Q3 2026

KAI WAEHNER LANDSCAPE Q3 2026

DATA STREAMING LANDSCAPE

Workload Focus, Operating Model, and Sovereignty for Data in Motion

Published: September 2026

Kai Waehner
ADVISORY FIELD CTO

kai-waehner.de

INSIDE THIS REPORT

Contents

The data streaming market changed shape in less than a year, and one theme now cuts across every user and every vendor: the sovereignty debate that started in AI arrived in data infrastructure.

IBM closed its acquisition of Confluent in March 2026 and deprecated its own Kafka products in Confluent's favor. In August, founder Jay Kreps announced he is stepping back. CoreWeave absorbed Bufstream into its internal AI platform in May. Decodable disappeared into Redis. StreamNative, the company that spent years arguing against Kafka, now ships its own Kafka offering.

Any one of these would justify an update. The reason for a new model is bigger. Who controls where your data runs, and under whose jurisdiction, moved from a compliance footnote to a board-level architecture decision. That question reshaped how this landscape is organized, and it returns in almost every chapter.

The Data Streaming Landscape Q3 2026 is not a ranking. No vendor pays to appear here. The analysis is based on my own experience advising enterprises on data and AI architecture, combined with ongoing research into vendor positioning, product developments, and adoption patterns. It is an independent practitioner perspective, not a formal research methodology like Gartner or Forrester.

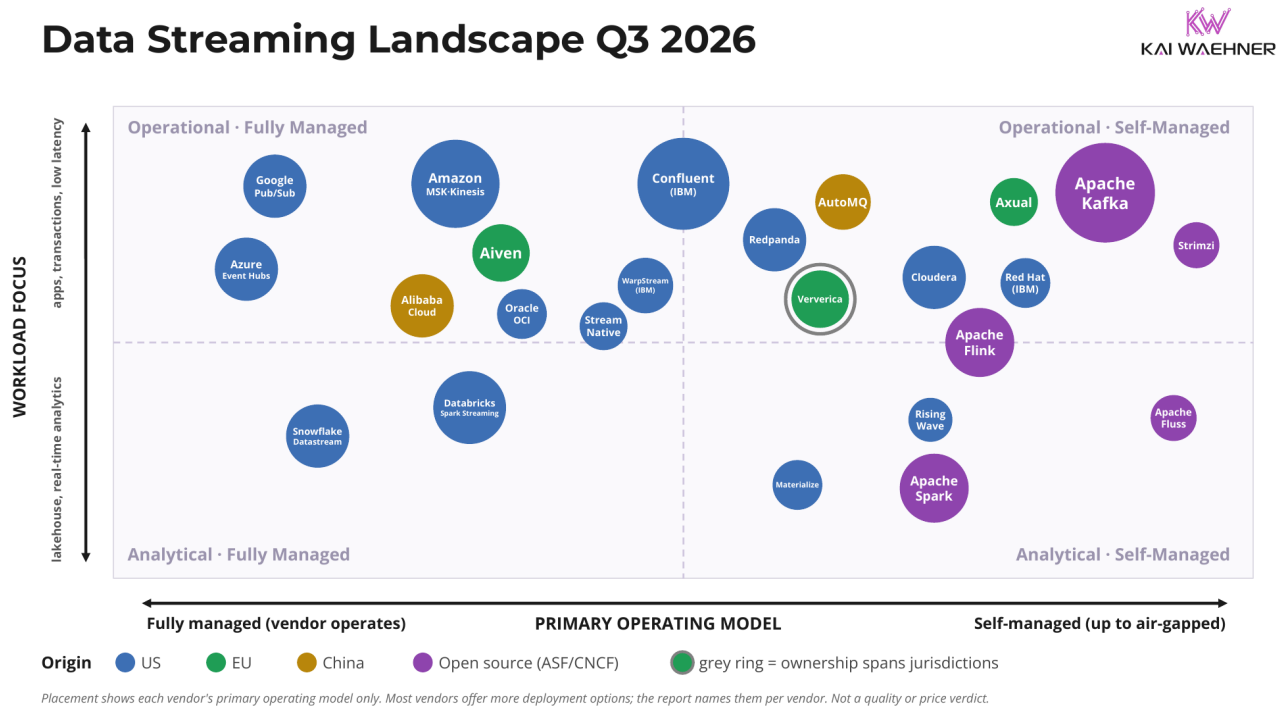

Figure 01. The Data Streaming Landscape Q3 2026. Bubble size shows market footprint, color marks vendor origin, and a grey ring marks ownership that spans jurisdictions. Position reflects workload and control, never platform quality or price.

01

CHAPTER 01

Scope and How to Read the Landscape

A definition of data streaming, the technologies the map includes and leaves out, how to read the axes, bubbles, and colors, and why the model changed this year.

Data streaming moves and processes data continuously as events happen, instead of in scheduled batches or on-demand requests. The category is built on event-driven architecture. Apache Kafka is the de facto standard and the protocol most vendors and frameworks now implement, used by more than 150,000 organizations worldwide. It is not the only implementation, and this landscape covers the alternatives as well: stream processing frameworks, cloud-native services that never adopted the protocol, and analytics platforms that now speak it.

How to read the landscape

The vertical axis is workload focus. Operational at the top: applications, transactions, low latency, decoupled microservices. Analytical at the bottom: lakehouse, real-time analytics, reporting.

The horizontal axis is the primary operating model. Fully managed and vendor-operated on the left. Self-managed on the right, up to air-gapped. Each vendor appears once, at the model that leads its business today. The vendor entries in the quadrant chapter below name every deployment option each one offers. This is where the sovereignty discussion lives.

Every bubble is one vendor or one open-source project. Bubble size shows market footprint, and bigger is not better. A focused, lower-cost service is the right answer for many teams. Color marks vendor origin: US, EU, China, or vendor-neutral open source under the ASF or CNCF. A grey ring marks companies whose ownership spans jurisdictions. Origin is a fact, not a verdict, and position reflects workload and control, never platform quality or price.

A bubble marks each vendor's center of gravity, not the limit of its portfolio. Many vendors span the control axis: Confluent sells Confluent Cloud, Confluent Private Cloud, and self-managed Confluent Platform, and Oracle, Cloudera, and others ship more than one deployment model. Each vendor entry in the quadrant analysis names the models it offers, and the sovereignty chapter covers why the spread matters.

One owner can appear more than once. Where a company sells several streaming products that lead with different operating models, each product is its own bubble with the owner in parentheses: Confluent (IBM) at the midline, WarpStream (IBM) on the managed side, Red Hat (IBM) on the self-managed side. Where the products share one operating model, the owner is one bubble: Amazon covers MSK and Kinesis, all of it vendor-operated inside AWS. Open-source projects follow the same rule: the bubble sits where the project reaches production today, not where its license would put it. Apache Kafka, Flink, and Spark carry broad managed ecosystems and sit closer to the midline than Strimzi or Fluss, which are run by hand. The map plots products where they are used, not corporate org charts.

Streaming itself is not a goal: the workload decides whether it is even the right tool. Plenty of use cases are served better by batch, by a message queue, or by a simple API call. I covered that in When NOT to Use Stream Processing and in Kafka vs Flink vs Spark: Do You Really Need Real-Time?.

Which technologies are included, and which are not

A technology appears on the chart if two things hold. Its core purpose is moving, processing, or storing data in motion. And it has production adoption that can be backed up publicly, through customer references and revenue rather than marketing claims.

