without requiring a new vendor selection every time the landscape shifts. The right choice depends on what the organization is integrating, how many systems are involved, and whether the integration function is a small team solving specific problems or a platform capability shared across the enterprise.

The Three Communication Paradigms

Understanding the differences is one of the most consequential architectural decisions in data integration. It shapes vendor selection, deployment model, and how well the architecture serves future requirements.

Request-Response

Request-response is the most familiar paradigm. A system asks another system for data and waits for the answer. REST APIs, SOAP web services, GraphQL, and most modern API management tools operate this way. The pattern is synchronous and point-to-point by nature. It works well for transactional interactions: fetch a customer record, submit an order, authenticate a user. Async patterns like webhooks and long polling exist on top of this model, but they simulate asynchronous behavior rather than providing it natively.

The limitation is coupling. When system A needs data from system B, A must know where B is, what its interface looks like, and it must wait. If B is slow or unavailable, A is affected.

Request-response has gone through its own standards evolution worth understanding. In the SOA era, the WS-* stack attempted to standardize everything from security to reliable messaging to atomic transactions over SOAP. The result was a web of interdependent specifications so complex it became known as WS-* hell: brittle, expensive to implement, and painful to interoperate across vendors. HTTP and REST cut through that complexity by standardizing on a small number of well-understood primitives. The lesson is instructive for anyone evaluating event-driven architecture standards today.

Batch

Batch integration moves data in bulk at scheduled intervals. A nightly ETL job extracts records changed since yesterday, transforms them into the target format, and loads them into the destination system. Batch is well-understood and reliable for large-volume historical data loads, analytics pipelines, and reporting workloads that do not require immediacy. **The limitation is latency:** data is always hours or at least minutes old by the time it arrives, which is a fundamental problem for AI models making decisions about what is happening right now.

Batch has no universal transport standard. **SQL is a near-universal query language, and formats like Parquet and Apache Iceberg are emerging as open standards for storage.** But there is no single pipeline standard, which is part of why the batch ETL market has so many tools and why platform lock-in tends to be at the tooling level rather than at the protocol level.

The emerging category in the batch column of the data integration landscape is intentionally empty. New entrants are building for streaming and incremental CDC, not for scheduled batch. The direction of innovation in data movement has shifted decisively toward the event streaming layer, driven by business demands for near-real-time data and the requirements of agentic AI systems that cannot operate on overnight pipeline outputs.

Event Streaming

Event streaming moves data continuously as events occur. When a transaction completes, an order is placed, a sensor reading arrives, or a record changes, that event is published to a stream. Any system that needs it can consume the stream independently, at its own pace, without asking the source system for anything. **The paradigm is asynchronous, decoupled, and persistent:** the event log retains history so late-arriving consumers can catch up.

Within event streaming, three communication patterns coexist. **Fire and forget**: a producer publishes an event without caring whether or how it is consumed. **Publish-subscribe**: one event is consumed by multiple independent consumers, each maintaining their own position in the stream. **Request-reply**: an asynchronous variant where a producer publishes a request event and a consumer publishes a reply event correlated by an identifier. **Request-reply** over an event streaming platform achieves the logical outcome of request-response without the coupling. The producer does not block. The consumer does not need to be available at the moment of the request. The exchange is fully auditable through the event log.

This is why event streaming is the architectural foundation for modern enterprise data integration and for agentic AI. An AI agent needs to act on what is happening right now. It also needs to be decoupled from the operational systems it draws from, so that a slow CRM or ERP response does not block the entire workflow. To be clear, not every organization needs an event streaming backbone today. A mid-size company with a primarily SaaS landscape, no real-time operational use cases, and analytics refreshed daily is often well served by an iPaaS, CDC into a warehouse, and a few managed APIs. Event streaming earns its place when multiple consumers need the same data, when latency affects business outcomes, or when agentic AI moves from pilot to production. The honest guidance is to adopt it when the use cases demand it, not because the market direction says so.

The Standards Fragmentation Problem in Event-Driven Architecture

Event-driven architecture has a fragmentation problem that request-response, after the REST simplification, largely does not. The space includes AMQP for reliable message queuing, MQTT for lightweight IoT telemetry, WebSocket for bidirectional browser connections, CloudEvents as a vendor-neutral event envelope specification, JMS for Java-based messaging, Server-Sent Events for server-to-browser push, and the Kafka protocol itself. Each serves a legitimate purpose in the right context.

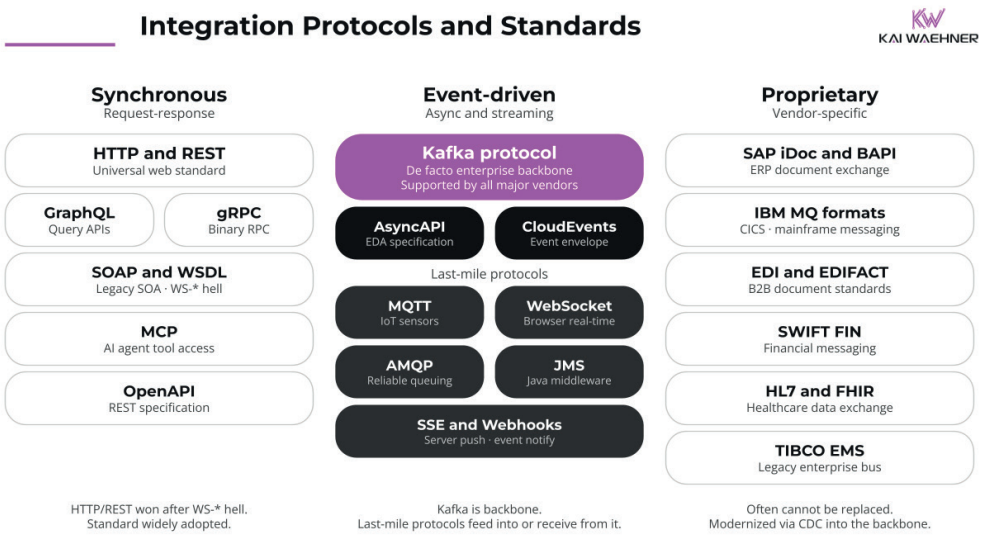

Figure 02. No single standard won for event-driven. Kafka became the de facto backbone through adoption, not standardization.

The important distinction is between the central nervous system and the last mile. For the central nervous system of modern enterprise data architecture, Apache Kafka has become the de facto standard through adoption rather than formal standardization. It became the gravitational center the way Amazon S3 became the de facto standard for object storage: enough producers, consumers, connectors, and cloud services aligned around it that the ecosystem became self-reinforcing.

What matters today is the Kafka protocol, not any single implementation. Azure Event Hubs, Amazon MSK, Confluent, WarpStream, and Aiven all support it natively. Some run Apache Kafka directly. Others re-implement the protocol from scratch. Either way, choosing Kafka-compatible infrastructure does not lock an organization to a single vendor or runtime. Analytical platforms are now adopting the same protocol for ingestion. Snowflake Datastream and Databricks Zerobus Ingest do not make those platforms streaming infrastructure, but they show how widely the Kafka protocol has been adopted as the standard interface for getting data in.

For last-mile delivery, other protocols are the right choice and are complementary. MQTT is correct for IoT sensors and constrained devices where Kafka's overhead is inappropriate. WebSocket is correct for real-time browser updates. AsyncAPI is the right specification format for documenting event-driven interfaces. These are not competitors to Kafka at the architectural center. They are the appropriate interface at the edge, feeding into or receiving from the central streaming backbone.

02

CHAPTER 02

The Architecture Behind the Landscape

Enterprise software vendors already made a shift towards the architecture choices in this landscape: they are moving from batch pipelines and point-to-point APIs toward event-driven backbones that move data continuously and decouple producers from consumers.

The architecture choices in this landscape did not emerge in isolation. They reflect a shift already underway at the largest enterprise software companies in the world: **moving from batch pipelines and point-to-point APIs toward event-driven backbones** that move data continuously and decouple producers from consumers.

How Enterprise Software Vendors Already Made This Shift

The move from request-response to event-driven is not theoretical. It has already happened at some of the largest software companies in the world, and their experience is instructive for any enterprise architect evaluating integration strategy.

Salesforce built its original public integration architecture on WSDL and SOAP web services. It moved to REST APIs, then introduced Platform Events as a time-ordered immutable event stream that enables event-driven communication both within the Salesforce platform and with external systems. Internally, Salesforce runs Apache Kafka across its global data centers as a unified transport for system metrics, logs, network flow data, and application events. The two layers are separate: the public API surface that external systems integrate with, and the internal Kafka backbone that moves data across Salesforce's own infrastructure.

SAP followed a similar path, moving from iDoc and BAPI-based integration toward SAP Event Mesh within the Business Technology Platform (BTP), enabling event-driven communication between SAP and non-SAP systems.

IBM covered all three paradigms through a combination of legacy products and acquisitions. IBM MQ for messaging, IBM DataStage for batch ETL, and IBM Event Streams for managed Kafka were already in the portfolio before the Confluent acquisition added the most complete event streaming platform in the market.

Event Streaming as the Backbone, Not a Replacement

Most enterprises are not choosing between paradigms. **They use all three communication paradigms simultaneously.** What matters is the architecture: event streaming at the center, with request-response and batch as consumer interfaces on top.

