kurrent

Reduce Your Kafka Infrastructure by 75% with Kurrent

While Apache Kafka delivers exceptional performance for high-throughput data transmission, many organizations find themselves needing more than basic messaging capabilities. Modern data architectures require sophisticated read models, materialized views, and purpose-built data products to serve diverse downstream applications. Since Kafka wasn't designed as a comprehensive data processing platform, organizations must often build intricate and expensive supplementary architecture to compensate for its limitations.

01 Required Capabilities

Time travel - materialize the state of a business entity at any point in time

Replay - re-execute event sequences to visualize processes or journeys

Complex querying - workflows requiring relational capabilities and/or random access patterns (e.g. multi-topic JOINS, complex WHERE clauses, arbitrary time windows)

System of record - authoritative source of truth for business entities and state transitions across distributed systems

02 Kafka Limitations

Retention - built to transmit and expire, not persist indefinitely for historical analysis

Meaning - no semantic understanding of the data; events are just bytes without business context

Sequence - event ordering within single partitions only, not globally across topics or the entire system

Correlation - doesn't capture the relationships amongst events or maintain causal chains

Recall - sequential access only, with no index or random access capabilities

Trust - no built-in cryptographic verification, tamper-evident audit trails, or end-to-end custody chain tracking for compliance requirements

03 Resulting Complexity

Within Kafka - 2-5x larger clusters

- Partition sprawl to achieve parallelism, key alignment and ordering guarantees
- Intermediary topics for stateful processing pipelines
- Embedded state stores (RocksDB) requiring dedicated infrastructure
- Schema registries for data governance
- Long retention windows consuming excessive storage

Around Kafka - outsized compute and operational overhead

- Heavy stream processing using KStreams, Flink, or custom processors
- Specialized 3rd party systems for indexing, querying, and materialized views
- Fragile data pipelines spanning multiple interconnected systems
- PKI infrastructure for cryptographic trust and artifact signing
- Centralized log aggregation and tamper-evident audit systems
- Event lineage tracking and custody chain verification tools

No single source of truth - derived data sets spread across multiple systems

- Audit / traceability challenges across fragmented infrastructure
- Complex rehydration processes requiring coordination between systems
- Difficult onboarding of new use cases due to architectural complexity

