NSTREAM

EBOOK

How to Scale Real-Time Streaming Data Applications Without Breaking the Bank

If your organization is like most, it's clamoring for tools and technologies that answer the essential questions:

What is the state of our business right now?

What immediate actions should we take in response?

Until recently, so-called real-time data processing techniques were too laggy to deliver these answers, particularly at the scale required. Any increase in the speed and volume of data, the number of streams processed, or the amount of data required to add crucial context would slow down output. The only way to compensate for this slowdown was to increase compute and data storage capacity. This strategy quickly became cost-prohibitive without actually delivering truly real-time results.

Enter modern streaming data applications, capable of processing data from multiple data streams and databases and delivering truly real-time results in a fully contextualized, human-readable format. In this ebook, we'll show you how a modern approach using stateful entities, streaming APIs, and real-time UIs can help you build applications that process real-time data at scale — minus the exorbitant costs.

Real-time streaming applications: A quick refresher

Harnessing value from the huge volumes of streaming data created by today's enterprises is critical for the creation of both industry-specific and vertical-agnostic applications, such as:

Real-time customer 360

to provide highly contextual, personalized offers and recommendations or proactive support based on a complete and instantaneous understanding of the customer and their needs.

Real-time marketplaces

such as ride-sharing or on-demand delivery services, using data from multiple sources to create a seamless customer experience by providing real-time inventory and dynamic pricing, as well as calculating accurate ETAs.

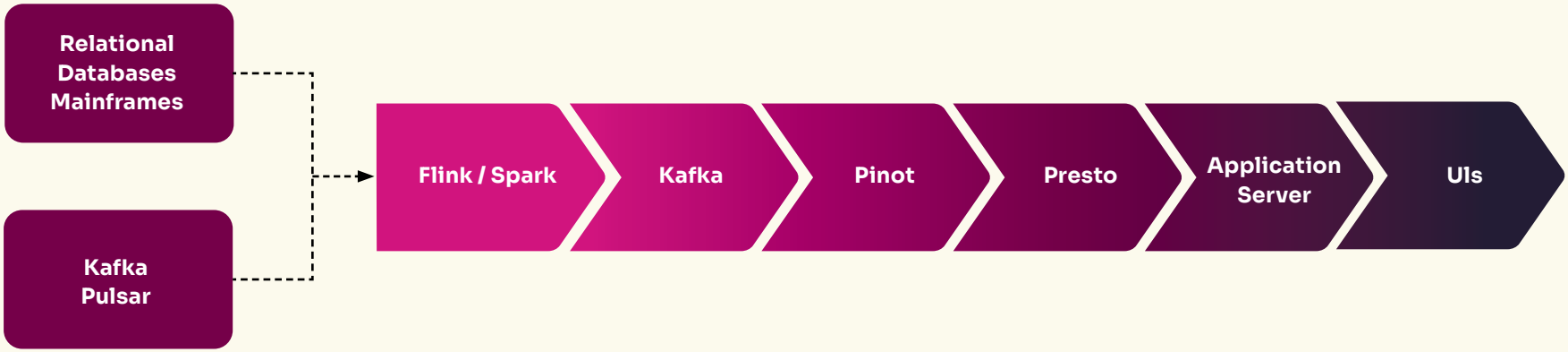
Real-time asset management

such as predictive maintenance, inventory management, or fleet tracking, with up-to-the-millisecond accuracy and complete business context.

We've learned a great deal about how to stream data at speed and scale over the years. But when it comes to processing that data, we're still using tools that were built in the 1990s to retrieve and exchange static documents from a database. Apply that legacy architecture to fast-moving, rapidly changing (i.e., streaming) data, and you immediately run into problems. Data floods in, and it is immediately stopped in its tracks as the application performs complex read-modify-writes to update the state of business entities to an external database.