Everything else lives in the text instead: table formats such as Apache Iceberg, consuming-only OLAP engines such as Apache Pinot, Apache Druid, ClickHouse, StarRocks, and Apache Doris, Kafka gateways, tooling, and the long tail of managed Kafka services including Canonical, DigitalOcean, Heroku, and Instaclustr.

Why the landscape model changed this year

Earlier editions organized vendors by deployment model. The 2023 and 2024 editions used self-managed, PaaS, and SaaS. The 2025 edition added BYOC as a row after WarpStream pioneered it. That model worked while most vendors picked one lane.

Two things broke it. First, deployment stopped being a vendor property. The same product family now ships as fully managed, BYOC, and self-managed at once, so a row no longer describes anything. Second, and more important, **sovereignty stopped being a footnote**. It is now a structural dimension of the buying decision, and the old model had nowhere to put it.

So the operating model became an axis rather than a set of rows, with each vendor plotted once at its primary model. The second change is the analytical half. Earlier charts contained streaming platforms only; Snowflake, Databricks, and the streaming-lakehouse projects appeared in the text and in separate posts. They are plotted now because the boundary between the streaming layer and the lakehouse is where much of the market movement happens, and a landscape that leaves it off no longer explains the market.

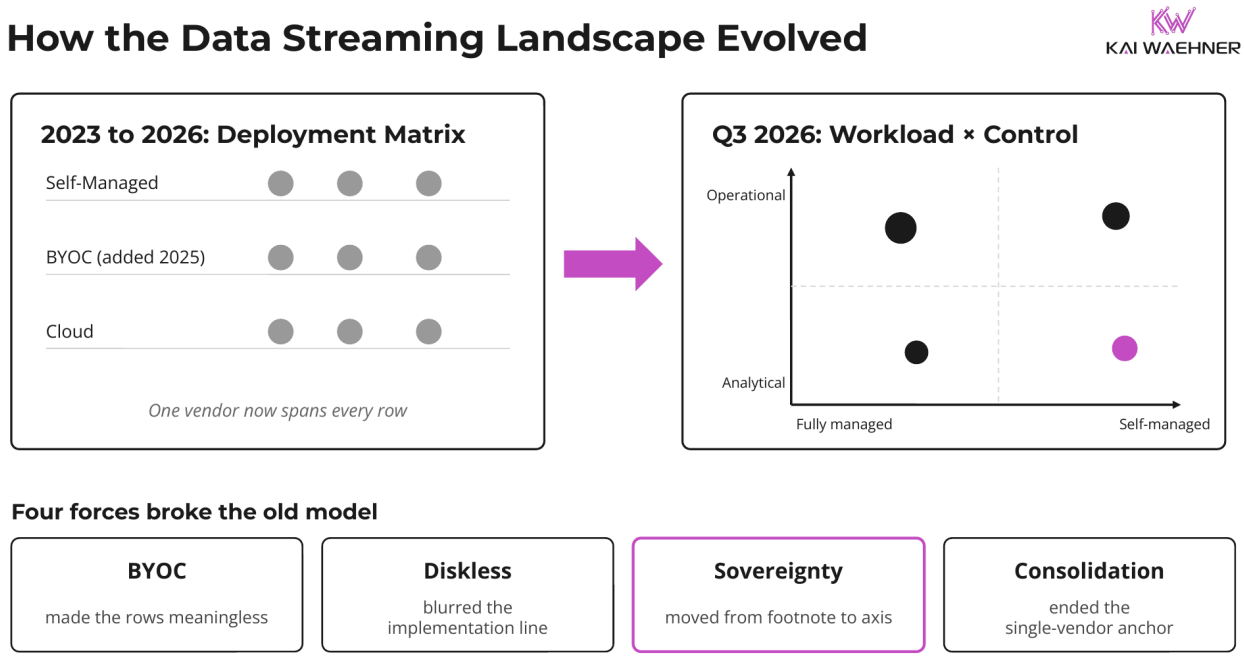

Figure 02. How the data streaming landscape model evolved: from the deployment matrix of earlier editions to the workload and control quadrant of Q3 2026.

02

CHAPTER 02

The Kafka Protocol and the Architecture Underneath

How vendors relate to the Kafka protocol, and the two architecture shifts inside the project: KRaft replaced ZooKeeper, and diskless designs move the write path to object storage.

The Kafka protocol: de facto standard for most of the market

The most important standard in this market is not a product. It is the Kafka protocol. It became the de facto interface through adoption rather than formal standardization. The S3 API went the same way for object storage: no standards body declared it, but enough tools aligned on it that it became the interface everyone implements. Vendors and frameworks meet it in two different ways, and a third group sits outside it entirely.

Apache Kafka distributions ship the open-source project or a fork of it: Apache Kafka, Strimzi, Confluent Platform, Amazon MSK, Aiven, Cloudera, Axual, and StreamNative UFK, a Kafka fork on its own storage engine. You get native semantics and access to the full Kafka ecosystem, including clients, Kafka Connect, and Kafka Streams. What each vendor packages and supports differs.

Proprietary Kafka engines re-implement the protocol underneath their own architecture. Confluent runs KORA in Confluent Cloud and added Freight for object-storage workloads. Redpanda has its R1 engine written in C++. WarpStream and AutoMQ are object-storage native. Azure Event Hubs is a protocol head. Snowflake Datastream is a stateless implementation. Databricks put a Kafka-protocol endpoint on Zerobus as well, though it accepts producers only, for ingestion into the lakehouse. Clients port via the protocol, which is the whole point, and workloads can move between these engines and real Kafka. The implementations are rarely 100 percent complete or identical, though: features, semantics, and operational behavior differ, and the differences surface in production rather than in the demo.

A third group never adopted the Kafka protocol at all. Amazon Kinesis, Google Pub/Sub, and Amazon EventBridge are proprietary SaaS, each tied to its own cloud. The trade-off is portability and reach: applications written against them run in one cloud only, and the surrounding ecosystem is a fraction of Kafka's. They work well for specific use cases inside their own cloud, and they are not a substitute for a Kafka-based backbone. Classic message brokers such as IBM MQ, RabbitMQ, and Solace sit outside the protocol too, but they solve messaging rather than streaming; my Data Integration Landscape covers them. The clearest evidence that the protocol won is that every hyperscaler now sells managed Kafka right beside its own proprietary service.