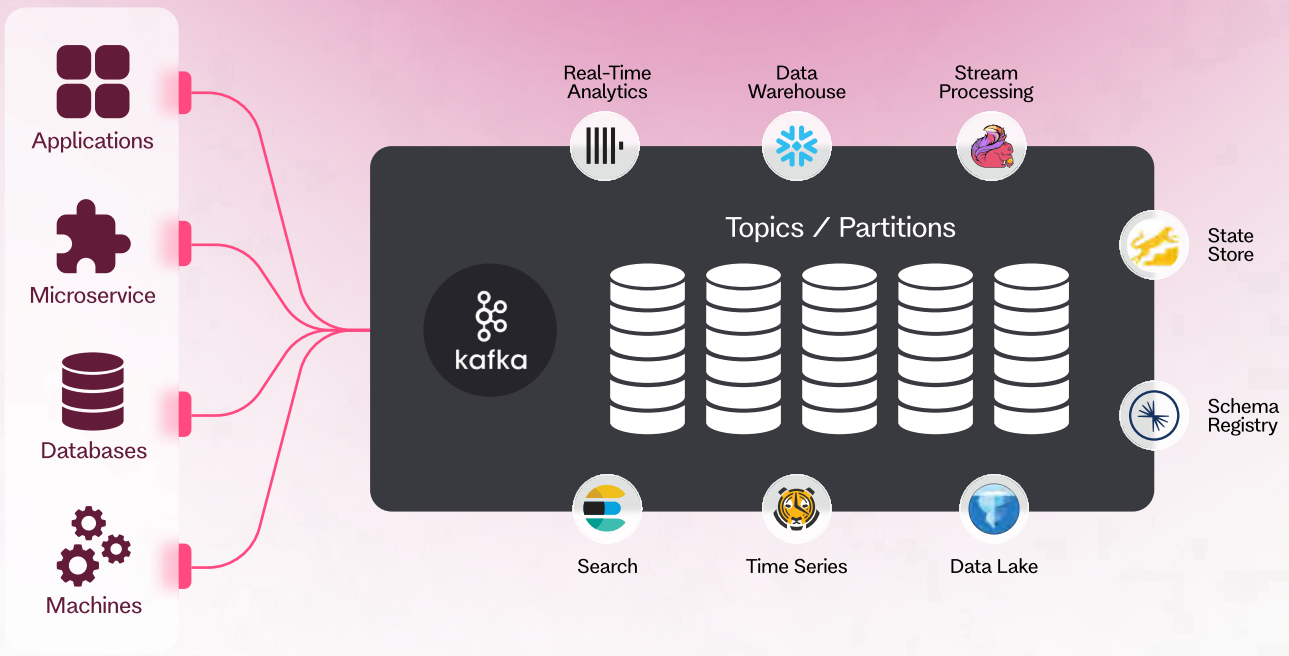

Figure 1: Compensating for Kafka limitations leads to complex, costly architectures.

Kurrent is an event-native database that stores, organizes and recalls events “lost” in Kafka

Kurrent operates as a specialized Kafka Consumer that stores events and applies five unique "superpowers" to organize them into a '4D memory' (e.g. what, when, how, why) for recall by downstream systems and applications.

Superpower 1

Model Events

Enrich events with business context, meaning, and structure as they arrive

Superpower 2

Sequence Events

Provide global ordering across all events for true chronological understanding

Superpower 3

Correlate Events

Maintain causal relationships amongst events to compose meaningful business processes

Superpower 4

Index Events

Enable efficient random access patterns without sequential scanning

Superpower 5

Trust

Provide cryptographic verification, immutable audit trails, and end-to-end custody chain tracking for enterprise compliance

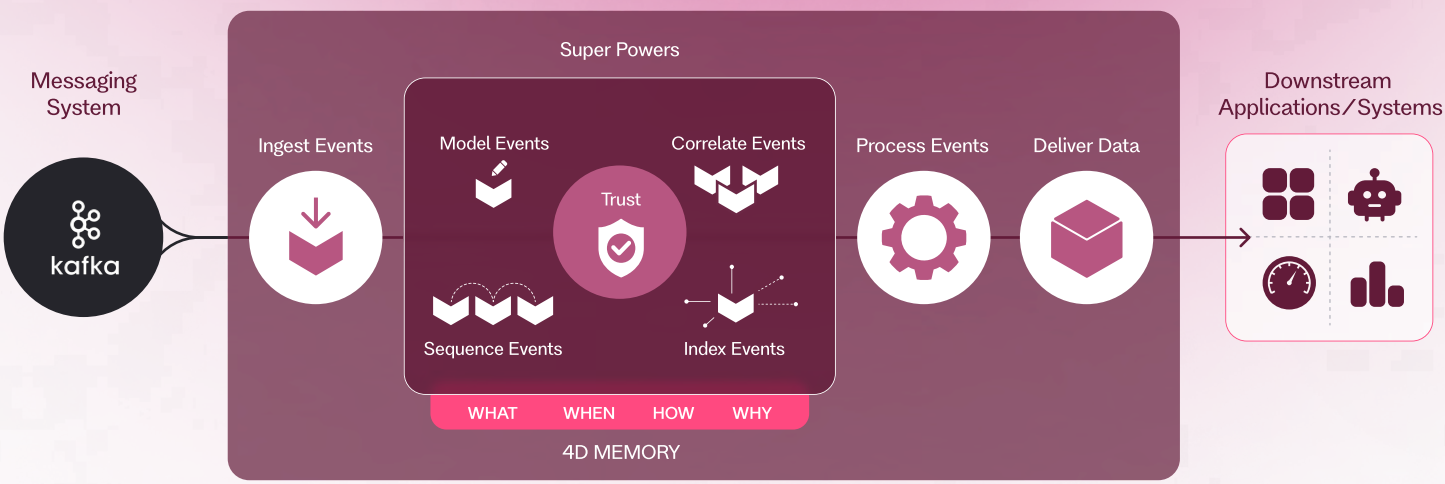

Figure 2: Kurrent applies its five superpowers to create a 4D memory

The Benefits of Kurrent

Kurrent's inherent capabilities deliver superior functionality, while enabling organizations to reduce the footprint of their Kafka ecosystem **by as much as 75%.**

New Analytical Capabilities for Events, Out of the Box

Time Travel - reconstruct the state of an entity at any point in time, and maintain a complete record of state transitions

