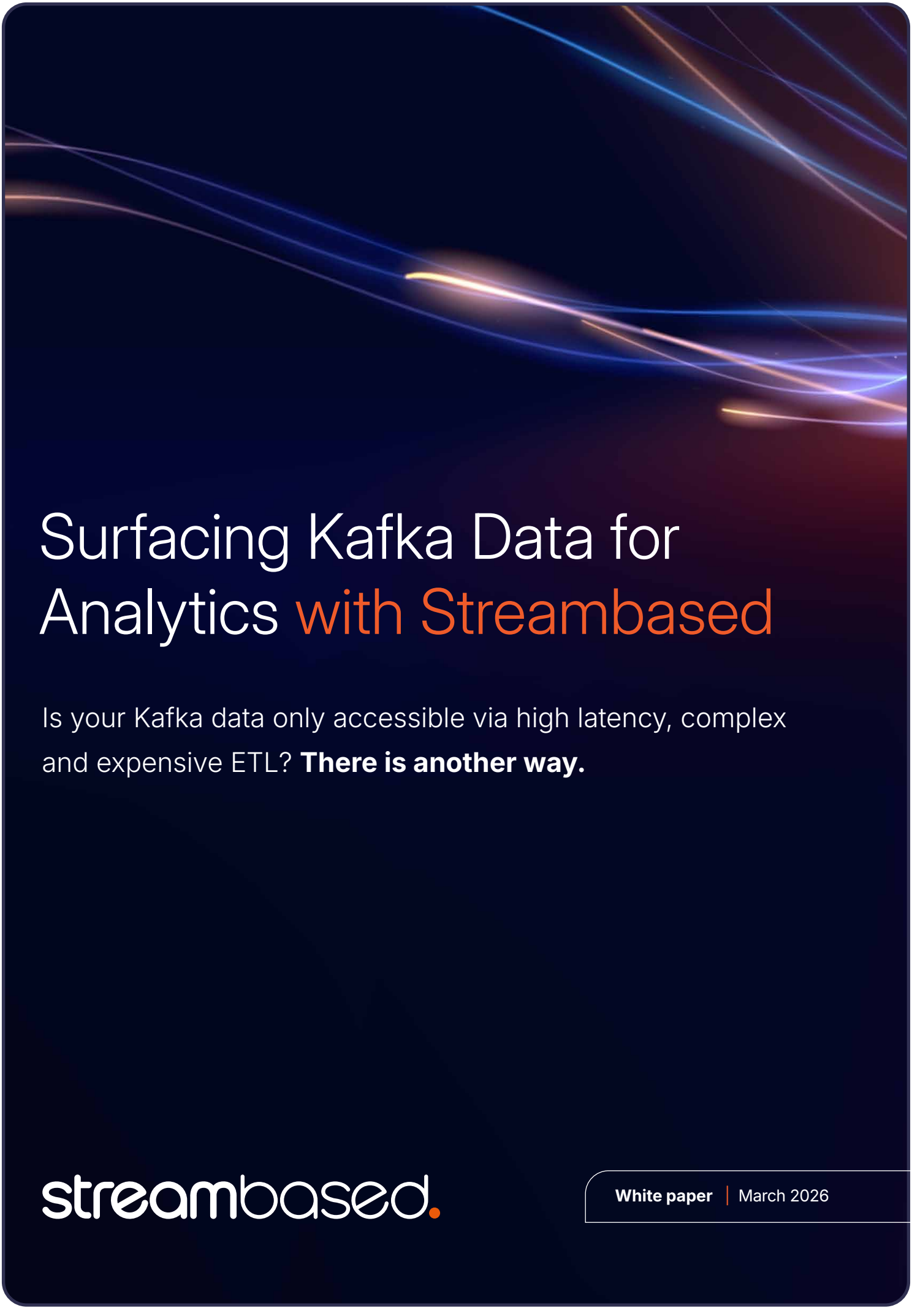


Surfacing Kafka Data for Analytics with Streambased

Is your Kafka data only accessible via high latency, complex and expensive ETL? **There is another way.**

streambased.