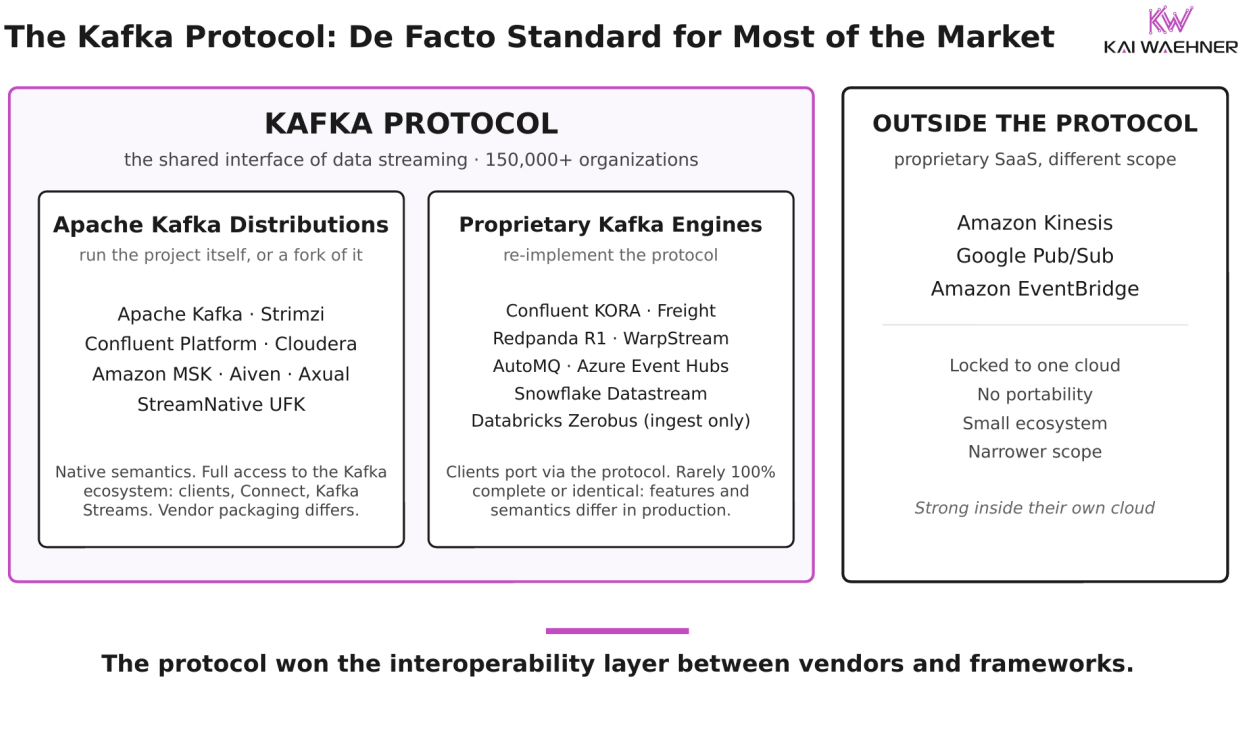

Figure 03. How vendors relate to the Kafka protocol, and who sits outside it.

Two architecture shifts inside Kafka

Two shifts inside Kafka get conflated because both remove something from the cluster, but they sit on different layers. KRaft changes how a cluster coordinates itself, diskless changes where the data lives, and a team can adopt one without the other. Most will end up with both.

Metadata: ZooKeeper removal through KRaft

ZooKeeper removal, delivered through KRaft, took a second distributed system out of every Kafka cluster. ZooKeeper stored the metadata and coordinated the brokers, and operators had to run, tune, and secure it just to get Kafka running. KRaft moved that job into Kafka itself. The transition is complete, community support for the last ZooKeeper-based releases is running out in 2026, and every new cluster should start without ZooKeeper.

Storage: diskless Kafka and what it changed in the market

Diskless is the second shift, on a different layer entirely: it removes the partition leader and local disk from the write path, so brokers stop being the durability layer and object storage takes over.

Diskless matters because of cloud economics. The expensive part of a high-volume Kafka cluster in the cloud is not storage. It is cross-availability-zone replication traffic between brokers, billed per gigabyte, on every message. Writing straight to object storage lets the cloud provider handle durability and removes that traffic. For latency-relaxed workloads, still measured in tens of milliseconds rather than single digits, enterprises running gigabytes per second saw their cost structure change rather than improve at the margin.

WarpStream started that conversation, and its acquisition changed the competitive picture. Confluent had a real gap at extreme scale where no strict latency SLA applied, and buyers often chose MSK or self-managed Kafka on cost alone. Buying WarpStream gave Confluent a price-competitive answer for exactly those workloads. The pattern repeated across the market: everyone now has a diskless story.

Where diskless Kafka stands today

In Apache Kafka: not yet. KIP-1150 was accepted in March 2026 as a directional proposal, and the implementation KIPs are still under discussion. Mainline releases have no diskless topic configuration.

Products are ahead of the project: four ship diskless today. Aiven Inkless is an open fork that Aiven intends to sunset once the work lands upstream. Confluent Freight reached general availability in early 2025. WarpStream and AutoMQ were built object-storage-native from the start. Tiered storage is a different thing again: it moves sealed segments to object storage and leaves the active write path on local disk.

03

CHAPTER 03

The Four Quadrants: Vendor by Vendor

Every vendor and open-source project on the map, alphabetically within its quadrant, with the deployment models each one offers.

Vendors appear alphabetically within each quadrant. Order carries no ranking.

Operational and Fully Managed

The top-left quadrant holds vendor-operated services for operational workloads: applications, transactions, and low-latency pipelines where someone else runs the infrastructure.

Aiven runs managed Apache Kafka, Kafka Connect, and Apache Flink on all major hyperscalers, with a BYOC option for teams that need the data plane in their own account. Its Inkless clusters are the open diskless fork covered in chapter 2, which Aiven intends to retire once the work lands upstream in Apache Kafka. The company is headquartered in Helsinki, which places it under EU jurisdiction.

Alibaba Cloud anchors the Chinese market for managed data streaming: managed Kafka, Realtime Compute for Flink, RocketMQ, and EventBridge. For enterprises operating in China it is the default backbone. ByteDance Volcano Engine, Huawei Cloud, and Tencent Cloud offer comparable managed Kafka and Flink services.

Amazon covers the most ground: MSK for managed Apache Kafka, Managed Service for Apache Flink as a separate processing service, and Kinesis as its own proprietary option. All of it is vendor-operated: provisioned MSK sits closest to PaaS, Kinesis and MSK Serverless are pure SaaS, and none of it runs outside AWS. MSK provides Kafka infrastructure without the governance and processing layers of a complete data streaming platform (DSP), so expect assembly from additional AWS components. MSK Replicator now explicitly targets migrations from rival Kafka services, naming competitors as sources.

Azure Event Hubs is a protocol head, fully managed, SaaS only, and deliberately narrow. Good for ingestion and transport inside the Microsoft ecosystem. Stateful processing needs additional services such as Azure Stream Analytics, and Microsoft Fabric Real-Time Intelligence covers the analytical side.

Confluent (IBM) remains the reference for a complete data streaming platform: Kafka, Flink, connectors, governance, and Tableflow for Iceberg in one product, with Confluent Platform for self-managed deployments and Freight for object-storage economics. Complete is meant literally: no other vendor covers ingestion, processing, governance, and lakehouse integration at this depth in one product. It sits at the midline of this map deliberately: Confluent Cloud leads the business, while Confluent Platform and Private Cloud carry many of the most critical self-managed and air-gapped deployments. Its install base spans a large share of the Fortune 500 with deep, public reference stories. The IBM chapter below covers what the acquisition changes.

Google Cloud runs Pub/Sub for messaging and Dataflow for stream processing as its proprietary pair and added Managed Service for Apache Kafka as the protocol answer. The Kafka service is real but younger than MSK, and ecosystem depth is still catching up. All offerings are vendor-operated.

Oracle runs two Kafka-facing services on OCI: the older OCI Streaming with its Kafka-compatible API, and OCI Streaming with Apache Kafka, a fully managed distribution of the open-source project that reached general availability in late 2025 and prices itself aggressively against MSK and Confluent Cloud. GoldenGate feeds change data from Oracle databases into either. The distinguishing feature is where OCI itself can run: public regions, the EU Sovereign Cloud, Dedicated Region, and Cloud@Customer bring the same managed service into a customer-controlled data center, which matters for the sovereignty chapter. For Oracle-centric enterprises it is the natural backbone. Ecosystem depth and processing services still trail the larger hyperscalers.