A business unit running iPaaS workflows, an analytics team consuming batch exports, and an agentic AI workflow querying current state through an API can all be served from the same event streaming backbone without coupling to each other or to the source systems. The paradigm choice is not either/or. The architectural choice is which paradigm sits at the center and which sit at the edges.
Connecting those edges to the center is an engineering task, not an abstraction. A consumer that speaks REST does not run a Kafka client. A serverless function or a partner endpoint expects to be called, not to poll a topic. This is where event-native API gateways fit. The gateway consumes from a stream and exposes it as a managed interface: a REST endpoint, a Server-Sent Events feed, or a webhook that pushes each new event to a registered callback URL. The streaming backbone stays the source of truth. The gateway handles protocol mediation, authentication, rate limiting, and payload shaping at the boundary, with per-consumer policies instead of custom glue code in every downstream service.

Push delivery matters most for systems that cannot or should not maintain a long-lived consumer. A function such as AWS Lambda, an event bus such as Amazon EventBridge, or a SaaS webhook receives events as they arrive, already filtered and shaped for that destination. One stream can feed several targets, each with its own contract: a CloudEvents envelope for one, a flattened JSON or XML payload for another, a different topic or filter for a third. Gateways from vendors such as Gravitee and Kong cover this event-native role today. The pattern connects the streaming backbone not only to request-response consumers but also to other event-driven systems, including the application-level event buses that route work inside a single cloud.

No Large Organization Runs One Single Platform

One of the most important things this landscape does not show is what the inside of a real enterprise looks like. No large organization runs a single integration platform. A typical Global 2000 enterprise runs TIBCO for EDI and financial messaging, MuleSoft for API-led connectivity, Kafka for event streaming, Informatica for data quality and batch pipelines, and another legacy ESB that nobody has the budget to replace yet. That is not a failure of architecture. It is the reality of decades of technology decisions layered on top of each other.

The right approach is not to replace everything with a single platform. It is to connect the existing platforms through a well-designed integration architecture, with event-driven principles at the center to normalize data flows across paradigms. TIBCO handles EDI formats that Kafka cannot parse natively. Kafka carries the normalized events onward to modern consumers. MuleSoft manages the API surface for synchronous queries. Informatica governs the data quality layer. Each platform does what it does best. **The integration backbone connects them without requiring any of them to be replaced.**

This is why the landscape is organized by paradigm rather than by vendor. The question is not which single platform to pick. It is which platforms cover which paradigms, and how to design the architecture so they work together without creating new coupling at the integration layer.

Three Generations of Integration Platforms: Cloud-Native vs. CloudWashing

Before examining individual vendors, one architectural distinction is worth understanding explicitly because it affects every integration platform evaluation: iPaaS, ESB, API Management, data pipelines, and event streaming infrastructure alike.

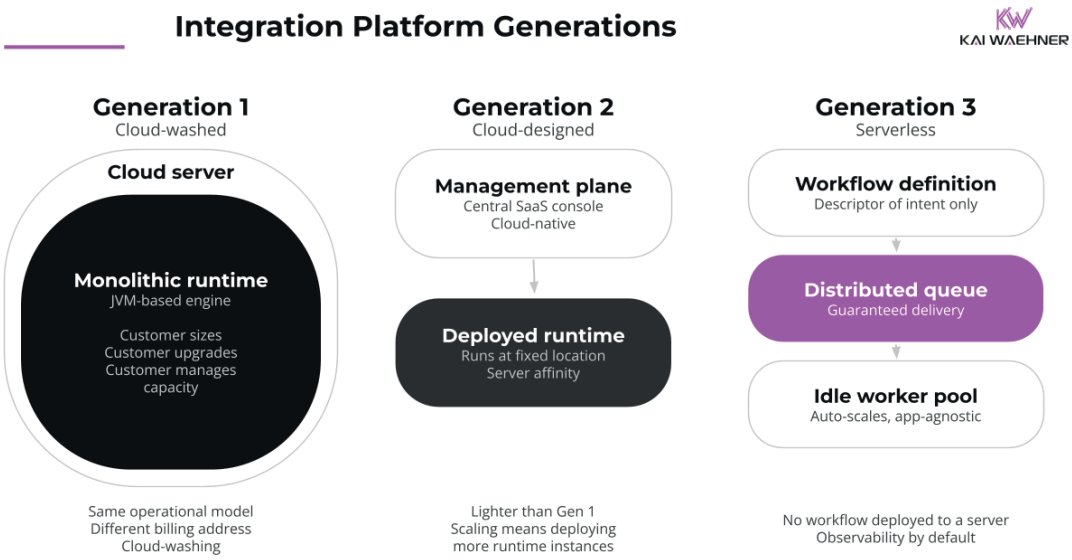

Figure 03. Integration Platform Generations: Cloud-washed, Cloud-designed and Serverless

Integration platforms have evolved through three distinct generations, each representing a fundamentally different approach to cloud deployment and operational model.

The first generation of cloud integration took on-premises middleware and hosted it in the cloud. The runtime, the operational model, the capacity planning: all unchanged. The only thing that moved was the server. TIBCO's hosted BusinessWorks offering is a clear example: the same BusinessWorks that ran in customer data centers, then running on AWS. Customers still sized instances, managed version upgrades, and handled all the operational overhead. This is cloud-washing, not cloud-native. The infrastructure problem moved from the customer's data center to the vendor's infrastructure. The operational model did not change.

The second generation was designed for the cloud but retained deployed runtimes. Boomi's Atom architecture represents this generation: a lightweight execution engine designed specifically for cloud delivery, centrally managed, but still deployed to a specific location with server affinity. When you deploy a process to a Boomi Atom, that Atom runs it. Capacity belongs to that Atom. Scaling means adding Atoms. This is meaningfully more cloud-native than generation one, but it is not serverless. The operational burden is significantly lighter than MuleSoft's runtime model, but it still exists.

The third generation separates the workflow definition entirely from the execution infrastructure. Workato's architecture represents this: recipes are descriptors of intent, decoupled from execution. Events are captured by a dedicated auto-scaling container layer, persisted into a distributed queue, and executed by an independent pool of idle workers. No process is deployed to a server. No capacity is pre-allocated per integration. Guaranteed delivery, error handling, retry logic, and observability are platform capabilities by default, not developer responsibilities. That is a fundamentally different operational model, and its consequences for time to market, team size, and total cost of ownership are measurable.

This distinction is more useful than reading vendor claims about cloud-native architecture. **The test is simple: when integration volume doubles overnight, does the organization running the platform need to do anything?** If the answer is no, the platform is serverless, elastic, and multi-tenant, with consumption-based pricing that scales automatically with usage.

Integration Platform Strategy vs Point-to-Point: The Governance Question

The answer depends on how the platform is used. Any integration platform works for enterprise platform strategy when the integration landscape is well-defined, the team has clear architectural standards, and governance is treated as a first-class concern from the start.

Where it breaks down is when organizations use integration platforms of any kind to build point-to-point connections at speed without architectural discipline for reuse and decoupling. The platform makes it easy to connect system A to system B. It does not prevent the same spaghetti from emerging that existed before, just built faster and at lower cost per connection. Replacing TIBCO BusinessWorks with Workato without changing the integration architecture produces the same coupling problem on a different platform. The same is true for event streaming: replacing a legacy ESB with Kafka without redesigning producer-consumer relationships produces a distributed version of the same coupling.

The organizations that succeed treat the integration platform as an enterprise service, with governance policies, reusability standards, and an integration center of excellence. The organizations that struggle treat it as a self-service tool where every team builds its own connections independently.

Deployment Model: SaaS, Managed Runtime, On-Premises, and Edge

The deployment model question deserves explicit attention because it affects **security, cost, compliance, and operational overhead** in ways that connector count comparisons do not capture.

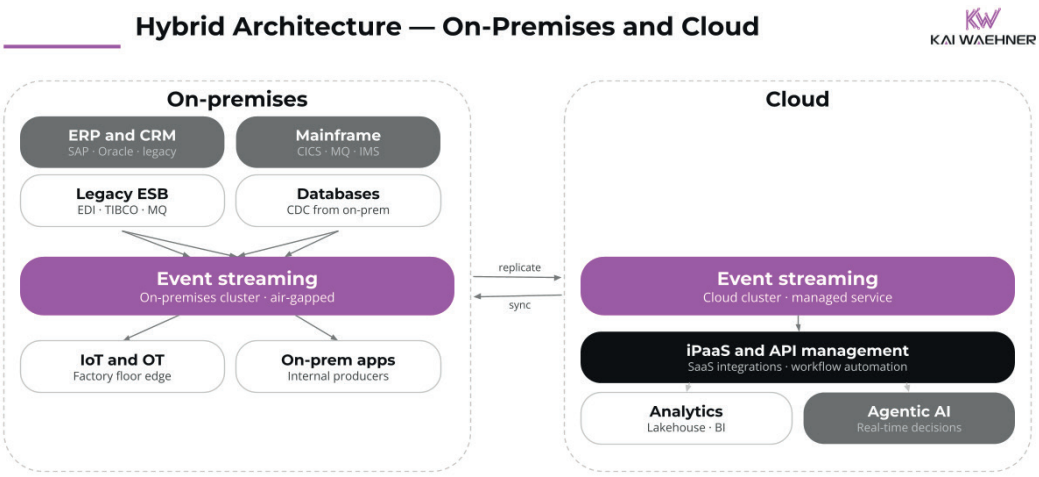

Figure 04. The two event streaming clusters synchronize bidirectionally. Data sovereignty stays on-premises where required.

Fully managed multi-tenant SaaS is the fastest to deploy and the cheapest to operate. Workato, Confluent Cloud, and Snowflake are examples of this model. The trade-off is that data flows through the vendor's infrastructure, which creates data residency considerations for regulated industries.

Cloud-managed runtimes with local execution agents cover the middle ground. The management and monitoring layer stays cloud-hosted while the runtime executes locally or in a private environment. Boomi Atom Cloud and MuleSoft CloudHub follow this model for iPaaS. Amazon MSK and Confluent dedicated clusters follow the equivalent model for event streaming. This addresses most data residency concerns while keeping operational overhead low.