White paper | March 2026

Surfacing Kafka Data for Analytics with Streambased

Is your Kafka data only accessible via high-latency, complex and expensive ETL? There is another way.

Does the reporting and decision-making driven by your data lake feel a little stale? Real-time data increases accuracy, shortens time to value and enriches your data experience.

However, the path to real-time is not an easy one. Traditional ETL practices are outdated for real-time purposes and result in expensive and complex pipelines that feed specialist tools and techniques that must be adopted by analytical users.

Fortunately there's an alternative. Streambased takes an alternative approach to provide a zero-latency view over your real-time estate. With this view no complex pipelines are required and existing analytical tools seamlessly integrate and benefit from real-time opportunities.

In this whitepaper, we'll give an overview of Streambased technology, how it works and why it outperforms traditional data practices for real-time data.

Operational and Analytical System Convergence

The data lake is a collection of largely static data from disparate sources, brought together for analytical purposes. The event-streaming platform is a similar collection from disparate sources but this time the data is real-time in nature and the purpose is operational.

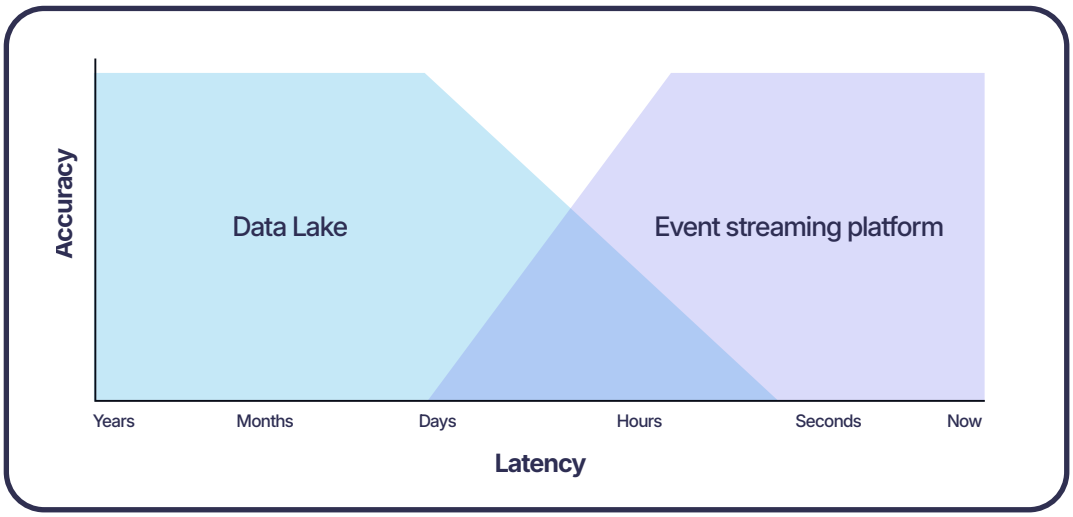
There are great similarities between both. In both systems data is relevant, valuable and actionable and both systems serve the purpose of joining datasets that you couldn't jointly query otherwise. However, one system tends to be focused on providing this data for human consumption (the lake) and the other for machine consumption.

Whilst similar in purpose, the way in which each class of system optimises for its role introduces a series of trade-offs that make each unsuitable for operation in the domain of the other. For instance, as the image below shows, lakes tend to be optimised for high volume, high latency access whereas event streaming systems are optimised for low latency and low volumes.