WarpStream (IBM) pioneered diskless BYOC: stateless agents in your VPC, data never leaving your account. The split has a real upside. The control plane carries all the complex coordination logic, so customers deploy only stateless services that are easy to operate. WarpStream never needs access to the data plane at all, which is stronger isolation than most BYOC offerings provide. The caveat is the flip side of the same design: it cannot run without that vendor-operated control plane, which is why it sits at the midline and not on the right.

> **GUIDANCE**
>
> The question here is when a fully managed deployment fits, not which vendor is best, since several of them also sell self-managed and private-cloud editions. Treat it as an enterprise architecture decision. If your consumers, budget, and regulators all sit comfortably in one cloud, the ecosystem-native managed service is often enough. Buy the complete platform when governance, processing, and support across many teams are the real requirement. A single team with one non-critical use case does not need it.

Operational and Self-Managed

The top-right quadrant holds the open-source projects and the vendors whose primary model is software you run yourself, up to fully air-gapped.

Apache Flink is the de facto standard for stateful stream processing. Many commercial processing offerings on this landscape run Flink underneath, though not all: Google Dataflow is built on Beam, Databricks on Spark, and Redpanda and Snowflake use their own engines. Kafka Streams is the lightweight alternative: stream processing as a library inside your application instead of a separate cluster. I compared the two in Apache Kafka and Apache Flink: A Match Made in Heaven.

Apache Kafka is the base of nearly everything else on this map and the largest bubble on the chart for that reason. Kafka 4.x removed ZooKeeper entirely, as covered above. It also brought queue semantics into the protocol through Queues for Kafka (QfK), generally available since Apache Kafka 4.2, which closes a long-standing gap against classic message brokers.

AutoMQ re-architected Kafka on object storage under Apache 2.0, with production references at JD.com, Grab, and Tencent Music, and first-party status on Tencent Cloud. It is the strongest China-origin challenger and increasingly visible in Western evaluations. BYOC leads, with control plane and data plane both inside the customer's cloud account. The Apache 2.0 distribution covers self-managed on Kubernetes or in the data center, and Tencent Cloud sells it as a first-party managed service.

Axual has built sovereignty-first Kafka on Strimzi for a decade, starting in Dutch banking and growing into grid operators and public-sector references across the EU. On-premises and air-gapped deployment is the design point, and a managed cloud offering exists as well. Its growth is a useful signal in itself: sovereignty is winning deals in industries that used to buy on features alone. The company is headquartered in the Netherlands.

Cloudera ships Kafka and Flink for on-premises and hybrid enterprises, one of the few complete self-managed platforms left, with strength in regulated industries that never went all-in on public cloud. Self-managed and air-gapped first, with managed cloud available.

Redpanda competed for years on performance and cost benchmarks that independent tests often did not confirm, then pivoted its marketing almost entirely to the Agentic Data Plane. The 2026 messaging re-couples both stories: streaming pays the bills, agents are the bet on the future. The most substantial new capability in 2026 is Shadowing: in-broker replication that migrates topic data, schemas, offsets, and ACLs from Confluent and other Kafka services without MirrorMaker. Deployment spans SaaS, BYOC, self-managed, and air-gapped.

Red Hat (IBM) sells streams for Apache Kafka, the supported distribution of Strimzi on OpenShift, with its own console, a Kroxylicious-based proxy, and Debezium for change data capture. It is self-managed by design and runs wherever OpenShift runs, including air-gapped. Since the Confluent acquisition closed, IBM owns two supported Kafka distributions: one commercial platform and one built on the CNCF operator. Red Hat kept shipping through 2026, and the overlap is the open roadmap question the IBM chapter covers. For OpenShift-standardized enterprises it remains the low-friction choice.

StreamNative spent years positioning Apache Pulsar against Kafka, much of it through benchmark marketing rather than ecosystem building, and I wrote about those myths at the time. Pulsar never gained broad enterprise traction. Now the company ships UFK, its own Apache Kafka fork running on the Ursa storage engine, across managed cloud, BYOC, and self-managed deployments. The leading Pulsar vendor sells Kafka to win share, which says more about the protocol than any benchmark ever did.

Strimzi runs Kafka on Kubernetes through a CNCF operator, and it is a serious deployment strategy rather than a hobby. Honeycomb publicly migrated from a commercial platform to Strimzi. Free as in license, not free as in effort.

Ververica has expanded from a Flink runtime into a platform with the VERA engine, managed cloud, BYOC on AWS and Azure, self-managed deployment, and integration with Fluss and Paimon. It also carries the grey ownership ring, explained in the sovereignty chapter.

> **GUIDANCE**
>
> Self-managed is the right decision when a vendor-operated control plane is not acceptable: regulation or jurisdiction requires it, continuity planning demands it, or the volume makes managed pricing uneconomic. Apache Kafka on Strimzi is the lowest-cost entry and carries the highest operational burden. A commercial distribution buys support, tooling, and a roadmap. Budget for the operations team before the license, because free as in license is not free as in effort.

Analytical and Fully Managed

The bottom-left quadrant is new in this edition: analytics platforms that now speak the Kafka protocol and pull streaming into the lakehouse perimeter.

Databricks carries the largest stream processing footprint of any analytics platform through Spark Structured Streaming, extended by real-time mode, now generally available with millisecond latencies, and by Zerobus for Kafka-protocol ingestion. The Kafka support in Zerobus accepts producers only: data streams in, nothing consumes back out. The design point remains analytical, and the classic deployment model splits a Databricks-operated control plane from a data plane in your own account.

Snowflake built more than an ingestion path. Datastream is a complete Kafka protocol implementation: stateless processors writing to object storage with a separate metadata service, supporting producers and consumers, currently in private preview. It is SaaS only, inside the Snowflake perimeter; there is no BYOC or self-managed path. This is real engineering and deserves to be named as such. The open question is architectural rather than technical: whether an analytics platform's perimeter is the right home for an operational backbone. I covered the distinction in Why Databricks and Snowflake Speak the Kafka Protocol.

> **GUIDANCE**
>
> These platforms fit when the consumers of the stream are analytical and already live in the lakehouse. Both offer two ways in: the established connector path, and a native Kafka-protocol path that lands topics as governed tables, with Snowflake Datastream still in private preview. Both remain analytics platforms first, with one cloud perimeter and no self-managed path. Use them to shorten the path into the lakehouse, and keep the operational event log outside the perimeter where polyglot consumers and delivery guarantees matter.

Analytical and Self-Managed

The bottom-right quadrant holds the open-source engines and the streaming-storage projects that are rebuilding the analytical side of streaming in the open.

Apache Fluss is the newest building block on this map, and the one to watch, because it fills a gap nothing else covers: streaming storage built for analytics. It is columnar where Kafka is row-oriented, Flink-native, and donated to the ASF in 2025, with production scale proven at Taobao, though adoption outside the Alibaba ecosystem is still young. Fluss holds the hot, changing data; Apache Paimon is the table format where that data settles for long-term storage. Kafka protocol compatibility is on the roadmap, not shipped. It complements the operational backbone rather than replacing it.

Apache Spark is the workhorse of analytical stream processing at lakehouse scale, and it keeps getting faster. Structured Streaming latency dropped from seconds toward milliseconds over the years, and the new real-time mode, generally available on Databricks and landing in open-source Spark 4.x, reaches sub-second processing. For most analytical use cases that is more than enough. The old micro-batch criticism also carries less weight than it sounds, since Kafka Streams batches records internally for throughput as well. The sharper distinction is purpose: Spark dominates analytical workloads, while Flink and Kafka Streams carry far more operational ones.