Customer-managed runtimes give maximum control over where execution happens and what data leaves the environment. Boomi Atom on-premises and MuleSoft Runtime Fabric on Kubernetes are iPaaS examples. Self-managed Kafka clusters and on-premises Confluent deployments serve the same role for event streaming. The trade-off is that the customer manages infrastructure, version upgrades, and capacity.

The BYOC (Bring Your Own Cloud) model, pioneered in event streaming through WarpStream, **allows the control plane to be vendor-managed while the data plane runs in the customer's own cloud account.** No data leaves the customer's environment. Integration platform vendors across all paradigms are being pushed in the same direction by organizations with strict data sovereignty requirements, though fully realized BYOC models outside event streaming are still emerging.

For most organizations, a hybrid approach is appropriate. Fully managed SaaS works well for new SaaS-to-SaaS integrations where data residency is not a concern. Customer-managed runtimes suit integrations involving sensitive operational data or regulated systems. Many major integration platforms support some combination of these deployment models, which is why the deployment model question should be asked per-integration-type and per-workload, not per-platform.

Edge deployment runs integration logic at or near the data source, before data reaches a central system. This is relevant for factory floors, retail locations, and remote infrastructure where connectivity is intermittent or latency requirements are sub-second. Lightweight Kafka-compatible brokers, MQTT brokers, and on-premises integration agents can all operate at the edge, feeding data toward a central event streaming cluster or cloud platform when connectivity allows. Edge deployment is not a replacement for a central integration architecture. It is the first hop that bridges operational technology to the integration backbone.

03

CHAPTER 03

The Landscape: Vendor Analysis by Paradigm

The landscape groups vendors by communication paradigm, with leaders at top, specialists in the middle, and emerging players below; multi-column vendors reflect distinct products, not single-product coverage.

The landscape organizes vendors across three columns representing the three communication paradigms. Within each column, platform leaders sit at the top by integration scope, specialists in the middle, and emerging players at the bottom. Vendors that appear in more than one column do so because they have distinct products for each paradigm, not because a single product covers everything.

Overlaps exist at the edges of every platform. MuleSoft includes a built-in messaging layer for asynchronous flows. Apache Flink on Confluent can handle batch processing alongside streaming workloads. Most batch ETL platforms now support some form of incremental or CDC-based ingestion. But almost every platform has a clear design point and primary use case, and the further it reaches outside that core, the more trade-offs appear in latency, scalability, or operational complexity.

In brownfield scenarios where a platform is already in the stack, using its secondary capabilities makes sense. For a greenfield strategic architecture, letting the tool drive the architecture rather than the architecture driving the tool choice leads to systems that are hard to scale and harder to change.

Request-Response: API Management, Synchronous Integration, and iPaaS

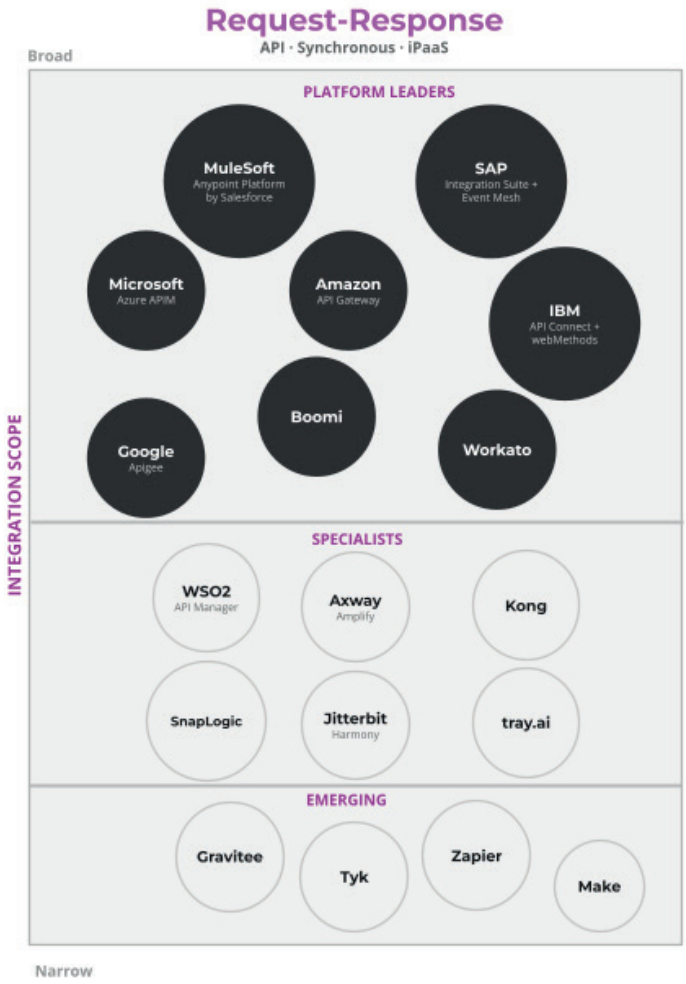

Figure 05. Request-Response paradigm — API management, synchronous integration, and iPaaS. Platform leaders (top tier), specialists (middle), and emerging players (bottom).

The request-response column covers a wide range of tools that share one characteristic: they move data on demand. This includes traditional API management platforms, enterprise iPaaS tools, and API gateways. Every major platform in this column is adding event-driven capabilities at the edges, typically through a built-in message broker, AsyncAPI support, or Kafka connectors. Some go further and expose streams as managed APIs, turning Kafka topics into REST endpoints, Server-Sent Events feeds, or webhooks that push events to downstream services. None of these vendors is repositioning as an event streaming platform, but all of them recognize that their customers need to handle asynchronous flows, and the additions are complementary rather than transformational.

Platform Leaders

The platform leaders in this column range from full API lifecycle management suites to ecosystemspecific gateways, each serving a distinct enterprise context and buyer.

MuleSoft (by Salesforce)

MuleSoft, one of the most established API management and integration platforms in this column, is now owned by Salesforce. **The Anypoint Platform covers API design, management, runtime, and monitoring in a unified environment.** Its connector library spans legacy systems including SAP, Salesforce, and mainframe environments. The API-led connectivity model has been in production in complex enterprise environments for over a decade. **Einstein AI integration** into Anypoint Code Builder lets developers generate integration flows and DataWeave transformations from natural language prompts.

MuleSoft has extended toward event-driven architecture through **Anypoint MQ for asynchronous flows and has adopted AsyncAPI as part of its API management strategy**, allowing organizations to govern event-driven interfaces alongside REST APIs in the same platform.

MuleSoft is a generation-one platform moved to the cloud. The Mule runtime is a JVM-based engine. When integration volume grows, you provision more compute capacity. The platform requires specialist expertise: DataWeave is a proprietary transformation language with its own learning curve, and implementations typically involve dedicated integration teams or certified partners. MuleSoft's pace of iPaaS innovation has slowed as the platform has deepened its alignment with Salesforce's Agentforce and CRM ecosystem. **Organizations that are not deeply Salesforce-committed will find the platform increasingly oriented toward Salesforce-centric use cases.** For organizations with complex legacy environments, a dedicated integration team, and a need for deep API governance, MuleSoft's depth is hard to match. For organizations that need SaaS automation at speed, it is a significant investment for problems that simpler tools solve faster.

SAP Integration Suite

SAP Integration Suite is the **primary integration platform for organizations running SAP at the center of their enterprise architecture**. It covers API management, event-based integration through SAP Event Mesh, pre-built integrations for SAP and third-party systems, and B2B messaging.

SAP Integration Suite Advanced **Event Mesh is powered by Solace technology under an OEM arrangement**, which means organizations using it are already running Solace's event broker infrastructure, whether they know it by that name or not. This matters for organizations evaluating event broker strategies: SAP's event-driven capabilities are built on Solace, which is also available directly as Solace PubSub+ and covered in the event-driven specialists section below.

SAP's broader move toward event-driven architecture, reflected in the strategic investment in Advanced Event Mesh, follows the same direction seen across major enterprise software vendors moving away from point-to-point integration toward continuous, decoupled data flows. The pre-built content library for SAP-to-SAP and SAP-to-third-party scenarios reduces implementation time significantly compared to building from scratch.

Outside the SAP ecosystem, SAP Integration Suite's value proposition weakens considerably. It is not a neutral enterprise integration platform. Organizations that do not run SAP as their core ERP will find limited reasons to choose it over MuleSoft or Boomi. Within SAP landscapes, it is the natural default and often the most efficient path for connecting SAP systems.

Hyperscaler API Management: Azure APIM and AWS API Gateway

Microsoft Azure APIM and AWS API Gateway cover the same architectural role in their respective clouds: **full API lifecycle management from design through deployment, throttling, analytics, and monitoring**. Azure APIM connects natively to Logic Apps, Functions, Event Hubs, and Microsoft Fabric. AWS API Gateway connects natively to Lambda, Step Functions, and the AWS services catalog.

They are the natural default within their cloud but **neither is designed for cross-cloud or on-premises scenarios**. Organizations that need API governance spanning multiple clouds or hybrid environments will find their scope too narrow. Apigee, MuleSoft, or a lighter API gateway such as Kong or WSO2 are the alternatives in those scenarios.

Azure APIM has extended into AI gateway capabilities, adding support for managing and governing calls to Azure OpenAI and other model endpoints alongside traditional APIs. This positions it as a control layer not just for REST APIs but for the AI services that sit behind agentic workflows. AWS is following a similar path through Amazon API Gateway's integration with Bedrock. For organizations building agentic AI on their respective cloud stacks, this makes the hyperscaler API management layer more strategically relevant than it was in a pure REST API world.