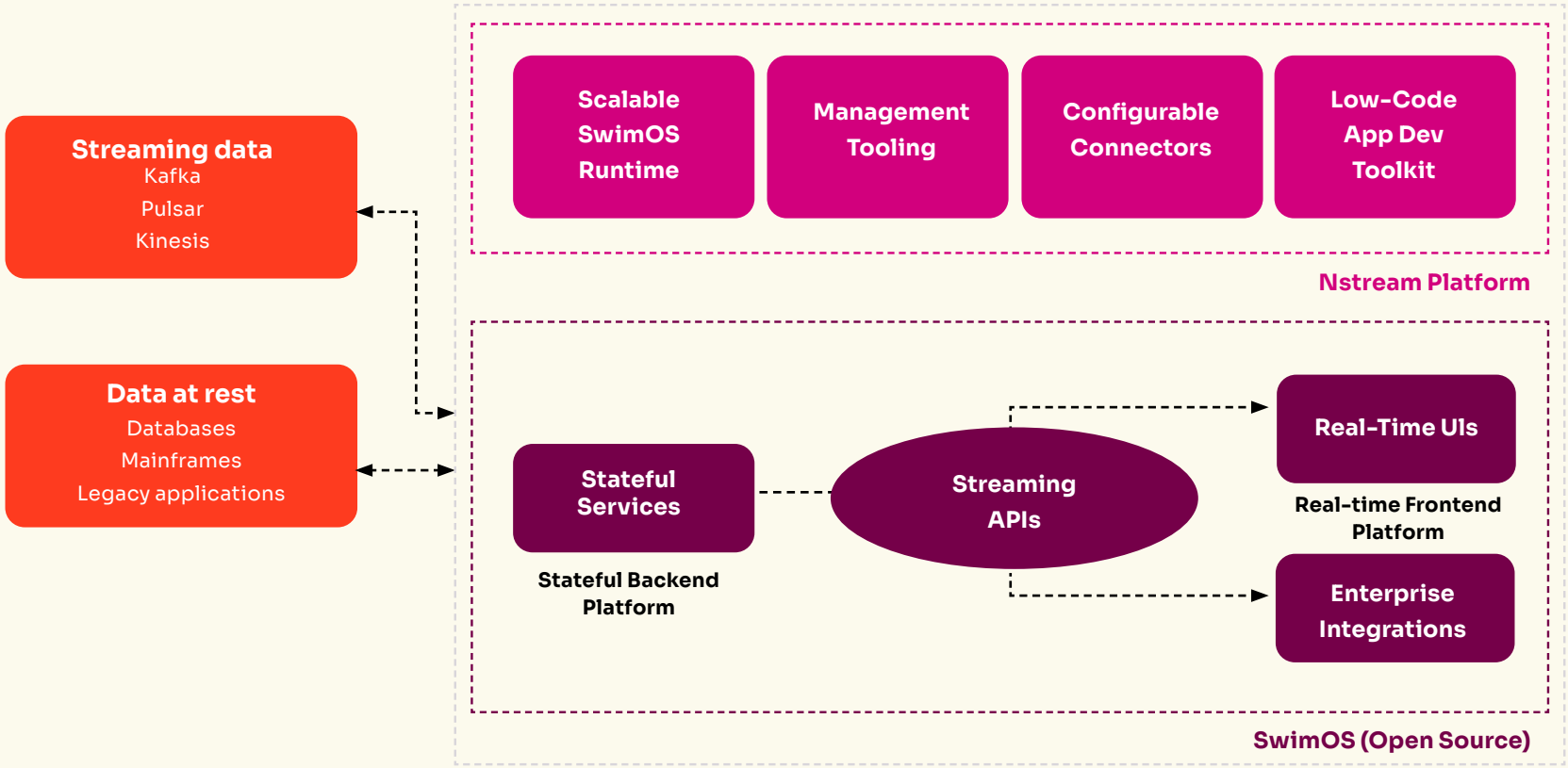
Each round trip in this store-then-process architecture — and each stream requiring processing — adds more lag until latency extends into minutes or hours and your data isn't really real-time anymore. This has real-world consequences, such as the inability to drive fine-tuned automation in response to unanticipated shifts. It's a bit like trying to drive a car while looking in the rearview mirror.