Materialize and RisingWave both dropped the streaming database label in 2026, repositioning around real-time platforms and operational data freshness. Both lead with managed services and both ship self-managed editions on Kubernetes; RisingWave's is open source, Materialize's requires a license key. Why the streaming database label never sold is covered in the lakehouse chapter.

GUIDANCE

This quadrant is where the analytical side of streaming is being rebuilt in the open, on several engines rather than one. Spark carries most of the production footprint today. Fluss and Paimon are the Flink-native newcomers with the largest architectural ambition and the youngest adoption outside Alibaba. Materialize and RisingWave serve fresh views for operational analytics. Evaluate each against the engine your lakehouse already runs on, and treat all of them as complements to the operational event log rather than replacements for it. Maturity here is measured in years, not quarters.

04

CHAPTER 04

Confluent Inside IBM: What Changes for the Market

What IBM bought, what the market loses, and the open questions for anyone making a long-term platform decision.

The facts first. IBM closed the acquisition in March 2026. In August, Jay Kreps announced he is stepping back after twelve years, with the long-time Chief Product Officer taking over, so product leadership continues from inside the company. IBM also deprecated Event Streams and Event Processing, naming Confluent as the go-forward Kafka and Flink offering. The consolidation is rational, and IBM's reach into regulated industries and global accounts can take Confluent into places a standalone company entered slowly.

The portfolio after the deal has more Kafka in it than the announcement suggested. Confluent brings Confluent Cloud, Confluent Platform, and Confluent Private Cloud, plus WarpStream for diskless BYOC. Red Hat, part of IBM since 2019, brings streams for Apache Kafka, the supported distribution of Strimzi, and the Kroxylicious proxy project. The two sides already share code: Confluent Private Cloud Gateway uses Kroxylicious as its proxy engine. IBM's own Event Streams and Event Processing are the retired products, while the Red Hat line kept shipping releases in 2026. The result is one owner with two supported Kafka distributions, the commercial Confluent Platform and the Strimzi-based Red Hat product, and the market has not heard how they will be positioned against each other. This is also why IBM is not one bubble on the map. The three products lead with three different operating models, so each is plotted where it is sold, with the owner in parentheses.

What changes is the story the market has heard for a decade. Confluent grew by evangelizing Kafka as the central nervous system, a backbone that displaces legacy messaging and batch middleware. Inside IBM, that pitch shares a portfolio with IBM MQ, webMethods, App Connect, and DataStage. The likely evolution is from displacement to coexistence. IBM will position Confluent for data streaming, and data streaming alone: the event backbone that moves and processes events in real time. Transactional messaging stays with IBM MQ, API-led integration stays with webMethods and App Connect, batch stays with DataStage, and AI stays with watsonx. Confluent used to sell the whole of that scope under one banner. Inside IBM it sells one slice of it, next to sibling products that own the rest. The portfolio logic is sound. The message is quieter, and a category built on loud evangelism now has nobody with that reach carrying it.

Three open questions matter for buyers, and they are questions rather than verdicts. How does the combined roadmap balance Confluent against sibling products with overlapping scope, including Red Hat's investment in Strimzi and Kroxylicious? Does the pace of a founder-led product organization survive the transition? And what happens to Flink? Confluent marketed Flink heavily, and commercial adoption never scaled the way the market expected. Whether digital natives with large-scale streaming and processing workloads now buy that from IBM, or prefer another vendor or open source, remains open.

None of this changes what the platform does today, and Confluent remains the most complete platform on this map, spanning fully managed and self-managed. All of it belongs in a multi-year platform decision. For every challenger the independent-Kafka story just became a sales asset, and the migration tooling now shipping across the market shows that everyone knows it.

05

CHAPTER 05

Data Sovereignty and the Return of Self-Managed Deployments

Self-managed and private deployments are recovering ground, and the reasons are jurisdiction, control, and continuity rather than cost.

For a decade the direction of travel was one way: out of the data center and into the cloud. The direction has changed. Self-managed and private deployments are recovering ground, and the shift is visible in vendor numbers rather than opinion pieces.

Confluent is the clearest public evidence. The company built its growth story around Confluent Cloud and told investors so repeatedly. Then the hybrid side reasserted itself. In Q1 2025 Confluent Platform, the self-managed product, grew 18 percent year over year, its strongest first-quarter growth in three years. Management described that deployment flexibility as resilience that makes growth less exposed to shifts in cloud spending. In October 2025 the company launched Confluent Private Cloud, bringing cloud-native operations behind the firewall for regulated industries. A cloud-first vendor investing in on-premises is not a retreat. It is a read of where regulated demand actually is. The same read explains why Confluent fits IBM: the group has run a hybrid cloud strategy since 2020, and its largest accounts sit in banking, insurance, telecommunications, and the public sector, where hybrid and on-premises deployment is the norm rather than the exception.

The reasons are jurisdiction, control, and continuity rather than cost. Regulation is loudest in Europe. The EU's Cloud and AI Development Act arrived in June 2026 with a cloud sovereignty framework and an open-source-first procurement principle. DORA treats concentration on a single hyperscaler as a risk to manage. The CLOUD Act tension underneath remains unresolved.

This is not a European problem. Middle East enterprises building national capability ask the same question. APAC data-localization rules enforce it. US public sector and defense buyers run air-gapped for their own reasons. China requires in-country operation. US, China, and EU policies pull in different directions, and any enterprise operating across all three needs an answer. Only the weighting differs by region.

This is why the operating model is an axis here rather than a paragraph. Fully managed SaaS brings the vendor's jurisdiction and operations with it. BYOC keeps data in your account, but it differs by vendor, and some BYOC products cannot run without the vendor's control plane. Self-managed and air-gapped deployments put the software fully under your control, which is why a US vendor's license is fine for most sovereignty requirements the moment the software runs in your own VPC or data center.

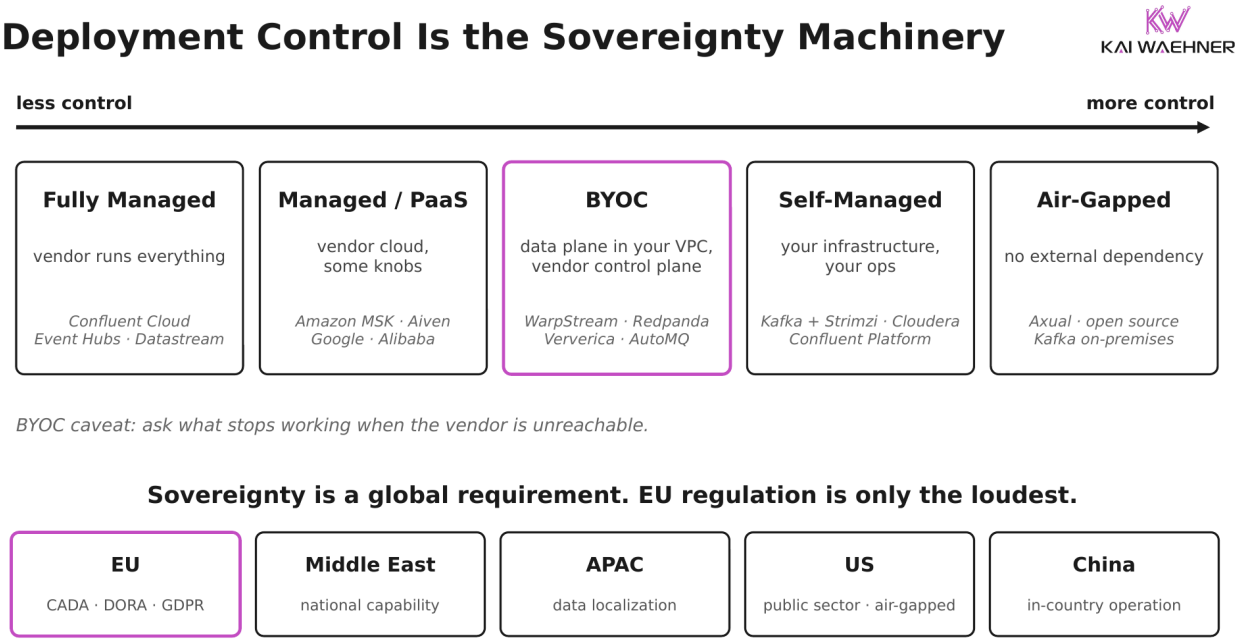

Figure 04. Deployment control is the sovereignty machinery: from fully managed SaaS through BYOC to self-managed and air-gapped.