IBM API Connect and webMethods

IBM API Connect covers API management and developer portal requirements for organizations with a broader IBM commitment. It manages the API lifecycle layer. IBM webMethods, the hybrid integration platform IBM acquired from Software AG, handles the integration and orchestration layer beneath it, connecting applications, automating workflows, and managing B2B exchanges across on-premises and cloud environments.

IBM webMethods has added AI-assisted integration and MCP support, aligning with the broader market move toward agent-ready integration infrastructure. **Together they form IBM's primary stack for organizations whose primary need is API governance and hybrid application integration rather than event streaming at scale.**

Both products are most valuable for organizations already standardized on IBM infrastructure. Outside IBM-committed environments, they face strong competition from Google Apigee, MuleSoft, Workato, and Boomi.

Google Apigee

Google Apigee covers API design, security, analytics, and developer experience at enterprise scale. Unlike the hyperscaler API management products from AWS and Microsoft, **Apigee supports hybrid deployment across multiple clouds and on-premises environments through Apigee Hybrid**, giving it relevance beyond organizations committed exclusively to GCP. The management plane remains Google-hosted, which is a consideration for organizations with strict control plane data residency requirements.

Apigee has also added AI gateway capabilities for governing calls to Gemini and Vertex AI model endpoints, extending its API management scope into the AI layer.

Apigee is complex to operate and has a steeper learning curve than simpler API gateway products. Pricing reflects its positioning for large-scale API programs. Organizations that need straightforward API gateway functionality without full lifecycle management depth will find lighter alternatives more practical.

Boomi

Boomi's broad customer base, 30,000-plus customers including more than a quarter of the Fortune 500, places it among the platform leaders by market footprint. Its Atom architecture is a second-generation cloud-native design: a **lightweight runtime engine deployable in the cloud, on-premises, or in hybrid configurations**. It is not serverless, but it is significantly lighter to operate than MuleSoft's JVMbased runtime, with strong deployment across manufacturing, healthcare, retail, and financial services.

Boomi has **added pub-sub capabilities and event streaming connectors, extending an API-first platform to handle asynchronous flows without repositioning as an event streaming platform**. Its B2B/EDI capabilities are more mature than most competitors in this column, giving it an advantage in regulated industries and supply chain environments. AgentStudio, now generally available, brings AI agent management into the platform with native MCP support.

Boomi sits between two stools architecturally. It is not as developer-flexible as MuleSoft for complex enterprise API governance, and it is not as operationally lightweight as Workato for high-velocity SaaS automation. The Atom's server-affinity model means platform upgrades require validation of existing flows, adding operational overhead that serverless platforms avoid by design. Boomi operates under private equity ownership by Francisco Partners and TPG, which is relevant context for organizations making long-term platform commitments. For organizations that need hybrid connectivity depth without MuleSoft's complexity or Workato's SaaS focus, Boomi is worth evaluating. For organizations choosing a greenfield integration platform today, the architecture and pricing model deserve careful evaluation before signing.

Workato

Workato represents the third generation of iPaaS architecture: **serverless, multi-tenant, with guaranteed delivery and observability built into the platform rather than delegated to developers**. Recipes are decoupled from execution infrastructure. The platform scales automatically. Operational overhead is minimal. With publicly reported 35 percent year-over-year ARR growth and adoption across half of the Fortune 500, it has moved beyond its SaaS-automation origins into enterprise integration territory.

The founding team built BusinessWorks at TIBCO and chose not to recreate it. They carried forward the integration patterns that work: pub-sub, guaranteed delivery, and process orchestration. They discarded the operational model that created the problems. Workato Event Streams is a pub-sub capability built into the platform, allowing workflows to be triggered by events and enabling decoupling between producers and consumers within the Workato ecosystem. It is built on Kafka infrastructure under the hood, providing proven delivery guarantees, though the Kafka protocol is not directly accessible to consumers outside the Workato platform.

Workato Enterprise MCP extends the same serverless execution model to AI agent orchestration, allowing organizations to govern and deploy AI agents with the same governance and observability built into the integration layer.

Workato's SaaS connectivity is broad but its legacy system depth is shallower than MuleSoft or Boomi. B2B/EDI capabilities are not as mature. For organizations with deep SAP, mainframe, or EDI requirements, MuleSoft or Boomi have certified connectors and production history that Workato does not yet fully match. The low-code model accelerates initial development but can encourage the same point-to-point patterns in recipe form that it was designed to replace. Governance and reusability discipline remain organizational responsibilities, not platform guarantees.

Specialists and Emerging: API Gateways, Automation, and Mid-Market iPaaS

The specialists and emerging vendors in this column serve specific integration needs that sit below the scope of a full enterprise platform, from lightweight API proxying to mid-market workflow automation.

API Gateways and B2B Specialists

Kong, WSO2, Gravitee, and Tyk are API gateways rather than full enterprise integration platforms. They serve organizations that want **lightweight, open-source-based, or cloud-agnostic API management without the overhead of a full iPaaS**.

Kong has grown significantly beyond its origins as a simple API proxy. With over 700 enterprise customers and ARR exceeding $100 million, it is a substantial commercial platform. Its launch of Kong Agent Gateway extended its scope to cover LLM traffic, MCP, and agent-to-agent communication, making it relevant not just for REST API management but for governing AI agent workflows.

WSO2 offers broader scope than a pure gateway, combining API management, identity and access management, and integration under a fully open-source stack, with strong adoption in financial services, telecommunications, and government particularly across Asia Pacific and EMEA. Gravitee positions itself as event-native, exposing Kafka topics as governed APIs and pushing stream data to webhooks, serverless functions, and event buses through per-consumer plans and policies. This places it at the boundary between the request-response and event-driven columns rather than purely in API management. Tyk targets developer-first teams that prioritize open-source flexibility over commercial support. Axway has broader scope than the pure gateway vendors, adding B2B messaging and managed file transfer, giving it relevance in supply chain and regulated B2B scenarios.

Mid-Market iPaaS and Automation

SnapLogic and Jitterbit occupy the mid-market iPaaS tier between API gateways and full enterprise platforms like MuleSoft or Boomi. SnapLogic has invested heavily in agentic AI, launching SnapGPT for AI-assisted integration and AgentCreator for building and deploying AI agents, positioning itself as an agentic integration platform rather than a conventional iPaaS. It covers both application and data integration scenarios. Jitterbit's Harmony platform covers iPaaS, API management, EDI, and low-code application development under a single platform. Its EDI capabilities give it a meaningful advantage in supply chain and B2B scenarios that SnapLogic does not match at the same depth. Both have added agentic AI capabilities and MCP support, targeting IT teams with primarily cloud and SaaS-oriented landscapes that do not need the full complexity of a tier-one iPaaS.

tray.ai sits between consumer automation tools and full enterprise iPaaS. It targets **mid-market organizations needing SaaS workflow automation with enterprise governance.** Its Merlin Agent Builder and Agent Gateway for MCP governance reflect a deliberate pivot toward AI agent orchestration on top of its iPaaS foundation, making it a relevant option for organizations building MCP-governed agent workflows.

Zapier and Make handle high-volume, low-complexity SaaS workflow automation for teams that do not need enterprise governance or legacy connectivity.

Organizations that outgrow API proxying or consumer automation will need to evaluate a full iPaaS or event streaming platform to cover the additional scope.

Event-Driven: Event Streaming, Pub-Sub, and CDC

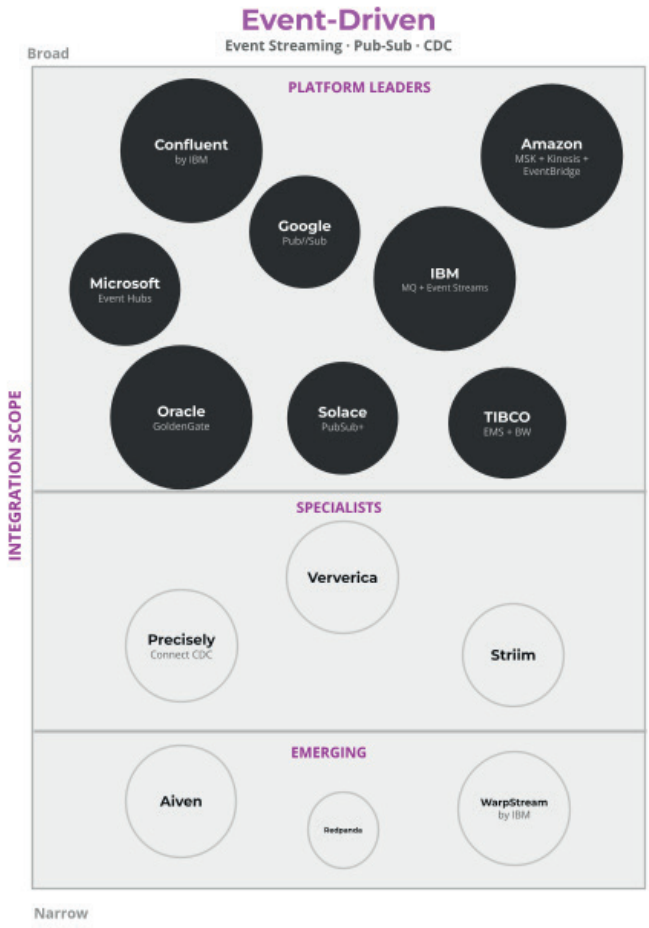

Figure 06. Event-Driven paradigm — event streaming, pub-sub, and change data capture (CDC). Platform leaders (top tier), specialists (middle), and emerging players (bottom).