Replay - re-execute event sequences to visualize business processes

Complex querying - create read models and materialized views across any dimension

Auditability - maintain a complete, globally ordered history of events with cryptographic integrity

Immediate Architecture Simplification

- ✓ No partition sprawl
- ✓ No intermediary topic hassles
- ✓ No stream processor complexity
- ✓ No embedded state stores
- ✓ Fewer 3rd party systems
- ✓ Create a single source of truth

Non-Disruptive to Implement

Kurrent integrates seamlessly with your existing Kafka infrastructure without requiring application refactoring or disruptive changes. Your current Kafka deployment continues handling high-speed data transmission exactly as before, while applications needing advanced capabilities like time travel, replay, or complex querying can simply connect to Kurrent instead.

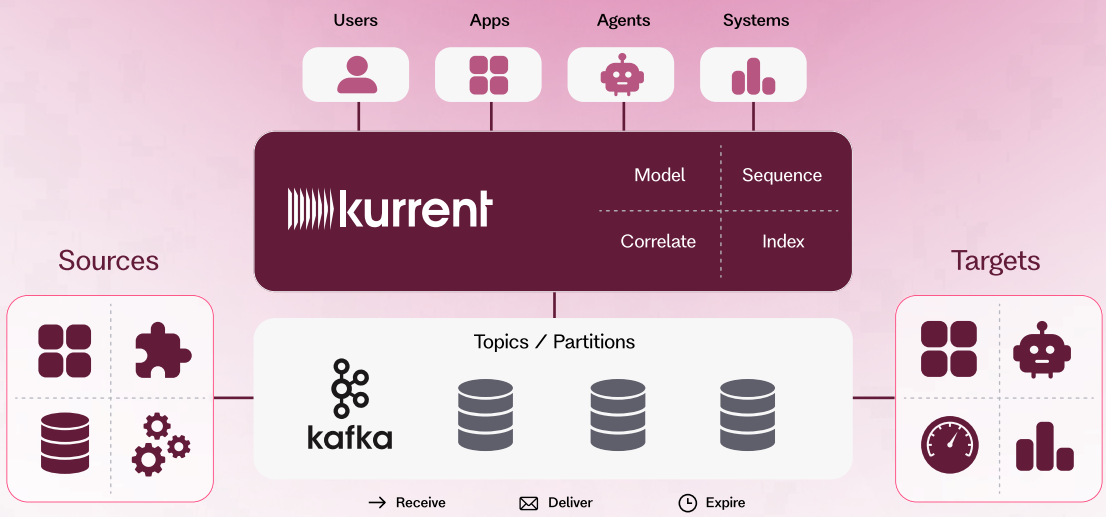

Figure 3: Kurrent creates a simpler, more functional architecture.

Recommended Starting Point:
Change Data Capture (CDC)

Traditional CDC approaches using Kafka often struggle with maintaining complete audit trails and enabling historical state reconstruction. Organizations typically capture database changes as individual events but lose the ability to understand how entities evolved over time or correlate related changes across different tables.

Kurrent transforms CDC by:

Complete State Transition History

Automatically maintains the full evolution of every database entity, enabling instant reconstruction of any record's state at any point in time

Cross-Table Correlation

Links related changes across multiple tables and schemas, providing a complete picture of how business transactions affect different parts of your data model

Simplified Compliance

Eliminates the need for complex audit table management by providing built-in time travel capabilities for regulatory reporting and data lineage requirements

Efficient Historical Queries

Enables point-in-time snapshots and temporal joins without the overhead of maintaining separate historical data stores or complex stream processing logic

Instead of managing multiple CDC streams, retention policies, and custom state reconstruction logic, organizations can simply connect their existing CDC pipeline to Kurrent and immediately gain sophisticated temporal capabilities across their entire database change history.

kurrent

Worth a Conversation?

If you are...

- ✓ Operating Kafka at-scale
- ✓ Needing time travel, replay or complex query capabilities
- ✓ Dealing with topic and/or partition sprawl
- ✓ Managing embedded state stores
- ✓ Using tools like Kstreams or Flink for stream processing
- ✓ Using CDC and need to maintain a complete record of state transitions

Let's discuss how Kurrent can simplify your infrastructure while unlocking new capabilities your business needs.

Contact sales@kurrent.io **to schedule a technical deep-dive and see Kurrent in action.**