The strongest argument for all of this came from AI, not from data infrastructure. In June 2026 the US government suspended access to Anthropic's two most capable models for foreign nationals, and the vendor disabled them globally until access was restored weeks later. A leading vendor was switched off by a government its customers had not chosen. That episode moved sovereignty from a slide in a compliance deck to a board agenda item, and the lesson does not stop at the model layer. Every dependency in the stack carries the same question, including the platform carrying your operational data. I covered the full argument in the Trusted Agentic AI Landscape Q3 2026.

The European vendor field is small but growing. Axual and Aiven are the two European vendors on this map, and Axual's expansion into EU public-sector references shows sovereignty winning deals on its own. The China field is broader than most Western coverage admits, anchored by Alibaba Cloud. AutoMQ, an independent company building on Apache-2.0 code, is the strongest challenger from that region.

Ververica is the case where a single origin label would mislead. It is legally a German GmbH. The German federal government was involved when Alibaba acquired it, and only a German citizen may serve as CEO. The ownership chain runs through Alibaba Netherlands to the Cayman-registered Alibaba Group, so there is no direct structural directive authority from China, and the company has passed procurement reviews at German blue-chip enterprises. Alibaba's ultimate ownership is also a fact. Both belong in the picture, which is exactly what the EU fill and the grey ring together show.

06

CHAPTER 06

Data Streaming Meets the Lakehouse: Zero-Copy in Theory and Practice

The topic becomes the table at every vendor. Most enterprises still store the data twice, and that is often the better pattern.

The analytical half of the chart is new, and this chapter is the reason it exists. The boundary between the streaming layer and the lakehouse is where much of the market movement happens.

The direction is the same at every vendor: the topic becomes the table. Confluent Tableflow exposes topics as Iceberg tables. StreamNative Ursa writes topics as Iceberg and Delta. Snowflake Datastream lands Kafka traffic as governed tables. Databricks Zerobus takes Kafka producers into the lakehouse. Fluss tiers into Paimon and Iceberg. Five implementations, one direction.

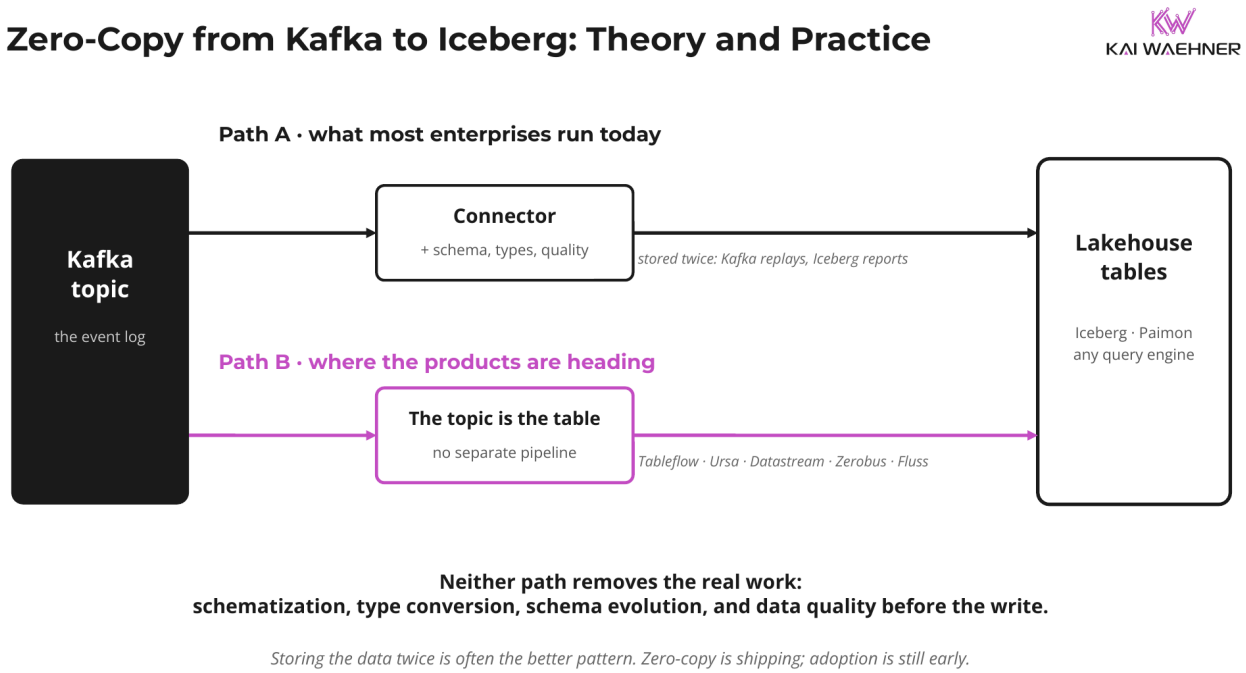

Figure 05. Zero-copy from Kafka to Iceberg in theory and practice: the connector path most enterprises run today, and the path where the topic itself becomes a governed table.

The reality in most enterprises is different. Zero-copy is rare, in the same way zero-ETL turned out to be rare. Most teams still run a connector from Kafka into object storage, and the data exists in both places. This is not a failure. **Storing the data twice is often the better pattern.** Kafka holds the raw event log for replay, reprocessing, and the operational workloads that need low latency, similar to a write-ahead log. Iceberg holds the structured, governed tables for analytics. A Kafka topic cannot serve the analytical job itself: it is built for sequential reads by many consumers, with no columns, no indexes, and no statistics, which is exactly why analytics needs a table. Two roles, two retention profiles, two access patterns.

What no copy count solves is the engineering in between: schematization of raw events, type conversion between Avro or Protobuf and Iceberg, schema evolution that stays compatible in both directions, and data quality rules applied before the write. I covered these in detail in Data Streaming Meets Lakehouse and in the Shift Left Architecture 2.0.

How Kafka, Flink, Fluss, and Paimon fit together

These four get confused constantly, so here is the short version, layer by layer. The figure below shows the same relationship. Kafka is the event log: row-oriented, replayable, low latency, consumed by any client in any language. Fluss is streaming storage built for analytics: columnar, optimized for queries and updates rather than polyglot consumption. Paimon and Iceberg are table formats where colder data lands. Flink is the processing engine that reads and writes all of them.

Fluss and Paimon extend the analytical side. They do not replace the operational event log, and the Fluss project says as much. For a Flink-centric lakehouse they are worth evaluating seriously. For an operational backbone serving polyglot consumers with delivery guarantees, Kafka's ecosystem maturity remains decisive.