This column is the architectural backbone of the modern enterprise data landscape. It is also where the most significant commercial consolidation has occurred. The vendors here provide the infrastructure for continuous data movement: event brokers, managed streaming services, change data capture engines, and stream processing platforms.

Platform Leaders

Confluent (an IBM Company)

Confluent, an IBM company, **is the largest standalone event streaming platform in the market.** It covers the full data streaming lifecycle: connecting to any source or target system, streaming data continuously at scale, processing it in motion with real-time and historical context, and governing it end to end through an integrated governance suite, with Tableflow and Apache Iceberg support for long-term storage.

Its publicly reported 6,500-plus customers and 40 percent Fortune 500 penetration reflect a decade of building the event streaming category. **The IBM acquisition gives it broader enterprise sales reach and tighter integration with IBM's hybrid cloud, mainframe, and watsonx AI stack.** Confluent continues to operate as a distinct brand and product within IBM.

The IBM acquisition introduces questions about long-term product independence. IBM kept Red Hat as an independent brand and operating unit after its 2019 acquisition, which is the closest precedent. Enterprise architects making five-year platform decisions should track the combined roadmap. Confluent's pricing has historically been a friction point for organizations that want Kafka without the commercial overhead.

For organizations that need a complete managed event streaming platform covering ingestion, processing, governance, and storage in a single product, **Confluent is the benchmark against which others are measured.**

Amazon MSK and Kinesis

Amazon MSK provides managed Apache Kafka, inheriting the full Kafka ecosystem while significantly reducing operational overhead. Amazon Kinesis is AWS's proprietary streaming service, optimized for AWS-native workloads. Together they handle a substantial share of cloud-native event streaming workloads given AWS's overall market position.

Amazon MSK provides Kafka infrastructure without the additional governance, processing, and storage capabilities of a full commercial streaming platform. Organizations that need fully managed Schema Registry, stream governance, or Apache Iceberg integration will find MSK requires more assembly from additional AWS components.

Kinesis is a streaming service designed for high-throughput, low-latency event ingestion within the AWS ecosystem. It integrates natively with AWS Lambda, S3, and analytics services, making it a natural fit for AWS-native pipelines. Its shard-based architecture requires capacity planning and shard management as workloads scale, and its protocol is specific to the AWS ecosystem.

For organizations deeply committed to AWS and willing to assemble Schema Registry, monitoring, and governance tooling from separate AWS components, MSK is a practical and cost-effective choice. For organizations that want a completely managed streaming platform with minimal assembly, Confluent or a comparable purpose-built platform is the more practical choice.

Amazon EventBridge is a serverless event bus for routing events between AWS services and SaaS applications based on rules and filters. It handles application-level event routing rather than high-throughput data streaming, making it complementary to MSK and Kinesis rather than a replacement for either.

Microsoft Azure Event Hubs

Azure Event Hubs is **Kafka-API-compatible**, meaning existing Kafka producers and consumers work without code changes. For organizations committed to Azure, it provides event streaming infrastructure **integrated with Azure Stream Analytics, Azure Functions, and the Microsoft Fabric data platform**.

Event Hubs covers ingestion and basic stream routing well. For complex stateful stream processing, Azure Stream Analytics is needed alongside it, which adds components and operational complexity. Within Microsoft Fabric, Eventstream provides a no-code experience for capturing, routing, and transforming real-time data, extending Event Hubs capabilities for teams that prefer a visual interface over direct Kafka API access.

Organizations that want a single platform covering the full streaming lifecycle will find Event Hubs requires more assembly than purpose-built streaming platforms.

Google Cloud Pub/Sub

Google Cloud **Pub/Sub** handles the **messaging and buffering layer** for GCP-committed organizations. **Dataflow**, built on the Apache Beam model, handles stateful stream processing on top of it, supporting exactly-once processing semantics and scaling to millions of events per second. Together they form a mature, well-integrated **streaming pipeline** for organizations standardized on GCP.

Google Cloud also offers **Google Cloud Managed Service for Apache Kafka**, providing Kafka-compatible infrastructure for organizations that want Kafka protocol compatibility within GCP. The service reached general availability in 2024 and is actively expanding its capabilities, though its ecosystem depth and enterprise adoption are still maturing relative to Amazon MSK and Confluent.

IBM MQ and IBM Event Streams

IBM MQ is a **message queuing system with decades of production deployment in financial services, healthcare, and government**. Organizations in those sectors rely on it for transaction-critical messaging where delivery guarantees and audit trails are non-negotiable. IBM MQ's strength is its reliability in regulated industries where it has been in production for decades. It is not a modern streaming platform and should not anchor new event-driven architectures. Organizations that rely on IBM MQ for financial transaction messaging should continue using it for that purpose and connect it to Kafka for modern event streaming workloads. **The combination of IBM MQ and Kafka is proven and does not require replacing IBM MQ.**

IBM Event Streams is IBM's managed Kafka service, built for organizations that want Kafka infrastructure within IBM's supported ecosystem. **IBM Event Automation extends this with a low-code stream processing layer built on Apache Flink**, providing a drag-and-drop canvas interface that allows business and IT users to build, test, and deploy event processing flows without deep Flink expertise. Together, IBM MQ, Event Streams, and Event Automation cover the messaging, streaming, and processing layers for organizations standardized on IBM infrastructure. With the Confluent acquisition complete, IBM Confluent is now positioned as the strategic streaming platform within IBM, available as IBM Confluent Cloud with a 99.99 percent uptime SLA and managed Apache Flink. Event Streams and Event Automation remain available within Cloud Pak for Integration, but enterprise architects should expect IBM to steer new streaming investment toward Confluent rather than the legacy Kafka offerings. Anyone evaluating this stack today should treat Confluent as the go-forward platform and the existing Event Streams deployments as a migration consideration over time.

WarpStream, acquired by Confluent before the IBM deal, **addresses cost concerns through a BYOC diskless architecture** and is covered in the Emerging section below.

Solace PubSub+

Solace PubSub+ is an **event broker platform for scenarios requiring guaranteed message delivery, fine-grained topic routing, and multi-protocol support across on-premises, cloud, and IoT edge environments**. It has deep deployment in financial services and government for high-reliability event-driven architectures where request-reply messaging patterns and strict delivery guarantees are architectural requirements that Kafka's throughput-optimized model does not natively address. It is also the technology powering SAP Integration Suite Advanced Event Mesh under an OEM arrangement, which means organizations using it are already running Solace infrastructure at the event broker layer.

Solace is well suited to these specific scenarios: financial trading systems, air-gapped government environments, and situations requiring **AMQP, MQTT, and JMS support from a single broker**. For most enterprise event streaming use cases, Kafka's ecosystem depth makes it the more practical architectural choice. Solace and Kafka are more complementary than competitive: Solace handling protocol translation and fine-grained routing at the integration boundary, Kafka at the center for the streaming backbone.

TIBCO

TIBCO, now within Cloud Software Group, **built the category of real-time middleware in enterprise computing**. Its EMS message broker and BusinessWorks integration server together cover event-driven integration and process orchestration. Its installed base in financial services, manufacturing, and logistics remains substantial.

The Cloud Software Group acquisition has raised legitimate questions about long-term product investment. **TIBCO is managing a portfolio of legacy middleware products under private equity ownership**, which typically prioritizes margin over product innovation. Organizations with existing TIBCO deployments should be planning migration paths rather than expanding their footprint. Organizations evaluating new event streaming infrastructure should look elsewhere. The use case where TIBCO remains valuable today is the multi-platform scenario: TIBCO handling complex protocols like EDI formats and legacy message translations, Kafka carrying normalized events onward. That combination works well and does not require replacing TIBCO on a fixed timeline.

Oracle GoldenGate

Oracle GoldenGate has decades of production history as a **real-time CDC and data replication platform**, with deep deployment in financial services and telecommunications. It captures changes from source databases with minimal latency and delivers them to targets including Kafka topics, cloud data warehouses, and other databases. The 26ai release adds an **AI microservice that performs inline analysis and transformation on streaming data during replication**, extending GoldenGate's role beyond pure CDC into AI-enriched data movement.

GoldenGate works well for Oracle-centric environments and has **expanded its non-Oracle source coverage** to include PostgreSQL, MySQL, SQL Server, and DB2. It is now available as a managed service within Oracle Cloud, Azure, and Google Cloud. Outside Oracle infrastructure, it faces strong competition from Qlik Replicate and Striim at lower price points. Implementation complexity is high and specialist expertise commands a premium.

Specialists: Ververica, Precisely, Striim

The specialists in this column each **address a specific layer of the event streaming architecture**: stream processing at the compute layer, or CDC and real-time data integration at the source.

Ververica, founded by the original creators of Apache Flink, is the leading independent commercial platform for **Flink-based stream processing.** Acquired by Alibaba Group in 2019, it operates as an independent product unit with deep roots in the Apache Flink open-source community. It offers deployments across on-premises, public cloud, private cloud, and BYOC environments.

Ververica has expanded significantly beyond a pure Flink runtime. **The company introduced Apache Fluss, a streaming storage layer that treats streams as queryable tables and integrates natively with Apache Paimon and Apache Iceberg.** This positions Ververica as a unified streaming data platform covering processing, real-time storage, and lakehouse integration in a single architecture. For analytics-focused architectures, Fluss can replace Kafka as the messaging layer while feeding the same open lakehouse formats, which is a meaningful architectural difference from platforms that use Kafka as the backbone throughout. Ververica also added Flink Agents, extending the platform toward AI agent orchestration on streaming data.

For organizations that need managed Kafka alongside Flink processing, Confluent or Aiven are the more complete options. For organizations that have already standardized on Kafka and need a Flink processing platform on top, or for those building analytics-first streaming architectures around open lakehouse formats, Ververica is the strongest independent option. Organizations with data sovereignty requirements or restrictions on Chinese-owned infrastructure should factor the Alibaba ownership into their evaluation.