Practically, this leads to a very different experience for data users for each class of system:

For real-time:

- **Latency is low** - the real-time system holds the very latest data
- **Data is raw** - typically the real-time system is the point of ingestion, holding the very most raw data available
- **Experimentation is difficult** - most real-time systems lack support for the accepted tools for querying and experimenting with the data they contain (SQL)
- **Discoverability is low** - real-time data structures are typically not documents and published in the same way as analytical structures (Kafka topics are in no ones's data catalog)
- **Volumes are small** - real-time systems are designed to operate iteratively on newly arriving data and not on large historical datasets



For data lakes:

- **Latency is high** - it is normal for the data in a lake to be out of date by a day or more.
- **Setup time is high** - data lakes involve high volume stores and massive processing power. This in turn leads to high setup and maintenance cost and effort.
- **Data access is excellent** - Data lakes usually provide best-of-breed tools for efficient exploration that integrate with all major analytical applications
- **Volumes are high** - lakes are optimised to work with high volume historical data that can be years old

As can be seen above, for a complete data picture today, both systems must be employed. However, having 2 data platforms like this introduces its own set of issues:

- **Cost is high** - 2 systems mean duplicated data and compute, an expensive option
- **Complexity is high** - The path of data must flow from operational to analytical store and this process must be built, maintained and evolved if there are changes in either system
- **Consistency is low** - 2 systems for data storage mean 2 sources of truth, which is to be trusted when there is disparity between them?
- **Scope is low** - Data flow from operational to analytical must be explicitly enabled. In most organisations analysts must request a flow of this type to be built before it will be created but this suffers from the lack of discoverability mentioned above.

What does great look like?

The optimal solution would provide a seamless dataset that stretches all the way from the newest real-time data point (written 1 nanosecond ago) to the oldest point available (written 5 years ago). Users would be able to interact with this view using the tools and techniques they use today – whether that's standard SQL, Iceberg-compatible query engines such as Spark, Trino, Snowflake and Databricks or existing Kafka clients and consuming applications – without adopting any special procedure or additional technologies.

After all, the surest sign that this optimal solution has been achieved would be that its users don't even know it's there. This would mean:

- **Easy integration with existing workflows** - The new system should support usage by the tools and techniques currently in use for either lake or event streaming systems. Kafka and Apache Iceberg have practically emerged as the de-facto standards for each, so the optimal solution should work seamlessly across both, supporting real-time Kafka-consuming applications and Iceberg-compatible analytical SQL engines alike, without requiring either to be replaced.
- **The ability to address the entire data estate** - All data from the entire organization should be accessible
- **A single set of concepts** - Concepts from the event-streaming world (Topic/Consumer Group/Message) can be different to those from the data lake world (Schema/Table/Row). The ideal system should have a single set of concepts that covers both systems.
- **High performance across the entire data estate** - for analytical workloads the system should perform like a data lake and for operational ones it should perform like an event streaming system.
- **Easy discovery and experimentation** - Users should be able to explore the full data estate with the tools and techniques they are used to.
- **Consistent non functional requirements** - the same security/resource management model etc. for batch and realtime.

Streambased: Fast Analytics on Operational data

Streambased is an acceleration and integration layer built on top of Apache Kafka, the world's premier event streaming platform. Using Streambased you can leverage Kafka's best-of-breed operational capabilities and add best-of-breed analytical capabilities too.

Architecturally Streambased sits above existing data-serving platforms (Kafka/Iceberg) and presents datasets composed of a combination of the real-time and analytical data available from both. These datasets are served by Streambased in either Iceberg or Kafka format.

Streambased is not a data store and does not maintain a copy of the data. Instead, it creates composable, unified views directly above your existing infrastructure, bringing queries to where the data lives, rather than moving data to where the queries are, and presenting the result to analytical tools as a single, seamless surface.

In this way Streambased maintains a single source of truth for data (Kafka and Iceberg) whilst serving both operational and analytical needs.

Event Streaming Platform/ Database duality

The analytical world is driven by database concepts and Streambased provides interchangeable duality between these and concepts from the event streaming world. This means analysts used to working with databases will not even be aware they are using an event streaming platform to serve data when using Streambased.

In addition to this, Streambased brings a number of concepts that are common in the analytical world into the event streaming world to ensure a consistent experience, a sample of these can be seen below:

1. **Schema on read** - The ability to impose a message/table structure at query time is a hallmark of the data lake world and Streambased applies this in the event streaming realm.
2. **Consistent governance** - Streambased takes the same governance policies applied to the event streaming system and surfaces them to the analytical systems. This means policies like access control can be applied once and apply across the entire data estate.