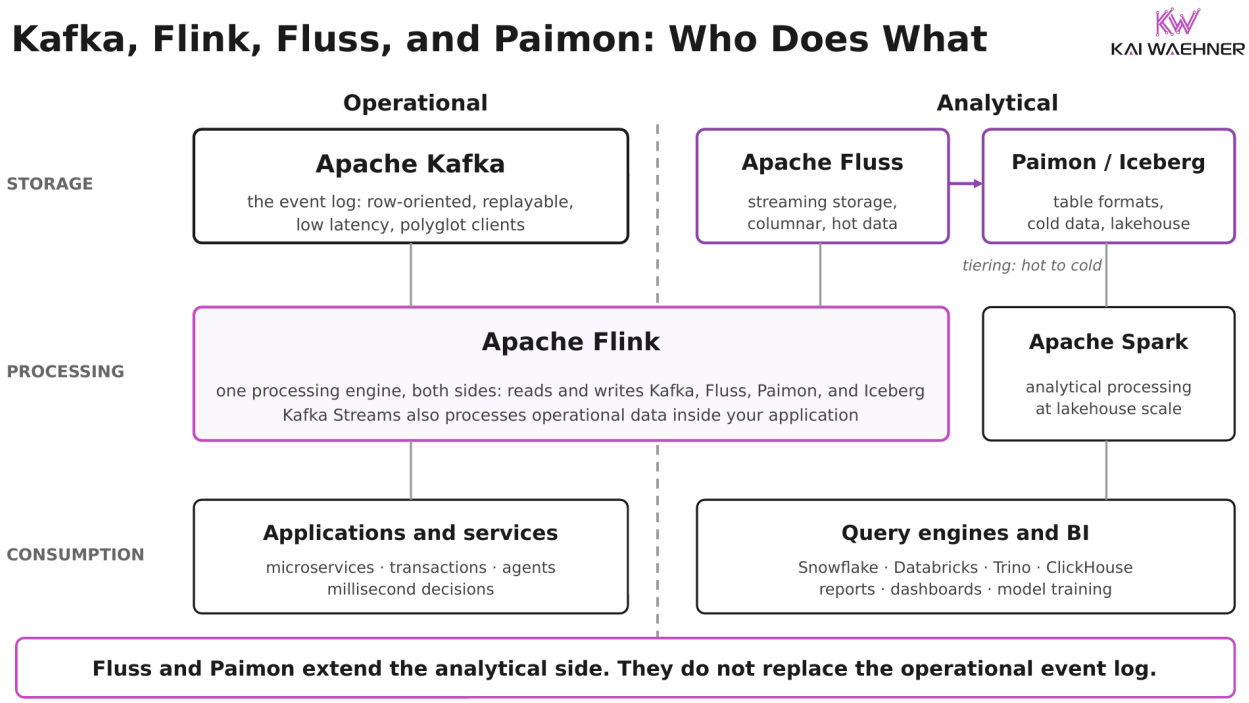

Figure 06. Kafka, Flink, Fluss, and Paimon across the storage, processing, and consumption layers.

Why streaming databases never became a category

Streaming databases were supposed to be a category. They never got there. The label promised a database and a stream processor in one, and buyers heard complexity twice. Nobody woke up needing a streaming database. They needed fresh data in a system they already understood, and explaining incremental view maintenance to a budget owner is a hard sell. The products were often good. The category was unsellable.

In 2026 the vendors dropped the word themselves. RisingWave now calls itself a real-time platform for agentic AI. Materialize sells operational data freshness. Fluss and Paimon approach a related problem from the storage layer rather than the query layer, which may prove the more durable path. The lesson travels: categories that describe architecture instead of outcomes do not survive contact with a budget owner.

07

CHAPTER 07

Governance Beyond a Single Platform: Lineage, Catalogs, and Gateways

Every platform governs itself deeply and sees its neighbors only at the surface. Open lineage, platform-independent catalogs, and Kafka gateways close the gap.

Data governance beyond a single platform: lineage and catalogs

Every platform in this landscape ships its own governance. Confluent has Schema Registry, lineage, and catalog synchronization. Databricks has Unity Catalog. Snowflake has Horizon. Each is strong inside its own layer, and none of them sees the others. Each vendor also pitches its catalog as the governance layer for everybody else. In practice every one of them governs its own platform deeply and sees the neighbors only at the surface, and that gap is the problem.

This is the problem enterprises face. Buy ten tools and you get ten disconnected lineage graphs, while the auditor asks one question that spans all of them. Platform-independent catalogs assemble the cross-platform picture: commercial ones such as Atlan, Collibra, and Microsoft Purview, and open-source ones such as DataHub and OpenMetadata. Open standards like OpenLineage keep the lineage itself from being trapped in one vendor's console. The argument matters more as vendors consolidate: when platforms get acquired, a governance layer coupled to one vendor turns every platform change into a governance migration. I made the full case in Beyond Enterprise Data Lineage.

Kafka gateways and proxies: a new layer for governance and migration

A proxy sits transparently between Kafka clients and the cluster and applies policy in the path. A gateway does the same and additionally exposes new interfaces on top of the stream. The market uses both words loosely, and most products do both. The policies are the point: encryption, masking, multi-tenancy, routing, quota enforcement, protocol mediation, and migration. On the gateway side, topics become HTTP endpoints, webhooks, or server-sent events for consumers that will never run a Kafka client.

The category emerged because governance, not throughput, became the bottleneck. Large enterprises run many clusters from several vendors, with dozens of teams sharing them, and the controls needed there do not exist inside a broker. Conduktor shipped the first enterprise-grade Kafka proxy in 2022, when the idea was still contested. Today the pattern is everywhere: Confluent ships a gateway built on Kroxylicious, the CNCF Kafka proxy originating at Red Hat, which is also available directly as an open-source project. Gravitee and Kong come from API management and bring event-native policies alongside HTTP. Aklivity Zilla focuses on protocol translation, exposing topics as HTTP, SSE, gRPC, MQTT, or WebSocket. Apache APISIX and an Envoy Kafka filter cover the infrastructure-tooling end.

The trade-offs are real: a proxy is another hop, another failure domain, and another thing to operate. I covered when it pays off in Kafka Proxy Demystified. This layer has grown enough that it deserves its own landscape.

08

CHAPTER 08

Ten Data Streaming Trends to Watch Through 2027

Consolidation, sovereignty, diskless architectures, the lakehouse boundary, and the discipline to say no.

Ten forces will shape this market over the next eighteen months.

- **Consolidation and migration in every direction.** IBM and Confluent, Redis and Decodable, CoreWeave and Bufstream, Redpanda and Oxla. Migration tooling became a competitive weapon on every side. Confluent released KCP, a free open-source CLI that automates moves from hosted Kafka services to Confluent Cloud. Amazon and Redpanda ship their own paths, and migrations run in both directions between commercial platforms and open source.
- **Sovereignty and deployment control as table stakes.** Every serious vendor now leads with where the software runs, and self-managed deployment is recovering ground it lost a decade ago.
- **One protocol, diverging implementations.** Compatibility claims multiply while semantics and ecosystem depth differ in production.
- **Diskless and object-storage-first architectures.** Accepted as a direction in the Apache Kafka project through KIP-1150, shipped by four vendors ahead of the standard, and years from production maturity in open-source Kafka itself.
- **Streaming meets the lakehouse.** Topics materializing as tables is the direction. Connectors and two copies remain the reality for most enterprises.
- **Operational maturity as a buying criterion.** Queues for Kafka (QfK), in-broker replication, and recovery guarantees moved from roadmap to requirement.
- **Stream processing beyond Flink SQL.** Spark real-time mode is generally available, the Flink DataStream API remains the choice for complex logic in Java, and complex event processing (CEP) returns for pattern detection that SQL expresses poorly.
- **Governance beyond the platform boundary.** Platform-independent catalogs and open lineage standards matter more as the vendor field consolidates.
- **Real-time context for agentic AI.** Redis, Confluent, and Snowflake all now sell a real-time context engine, and MCP endpoints are appearing across streaming platforms.
- **The discipline to say no.** The market matured enough to right-size. Not every workload needs a stream processor, and knowing when batch or a queue wins is now a mark of good architecture.