Precisely Connect addresses the narrow but critical problem of mainframe and IBM i CDC. For organizations with mainframe-centric architectures, it is one of the few production-proven paths to streaming mainframe data into modern systems without replacing the source. It belongs in the integration architecture alongside Kafka, not instead of it.

Striim provides real-time data integration with a focus on database replication and CDC, covering a wider range of source databases than GoldenGate at lower price points. It targets organizations running mixed database environments across Oracle, SQL Server, PostgreSQL, and cloud databases that need real-time streaming pipelines without GoldenGate's licensing overhead.

Emerging: Aiven, Redpanda, WarpStream

The emerging vendors in this column offer alternative infrastructure for organizations evaluating **specific deployment models, cost profiles**, or Kafka-compatible capabilities outside the major cloud platforms.

Aiven is a managed platform offering hosted Kafka, Flink, and related services across AWS, GCP, Azure, and other cloud providers. Aiven maintains a fully open-source stack. This **combination of multi-cloud portability and open-source foundations** makes it a natural choice for organizations that want Kafka capabilities without cloud or vendor dependency.

Redpanda started as a Kafka-compatible streaming platform built in C++ without the JVM overhead, delivering lower latency and reduced operational overhead compared to standard JVM-based Kafka deployments. It has since **rebranded almost entirely around the Agentic Data Plane**, a governed access layer for AI agent connectivity that combines streaming, SQL querying, Apache Iceberg support, and MCP integration. The rebrand reflects the **difficulty of competing with Confluent, Amazon MSK, and Aiven on streaming infrastructure alone**. The agentic data plane vision is coherent but the platform is still maturing, with key capabilities in beta or early availability as of early 2026.

WarpStream, now part of the Confluent and IBM portfolio following its acquisition, pioneered a diskless Kafka architecture that eliminates local broker storage by writing directly to object storage such as S3. This decouples compute from storage and significantly reduces the cost of high-volume Kafka workloads. It operates **exclusively as a BYOC deployment**, meaning the data plane runs in the customer's own cloud account while the control plane is vendor-managed. For organizations with high data volumes, strict data residency requirements, or cost constraints on traditional Kafka infrastructure, WarpStream is the **most cost-efficient Kafka-compatible option in the market.**

Batch: ETL, Data Management, and Lakehouse

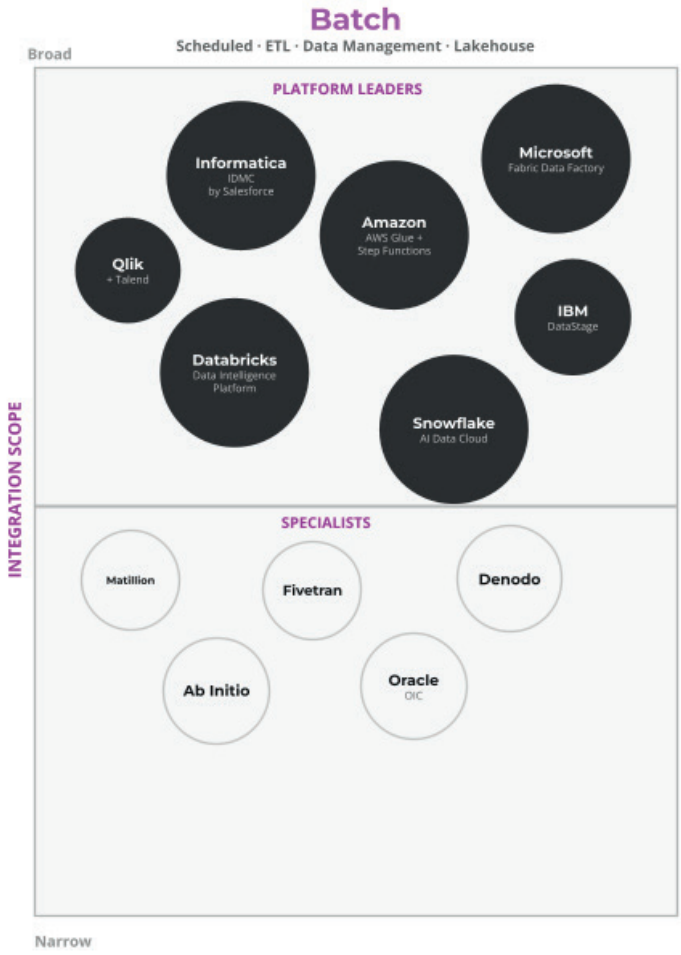

Figure 07. Batch paradigm — scheduled ETL, data management, and lakehouse. Platform leaders (top tier), specialists (middle), and emerging players (bottom).

Batch integration is not dying. It is evolving. Large-scale historical data loads, analytical pipeline refresh cycles, regulatory reporting, and model training all continue to require bulk data movement. What is changing is the expectation of freshness, and several platforms in this column are responding by adding streaming ingestion capabilities.

The emerging row in the batch column is intentionally empty. New entrants in data movement are building for streaming and incremental CDC rather than scheduled batch pipelines. The innovation has moved to the event streaming layer, driven by business demands for near-real-time data and agentic AI systems that cannot operate on overnight outputs. Batch will remain essential for historical loads and analytical workloads, but it is no longer where new architectural thinking is being applied.

Platform Leaders

The platform leaders in this column range from full-suite data management platforms to cloud-native lakehouse engines, each approaching batch integration from a different design point and target buyer.

Informatica IDMC (by Salesforce)

Informatica IDMC, a Salesforce company, is the largest and most comprehensive data management platform in this column. It covers **ETL, ELT, data quality, data catalog, data governance, master data management, and API integration under a single cloud-native platform**. CLAIRE, its AI engine, handles automated data discovery, quality scoring, and mapping recommendations. The combination of Informatica's data management depth with Salesforce's MuleSoft integration capabilities and Agentforce AI creates what Salesforce positions as a unified data architecture for agentic AI.

Implementation complexity is high. The Salesforce acquisition raises questions about whether Informatica will remain a neutral platform for non-Salesforce customers or gradually align more closely with the Salesforce ecosystem. At the scope of ETL, data quality, governance, cataloging, and MDM in a single platform, IDMC has no direct equivalent in the market. **For organizations whose primary need is ETL pipeline capability, the scope of IDMC exceeds the requirement.** Simpler tools will serve that need better.

AWS Glue and Step Functions

AWS Glue is a **serverless data integration service covering ETL, ELT, data quality, and lakehouse pipeline development** within the AWS ecosystem. It generates PySpark code automatically from schema discovery, making it **accessible to teams without deep Apache Spark expertise**, and has expanded its Apache Iceberg support significantly with fine-grained access control, row lineage tracking, and data quality monitoring with ML-based anomaly detection.

AWS Glue works well for AWS-native data lake and warehouse pipelines. Its serverless model means no infrastructure management. Outside AWS environments, it offers nothing. For complex transformations requiring fine-grained control, the auto-generated PySpark can be harder to debug than handwritten pipelines.

Step Functions orchestrates multi-step data workflows with error handling and retry logic. Beyond data pipelines, it is increasingly used for agentic AI workflow orchestration, coordinating sequences of model calls, tool invocations, and conditional branching within AWS-native agent architectures.

Microsoft Fabric Data Factory

Microsoft Fabric Data Factory is the **strategic data integration platform within Microsoft Fabric, sitting alongside Synapse Analytics, Power BI, and the Fabric lakehouse in a unified analytics platform**. It covers batch ETL, ELT, pipeline orchestration, and data movement across a broad connector library including both Microsoft and non-Microsoft sources.

Organizations running Azure Data Factory should be aware that Microsoft shifted primary development investment to Fabric Data Factory in mid-2024. New capabilities are being built exclusively in Fabric and are not being backported to ADF. A migration assistant launched in public preview in March 2026 to help move existing ADF and Synapse pipelines into Fabric with minimal disruption.

Like AWS Glue, its primary context is the vendor's cloud ecosystem. Organizations running multi-cloud or on-premises-heavy architectures will need supplementary tooling.

IBM DataStage

IBM DataStage is a **long-established ETL platform with particular strength in regulated industries requiring auditable, high-throughput data transformation at scale**. It has been extended with AI-assisted job design, natural language pipeline building, and cloud deployment options including DataStage as a Service on AWS. Its capabilities are now integrated within IBM watsonx.data, covering batch, real-time streaming, replication, and data observability across structured and unstructured data.

DataStage's strength is in environments where it has been running for years. **For greenfield cloud-native data engineering, more modern alternatives exist**. Its developer experience is showing its age despite the low-code and no-code additions. For organizations with existing DataStage investments that are performing well, migration cost rarely justifies switching.

Qlik and Talend

Qlik, following its acquisition of Talend in 2023, covers **real-time data integration through Qlik Replicate, batch ETL and data quality through Talend Data Fabric, analytics through Qlik Sense, and lakehouse data management through Qlik Open Lakehouse**. The Open Lakehouse, built on its Upsolver acquisition, provides high-throughput ingestion and real-time processing into Apache Iceberg tables with automatic optimization, giving the combined platform a clearer architectural position than it had at the time of the Talend merger.

Qlik Replicate handles CDC use cases well and remains one of its strongest capabilities. The April 2026 Connect announcements added agentic data engineering capabilities including declarative pipelines, an AI assistant for Talend Studio, MCP support for agentic process routing, and native Open Lakehouse Streaming. The Talend brand has lost some of its identity within Qlik but the combined platform is finding a coherent direction around governed lakehouse pipelines and agentic data engineering. For organizations that need deep CDC-driven data pipelines feeding a lakehouse, the Qlik and Talend combination is worth evaluating alongside Informatica.