Query everything, move nothing

At the heart of Streambased is a deceptively simple idea: rather than moving data to where your queries are, Streambased brings your queries to where the data already lives. It does this by creating composable, unified views that sit above your existing Kafka and Iceberg deployments – virtualising access across both at query time, with no physical data movement, no pre-staging and no ingestion jobs running in the background. To any standard analytical tool (Tableau, Power BI, Snowflake, Databricks, Trino....) the result looks like a single, native Iceberg table. The complexity beneath is entirely invisible.

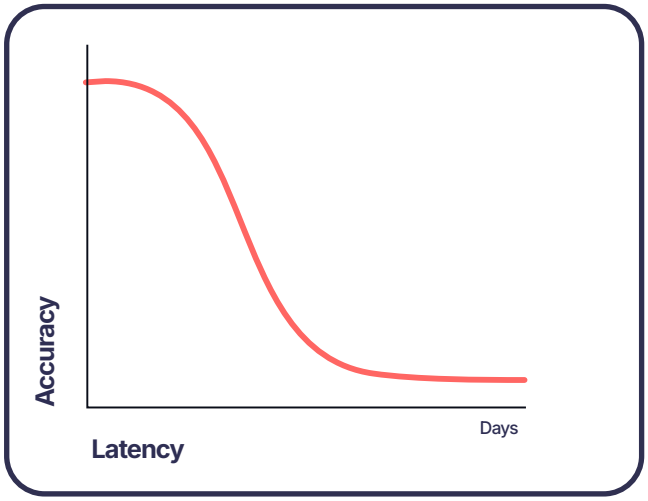
This is made possible by two complementary translation layers working in concert. The first, I.S.K. (Iceberg Service for Kafka), projects Kafka topics as Iceberg-compatible tables at query execution time, making new topics analytically queryable the moment they exist, with no need for any pipeline development. The second, K.S.I. (Kafka Service for Iceberg), works in the other direction, exposing Iceberg's deep historical archive back through the Kafka protocol so that streaming applications can reach years of cold history without the cost of extended Kafka retention. Together they resolve into a single logical offset timeline: one query can span data from the latest millisecond in Kafka all the way back to the earliest record in Iceberg.

Because Streambased is stateless and read-only by design, it deploys as an overlay on your existing infrastructure with no broker changes, no cutover and no disruption to current ETL jobs or downstream engines. Schema evolution in Kafka, such as field additions, deletions, type changes, is reflected in the Iceberg view immediately, with no replay or rewrite required. And because the same governance policies that protect your Kafka topics automatically extend to analytical queries, you get a single, consistent security model across your entire data estate without having to configure it twice.

Operational Unit	Analytical Unit	Description
Cluster	Database Schema	Each Operations cluster is represented as an analytical schema to indicate a separation of resources between them.
Topic	Table	As logical collections of data points both topics and tables are equivalent.
Message	Row	As logical data points messages and rows are equivalent

When to use Streambased: Business Use Cases

Up to now we have covered the drivers and benefits behind Streambased acceleration technology. Here we will marry these to real life use cases and principles when considering adopting Streambased technology.



Principle 1: Latency – Get data faster, make better decisions

As data latency increases, the accuracy of decisions made using that data decreases. A fraud model running on yesterday's transactions is already behind. A pricing algorithm drawing on last night's batch is reacting to a market that has moved on. The closer your data is to now, the more your decisions reflect reality, and the greater the competitive advantage that follows.

Use Cases:

Financial services: React to sudden market movements and validate trading strategies against live tick data before opportunities disappear.

Insurance: Have current policyholder and telematics information available at the moment underwriting decisions are made, not hours later.

Retail: Optimise pricing and inventory in response to demand shifts and emerging trends as they happen, not after overnight batch runs.

IoT: Correlate live sensor readings with historical failure patterns to predict and prevent device failures before they occur.