A dedicated trends deep dive follows in a separate post in a few weeks, as every year.

09

CHAPTER 09

Five Questions to Ask Before Choosing a Data Streaming Platform

Workload, control, jurisdiction, economics, and exit: five questions, and where each one lands on the map.

Start with the workload, not the vendor. The five questions below place a platform on this landscape more accurately than any marketing claim, and each maps to one element of the map: the two axes, the color and the ring, and the guidance in the quadrant chapter.

- **Workload.** Which parts of your architecture are operational, with polyglot consumers and delivery guarantees, and which are analytical? A platform that serves one well may serve the other badly, and forcing both through one product is how teams end up paying for capability they never use. The answer places a platform on the vertical axis.
- **Control.** What does your regulator require about where data runs and who can access it? If the vendor became unreachable tomorrow, what stops working? Can you run the software yourself if the commercial relationship ends? The answers place a platform on the horizontal axis.
- **Jurisdiction.** Whose law binds the vendor, and the parent behind the vendor? The map shows this as the bubble color and the grey ring, and the sovereignty chapter explains why a clean answer here is a fact, not a verdict.
- **Economics.** What does the cost model look like under object-storage economics rather than last year's broker math, at your actual throughput and retention? The map does not show price, by design. The diskless section of chapter 2 explains why the math changed.
- **Exit.** What is the migration path in, and more importantly out? Protocol compatibility makes the technical migration plausible. Governance, connectors, and operational tooling are where the real switching cost sits, and the vendor entries name each platform's deployment options for exactly this reason.

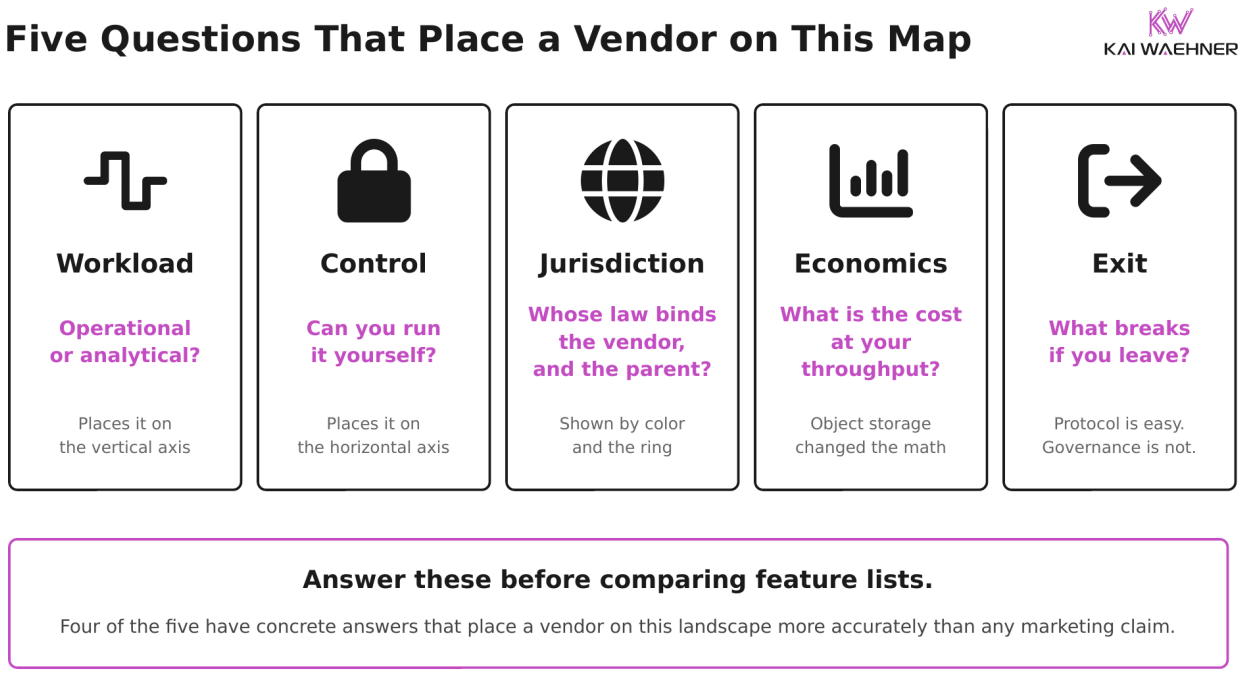

Figure 07. Five questions that place a vendor on this map. Workload and control are the two axes, jurisdiction is the color and the ring, economics and exit are covered in the text.

Four of the five have concrete answers before you ever read a feature list. Only economics needs your own numbers.

A CLOSING NOTE

Data streaming in the Trinity of modern data architecture

Data streaming is the foundation of the Trinity of modern data architecture. It is how most enterprises implement event-driven architecture, because Kafka brings the guarantees that layer needs: ordering, replay, and decoupling. The three layers only work together. Data streaming provides current, governed data in motion. Process intelligence and workflow orchestration turn that into an understanding of how the organization runs and coordinate what happens next. Trusted agentic AI acts on both.

AI agents make the dependency concrete. They do not just read data, they act on it, and an agent acting on stale data takes the wrong action automatically. For operational workloads the missing piece is context. An agent approving a payment or changing an order needs the current state of the business, not yesterday's snapshot. Fresh, governed context is also the best protection against hallucinated answers.

That dependency is why these landscapes belong together. The Data Integration Landscape 2026 maps how streaming connects to APIs and batch. The Process Intelligence Landscape 2026 covers the orchestration layer above it. The Trusted Agentic AI Landscape Q3 2026 maps the intelligence layer that consumes all of it.

The takeaway is narrower and harder than any vendor pitch. Apache Kafka is the dominant implementation and the protocol most vendors now speak. It is not the only implementation, and the analytical side runs on different engines entirely. **Every enterprise has to decide which workloads belong in the stream, and who controls where that stream runs.**

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 streaming, 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, quadrant placement, and market footprint reflect the author's assessment based on public information, vendor announcements, and two decades of enterprise architecture work with data streaming and data platforms. No vendor paid for inclusion or placement, and no vendor reviewed or approved its section before publication. Adoption, product, and regulatory details are drawn from public reporting and vendor statements where available; the market moves fast, and individual facts may change after publication. The author runs an advisory practice through Kai Waehner GmbH and has held roles at Talend, TIBCO, and Confluent. 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.

WEBSITE	**kai-waehner.de**
LINKEDIN	**linkedin.com/in/kaiwaehner**
NEWSLETTER	**Subscribe for data streaming, data integration, process intelligence, and trusted agentic AI insights**

KEEP READING · STAY INFORMED

Decide which workloads belong in the stream, and who controls where that stream runs.

BLOG & NEWSLETTER

Regular updates from the industry

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

kai-waehner.de/news

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 data streaming and enterprise AI landscape.

LinkedIn | X (Twitter)

MORE READING

The Trinity of Modern Data Architecture

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

Read more