Databricks

Databricks remains a **lakehouse and analytics platform at its core**, but it has **added a meaningful set of data engineering and streaming capabilities** that enterprise architects need to account for.

Lakeflow, the unified data engineering product covers ingestion through Lakeflow Connect, batch and streaming pipeline transformation through Lakeflow Spark Declarative Pipelines, and workflow orchestration through Lakeflow Jobs. Auto Loader handles incremental file ingestion. Lakeflow Spark Declarative Pipelines supports continuous data processing from Kafka, Kinesis, Event Hubs, and other message buses. Real Time Mode for Apache Spark, announced alongside Lakeflow, enables stream processing at significantly lower latencies than standard microbatch, moving Databricks meaningfully beyond its batch-oriented origins. Databricks has also added Kafka-compatible APIs to Zerobus Ingest, allowing existing Kafka producers to stream data directly into Databricks with a configuration change rather than code changes. Zerobus is generally available; the Kafka-compatible ingest layer is in beta.

Databricks is a **practical data engineering platform for organizations standardized on the Databricks lakehouse**. For operational event streaming, sub-second latency, decoupled producer-consumer architectures at scale, and mission-critical SLAs requiring zero data loss and enterprise-grade disaster recovery, purpose-built event streaming platforms hold a meaningful advantage. Databricks is designed for analytical resilience, not operational messaging guarantees. The trajectory is toward more streaming capability, but the design point remains analytical. Zerobus Ingest is Kafka-compatible ingestion into the lakehouse, not a substitute for running Kafka as the operational event-driven backbone of the enterprise.

Snowflake

Snowflake is a **cloud-native data platform built for analytical workloads, offering a fully managed data warehouse, data sharing, data marketplace, and a growing set of AI and ML capabilities** through Cortex. Its separation of storage and compute allows independent scaling and consumption-based pricing. Governance across structured, semi-structured, and unstructured data is handled through a single platform without moving data between systems. It is one of the most widely adopted analytical platforms in the enterprise market.

Snowflake has extended toward data ingestion and CDC through Snowpipe Streaming, which provides a low-latency SDK for writing rows directly into Snowflake tables from Kafka and CDC agents. Dynamic Tables bring declarative incremental transformation on top of streamed data. OpenFlow, now generally available on AWS, Azure, and Google Cloud, is a managed integration service built on Apache NiFi covering Kafka, databases, SaaS sources, and unstructured data. The OpenFlow Connector for Oracle adds near real-time CDC from Oracle, extending Snowflake's ingestion reach into database replication territory.

The most significant addition is Snowflake Datastream, announced at Snowflake Summit and heading into private preview. **It is a fully managed streaming service native to Snowflake that speaks the Kafka wire protocol.** Snowflake is direct about the implementation: **Datastream is Kafka-protocol-compatible but does not run Kafka underneath.** The technology is pure Snowflake. Existing Kafka producers and consumers can stream data straight into Snowflake or open Iceberg tables with a configuration change rather than code changes, inheriting Snowflake governance, security, and lineage. It confirms that Kafka protocol compatibility has become a baseline expectation for any platform that wants to receive streaming data.

Snowflake Datastream is **Kafka-compatible ingestion into the analytical platform, not a substitute for running Kafka as the operational event-driven backbone of the enterprise**. Sub-second latency, decoupled producer-consumer architectures at scale, and mission-critical delivery guarantees remain the territory of purpose-built streaming platforms. For organizations that want Snowflake as the analytical destination and Kafka as the operational streaming backbone, the combination works well. Datastream lowers the friction of that connection. The lakehouse becomes one important consumer of the streaming backbone, not a replacement for it.

Specialists

The specialists in this column each serve a specific integration scenario: API-led integration for Oracle environments, ELT transformation for cloud warehouses, automated pipeline connectors, high-throughput batch processing, or logical data virtualization without physical data movement.

Enterprise Application Integration: Oracle

Oracle Integration Cloud (OIC) covers API-led integration, batch data orchestration, process automation, and B2B integration within Oracle environments. Its pre-built integrations for Oracle ERP, HCM, and SCM reduce implementation effort significantly for organizations already running those applications. Oracle Integration has added AI assistant capabilities, MCP support for exposing integrations as AI agent tools, and native AI agent lifecycle management within projects. Outside Oracle environments, its relevance drops significantly.

ELT, Pipelines, and Data Virtualization: Matillion, Fivetran, Ab Initio, Denodo

Matillion provides ELT transformation optimized for Snowflake, Databricks, BigQuery, and Amazon Redshift. Data engineering teams on those platforms use it to build and manage ELT pipelines through a visual interface without writing transformation code from scratch. Its Maia AI assistant adds natural language pipeline building. It is not an enterprise integration platform for operational or event-driven workloads. Databricks is an investor, which reflects its tight alignment with the lakehouse ecosystem.

Fivetran provides automated pipeline connectors from hundreds of SaaS and database sources into cloud data warehouses. Its zero-maintenance managed pipeline model is its primary strength. Fivetran completed its all-stock merger with dbt Labs, the leading open-source data transformation framework, creating a combined platform approaching $600 million in ARR. The combination covers data movement through Fivetran and in-warehouse transformation through dbt, giving the merged company a stronger end-to-end position in the modern data stack than either had independently.

Ab Initio is a closed, high-performance batch processing platform with deployment history in **financial services** for large-scale transformation workloads requiring extreme throughput guarantees. Its closed nature, high cost, and aging developer experience make it a difficult choice for organizations not already running it. For existing customers with workloads that push it hard, it remains difficult to replace.

Denodo specializes in data virtualization, providing a logical integration layer that queries distributed sources without physically moving data. Data virtualization reduces pipeline complexity but introduces query latency and requires source systems to be available at query time. It is a complement to ETL, not a replacement.

04

CHAPTER 04

The Legacy Integration Challenge

Every enterprise has legacy integration debt. Most have more than they can measure.

Proprietary Enterprise Integration Formats

SAP iDoc, BAPI, ALE, and RFC carry the transactional backbone of most large manufacturing, retail, and logistics enterprises. Oracle e-Business Suite adapters, IBM CICS transaction interfaces, and native MQ message formats carry the equivalent infrastructure in financial services and government. These formats are not going away.

The integration challenge is accessing this data in a modern architecture without replacing the source. The modernization path runs through CDC and adapter connectivity, with the event streaming platform becoming the normalized layer through which modern consumers access what were previously isolated, vendor-specific data silos.

Open Vertical Standards

EDI, EDIFACT, and EDI X12 govern B2B document exchange in global trade and logistics. SWIFT FIN and ISO 20022 govern financial messaging globally, with the ISO 20022 migration representing the most significant financial messaging standard change in a generation. HL7 v2 remains the dominant healthcare data exchange standard, with FHIR accelerating as its modern replacement.

In all cases the modernization approach is the same: CDC or adapter-based extraction, normalization into event streaming, and delivery to modern consumers. The legacy format does not disappear. The integration architecture absorbs it at the edge.

05

CHAPTER 05

Industrial IoT: A Complementary World

Industrial IoT integration occupies a distinct architectural space from enterprise data integration. The two worlds are converging but they are not the same world.

Operational technology has historically been closed, proprietary, and focused on uptime above all else. A production line built to run for 30 years was not designed with API connectivity or cloud integration in mind. That is changing. **Industry 4.0 is pushing OT toward open standards, real-time data flows, and connectivity with enterprise IT systems.** Machines, sensors, PLCs, and SCADA systems are generating more data than ever, and that data carries enormous business value when it reaches analytics platforms, supply chain systems, and AI applications.

The open standards driving this convergence are **OPC-UA** for vendor-neutral machine-to-machine communication, **MQTT** for lightweight telemetry from constrained devices, and Sparkplug B as a standardized payload format over MQTT. These standards are making OT data increasingly accessible without replacing the source systems that generate it.

The enterprise integration platforms in this landscape are not designed for the last mile of OT connectivity. They do not speak PLC protocols natively. They do not run on factory floors with intermittent connectivity. That is not a gap, it is a design boundary. Purpose-built IIoT platforms and MQTT brokers handle the OT edge. Event streaming platforms handle the normalized data once it crosses into the IT layer. Edge and hybrid deployment are not optional in this architecture. Factories stay on-premises. Local edge processing combined with selective replication to a central cluster is the dominant pattern, with the integration platform spanning both environments and providing a consistent integration layer from the shop floor to the enterprise and cloud. **The two architectures are complementary by design: the IIoT layer captures and bridges, the integration layer carries and connects.**

Event streaming is a natural fit for industrial workloads given its real-time, scalable, and decoupled architecture. More detail on how it works with industrial data historians, OT-IT convergence, and edge architectures is available in the article Apache Kafka as Data Historian in IIoT and Industry 4.0. The architectural principles it describes remain valid. IT-OT convergence is a decades-long journey that is still in its early stages.

06

CHAPTER 06

Trends Worth Watching: Six Forces Shaping Data Integration Architecture Over the Next Two Years

There are six identified trends that are already visible in product roadmaps, acquisition activity, and the architectural decisions of companies in 2026.

The following trends are not predictions. They are already visible in product roadmaps, acquisition activity, and the architectural decisions that enterprise teams are making today.

Zero-ETL reduces pipeline overhead within a single vendor's ecosystem but creates point-to-point coupling that scales poorly when multiple consumers need the same data.

MCP is the emerging standard for connecting AI agents to enterprise tools, but it is a request-response protocol that sits on top of the integration layer. The data underneath still needs to be current and governed.

AI-native integration capabilities are now standard across major platforms. Generated mappings and natural language pipeline building accelerate initial development but remain immature for complex transformation logic and governance.