Fraud detection: Compare transactions against complete customer behavioural history during authorisation, before settlement completes.

Principle 2: Consistency – One copy of the truth, everywhere it's needed

When operational and analytical data live in separate systems, connected by ETL, you don't have one version of the truth. You have two, and they are never quite in sync. Reconciling them burns engineering time, erodes confidence in reporting and introduces risk wherever decisions depend on both systems agreeing. The only robust solution is to eliminate the duplication at source.

Streambased maintains a single copy of your data in Kafka and makes it available to every downstream application, operational and analytical alike, from that single source. There's no separate analytical store to drift out of alignment, no reconciliation process to run and no ambiguity about which system to trust when the numbers disagree.

It's worth noting that Streambased doesn't dictate how your organisation moves data; instead, it works with whatever transfer patterns you have in place. Most non-Streambased approaches force a compromise: ETL reduces availability by creating a lag between systems, Kafka Connect duplicates everything and inflates storage costs, and keeping everything in Kafka long-term becomes prohibitively expensive. Streambased sidesteps all of these trade-offs. For example, if your organisation runs a nightly transfer of data from Kafka to Iceberg, you won't lose access to that data during the transfer window, because Streambased maintains a continuous, consistent view regardless of your underlying transfer schedule.

Use Cases:

Financial services: Risk officers and trading desks draw from the same positions data, eliminating the intraday gap between actual exposure and reported exposure.

Retail: Inventory systems and demand forecasting tools operate from the same stock data, preventing the discrepancies between operational and analytical views that lead to stockouts and overbuying.

Healthcare: Clinical decision support and compliance reporting reference the same patient records, removing the risk of divergence between what was acted on and what was reported.

Telecoms: Telecoms: Network operations and customer experience analytics share a single view of real-time service data, so degradation is detected and reported consistently across teams.

Principle 3: Scope – More data, without more pipelines

In a pipeline-driven architecture, the only data you can analyse is the data someone has already decided to move. Over time, the gap between the data your organisation generates and the data your analysts can actually reach widens steadily.

Meanwhile, data engineers spend their time on plumbing instead of value. Every new data source requires a pipeline project. Every schema change breaks something downstream. Every team that wants access to a new dataset joins a queue. This complexity actively limits the scope of what your organisation can analyse, because only data that has been explicitly piped somewhere can be used.

With Streambased, any Kafka topic becomes immediately queryable the moment it exists. There are no pipeline projects to commission, no queues to join and no schema changes to negotiate before a new data source can be used. Adding a new data source takes minutes rather than months, and the energy your data team currently spends on maintenance can be redirected to the analysis that actually drives decisions. The result is that the full breadth of your operational data estate (telemetry, clickstream, transactional records and third-party feeds...) becomes analytically accessible as a matter of course, not as the outcome of months of engineering work.

Use Cases:

Financial services: Microtransactions and short-lived trades that ETL pipelines routinely exclude are fully visible, surfacing the early fraud signals and market shifts that aggregated batch data obscures.

Insurance: The complete range of vehicle telematics and smart-home sensor data becomes available for risk modelling, not just the fraction that has been explicitly piped to the analytical store.

Retail: Clickstream, loyalty programme activity and third-party market data sit alongside sales and inventory in a single queryable view, capturing the full customer journey rather than isolated transactional moments.

Manufacturing: Quality metrics, production line telemetry and supplier data are all immediately accessible together, enabling anomaly detection across the full operational picture rather than each silo in isolation.

Conclusion

Simplicity is the cornerstone of good architecture. In today's complex data architectures many hops, many intermediate stores and many disparate processors lead to a complex and fragile data estate that is underperforming relative to the data that is contained within it.

Streambased represents a leap forward to a simple single unified architecture that is ultra powerful for both operational and analytical workloads.

Furthermore Streambased can be "dropped in" to existing data architectures requiring no change in tooling or workflow for data users.

Get in touch

Contact hello@streambased.io

Book your demo today

Try it out

streambased.

streambased.io