Visual low-code tools and generated code are converging rather than competing. Visual tools suit governed enterprise workflows. Generated code suits developer-owned logic where flexibility matters more than built-in guardrails.

Data contracts and schema governance enforce agreements between producers and consumers about data structure and SLAs. Retrofitting governance after pipelines are in production costs significantly more than building it in from the start.

Hybrid cloud and on-premises deployment remain permanent operating models for most large enterprises. The deployment model question should be asked per workload, not per platform.

07

CHAPTER 07

Three Architecture Decisions That Matter More Than Vendor Selection

The acquisition activity of the past two years carries three clear architectural messages for enterprise technology leaders.

The Three-Paradigm Suite Is the New Competitive Baseline

IBM now covers event streaming through Confluent, messaging through IBM MQ, API management through IBM API Connect, and batch ETL through IBM DataStage. Salesforce covers API and iPaaS integration through MuleSoft, data management and batch ETL through Informatica, and event-driven integration through MuleSoft and Salesforce Platform Events. Microsoft covers API management through Azure APIM, event streaming through Azure Event Hubs, and batch ETL through Microsoft Fabric Data Factory, all unified within Microsoft Fabric. The same pattern holds for AWS and Google Cloud, where API Gateway, managed Kafka, and ETL services create equivalent ecosystem dependencies within their respective clouds.

Choosing any of these vendors increasingly means choosing a commitment to their ecosystem across all three paradigms. The lock-in spans API management, data integration, and event streaming simultaneously, which makes reversing a platform commitment significantly more expensive than any individual product replacement. Understanding which paradigms are covered, and which are not, before signing a platform contract is more important than comparing feature lists.

Event Streaming Is Where the Whole Market Is Heading

Databricks added Kafka-compatible APIs to its Zerobus Ingest service. Snowflake announced Datastream, its own Kafka-protocol-compatible streaming service. Most batch ETL vendors are adding CDC and near-real-time capabilities. Most iPaaS vendors are moving beyond simple connectors into event-driven workflow triggers, pub-sub capabilities, and agentic orchestration built on event streams. The direction is consistent across the entire landscape because AI agents and operational systems cannot wait for batch cycles.

The intentionally empty emerging row in the batch column reflects this clearly: the new entrants and new thinking are in the event streaming column, not the batch column. **Organizations that build their integration architecture around event streaming now will find it easier to serve future requirements.** Organizations that defer will find themselves retrofitting under pressure, which is always more expensive and more disruptive than building it in from the start.

Legacy Integration Shapes More of the Architecture Than Vendor Selection Does

Zero-ETL from AWS and Microsoft Fabric Mirroring work well within their respective ecosystems. They do not solve the SAP iDoc pipeline, the mainframe CICS integration, the EDIFACT B2B exchange, or the HL7 v2 feed from the hospital system. **Any realistic integration architecture in a large enterprise must handle legacy formats for years or decades alongside modern platforms.** Vendors with certified connectors and documented production deployments for mainframe, SAP, and Oracle CDC deserve more weight in evaluation than they typically receive. A platform that connects cleanly to the future but requires custom work to reach the past is not a complete platform for most enterprises.

08

CHAPTER 08

Event Streaming as the Foundation: What Comes Next

The direction is clear; the execution is not. Event streaming has become the architectural foundation of data integration in 2026.

The data integration landscape of 2026 is clearer in direction than it is in execution. Event streaming is the architectural foundation. Request-response and batch are consumer interfaces on top. The vendor landscape is consolidating around multi-paradigm platforms, and the remaining specialists are either deepening their niche or being acquired.

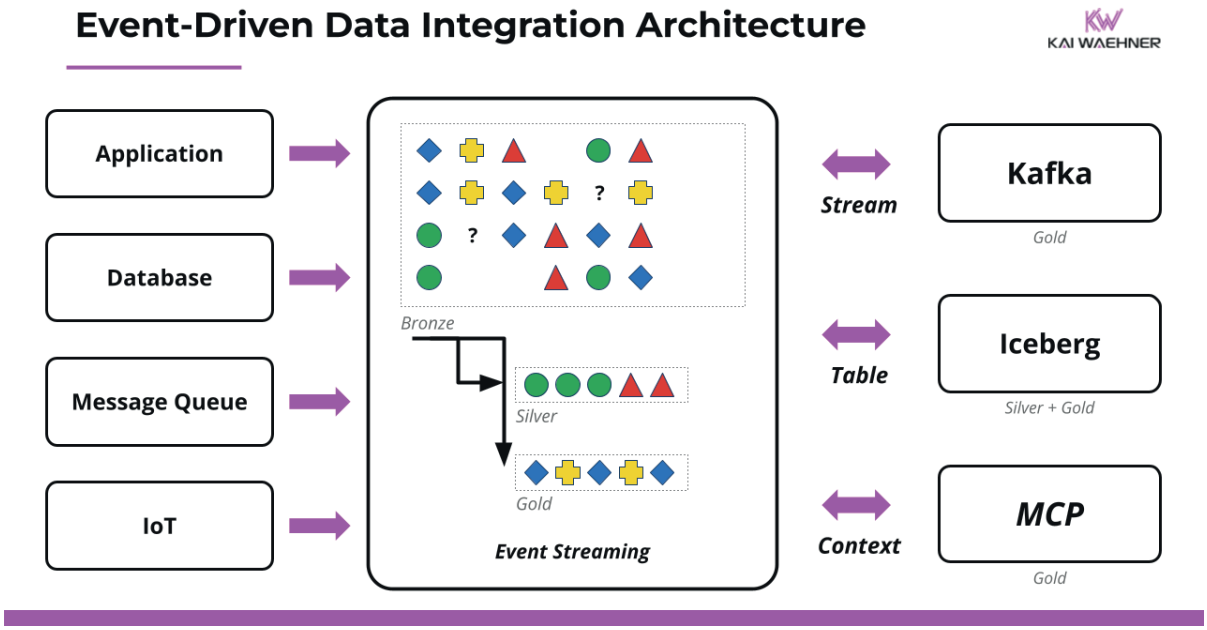

Figure 08. Event-Driven Data Integration Architecture

The architecture above captures how this works in practice. Data sources including mobile applications, databases, message queues, and industrial IoT connect to an event streaming backbone through purpose-built connectors and CDC tooling. Inside the backbone, raw data products flow through stream processing into curated data products. **Three output channels serve different consumer needs: a stream layer for operational systems, a table layer through Apache Iceberg for analytical systems, and a context layer through MCP for AI systems.** A single investment in the event streaming layer simultaneously serves operational systems, analytics platforms, and AI agents without duplication, which is why it is the most efficient foundation for organizations building for all three.

Data integration does not operate in isolation. It is the foundation layer beneath process intelligence and trusted agentic AI. The relationship between all three is explored in depth in The Trinity of Modern Data Architecture.

The vendors, paradigms, and trends in this landscape will continue to evolve. What will not change is the fundamental requirement: data must be clean, current, and connected. Everything else follows from that.

The architecture decisions made today will determine how much of this evolution is an opportunity and how much is a disruption. Which paradigm sits at the center. Which platforms handle the legacy edge. Whether governance is built in from the start or added under pressure later.

ABOUT THE AUTHOR

Kai Waehner

ADVISORY FIELD CTO

Kai Waehner is an Advisory Field CTO who has spent over 20 years helping enterprises make their most consequential data and AI architecture decisions. He works with Fortune 500 and Global 2000 companies across Europe, North America, the Middle East, Asia, and Australia, and follows a deliberately vendor-neutral approach in his advisory work that prioritizes the right architectural choice over the easiest sell.

His effectiveness as an advisor comes from range. He moves fluidly between a strategic conversation with a CIO and a deep architecture review with an engineering team, and across more than a dozen industries, from financial services and manufacturing to telecom, retail, and the public sector. That range is grounded in 100+ speaking engagements, from technical conferences like AWS re:Invent and QCon to CIO and CTO executive summits.

Kai is known for his independent technology landscapes for data integration, process intelligence, and trusted agentic AI. He also writes the blog at kai-waehner.de, covering industry use cases, technical best practices, and emerging topics for enterprise architects, CTOs, CDOs, and data engineers. Kai is available for advisory engagements, workshops, and keynotes worldwide, as well as media collaborations.

ABOUT THIS LANDSCAPE

This landscape is an independent analyst perspective, not a quantitative ranking. Vendor selection, tier placement, and market footprint reflect the author's assessment based on public information, vendor announcements, and two decades of enterprise architecture work with the platforms covered. No vendor paid for inclusion or placement, and no vendor reviewed or approved its section before publication. Revenue and customer figures are drawn from public vendor statements where available. The author runs an advisory practice through Kai Waehner GmbH and has held roles at TIBCO, Talend, and Confluent, spanning the three paradigms in this report. He also serves as Global Field CTO at Kestra, a workflow orchestration vendor outside this landscape's scope. This landscape was produced through Kai Waehner GmbH, independently of any vendor engagement.

KAI WAEHNER

2026

KEEP READING · STAY INFORMED

Data must be clean, current, and connected. Everything else follows from that.

BLOG & NEWSLETTER

Regular updates from the industry

Subscribe for the latest thinking on enterprise architecture, data integration, process intelligence, and trusted agentic AI.

kai-waehner.de

FREE BOOK

The Ultimate Data Streaming Guide

A practical resource covering streaming use cases, architectures, and real-world industry case studies.

Download Now

CONNECT

LinkedIn & X

Follow along for ongoing analysis of the enterprise integration landscape.

LinkedIn | X (Twitter)

MORE READING

The Trinity of Modern Data Architecture

How data integration, process intelligence, and agentic AI fit together.

Read more