Typical streaming application architecture



Nstream's approach to building streaming data applications turns the store-then-process route taken by big tech and digital-first companies on its head. Instead, it takes a "process, then store if needed" approach that decouples latency from storage cost. Data doesn't stop streaming when it arrives at the application. Rather, the data pipeline now extends fully across the application layer. This greatly simplifies the data architecture by allowing the application to apply business logic, enriched by any stored data necessary, directly within the stream before passing on the results to the UI, automated action, or other enterprise applications. Nstream can accomplish this by taking advantage of stateful objects or services, streaming APIs, and real-time UIs.

A simplified streaming data application architecture



A stateful object or service continuously consumes updates to maintain its state. Each update computes a new state that automatically gets synced to listeners without explicit action or coordination. Application developers can leverage stateless objects to represent real-world objects subject to any business logic they care to impose. These can include just about anything: cell towers, prospects, customers, physical assets, IoT, checking accounts, and more. Instead of database requests and queries, stateful objects subscribe to updates or have them pushed directly during ingestion whenever information for the underlying entity comes in or changes. This results in full-context business logic, incorporating the state of the entity itself as well as the states of all other surrounding entities.

A streaming API moves data the moment it is available and keeps moving it until the data has been sent to all subscribers. This decouples producers from consumers because the streaming API implementation handles the routing and can also enforce access controls. Using efficient routing, a stream's journey can satisfy multiple subscribers along the way, achieving massive efficiency gains. Thanks to streaming APIs, the business logic continually executes in a real-time state instead of responding to queries. Polling is no longer necessary, which removes inefficiencies. Users do not need to request data, as it is automatically being streamed to their UI or other endpoint as soon as it's available.

A real-time UI enables the business to visualize, prioritize, and act on alerts more quickly than previously possible. Using the same facilities that power stateful objects, true real-time UIs are connected via continuous streams that don't compound latency by pausing for queues and database results — even as users manipulate the data via the UI. A real-time UI needn't be an endpoint on its own, but can provide an oversight window into downstream automation or integration with other enterprise applications.

The result of this modern architecture is an application that is both scalable and cost-effective, as we'll see.

TIP

If you're not familiar with streaming application architecture and development, read this blog post for a thorough introduction.

7 ways streaming applications make large-scale, real-time data processing cost effective

An architecture built around stateful objects, streaming APIs, and real-time UIs offers tremendous advantages over the traditional approach. First and foremost, it is massively scalable to millions of objects. As there is no traffic to and from the database required, scale increases without also increasing latency. This means that your application can apply business logic to multiple firehoses of data at once and provide the necessary context with more static data held in a database. It can deliver this transformed data to a UI or other application to drive visualization or automation.

To picture this, imagine a real-world example: a map of the United States that aggregates and analyzes petabytes of streaming data showing the state of every cell tower, every device connected to them, and every interaction between them. This complex system can instantly score call quality, for example, and drive automation to resolve bottlenecks or other issues. Moreover, the business can drill down to gain insight at any level: state, city, neighborhood, or individual customer.

The ability to scale streaming data applications at network-level latency results in multiple cost-saving benefits that can lower total cost of ownership (TCO) more than 70%. Read on for seven of these cost-saving benefits.

70%

Cost-savings from the ability to scale streaming data applications at network-level latency

1. **Lowering infrastructure costs:** Replacing "brute force" infrastructure costs (network speed, data storage) with a much more efficient architecture of stateful objects and streaming APIs lets you seamlessly scale to millions of streams in parallel with minimal additional costs.

2. **Improving agility:** Real-time UIs give users a complete picture so they can make better decisions in real-time. Connecting streaming data applications to enterprise applications (ServiceNow, Twilio, etc.) or microservices to create an automated workflow allows even faster remediation — further reducing costs surrounding downtime and customer satisfaction.

3. **Augmenting your current technology stack:** You can leverage streaming data applications to extend the capabilities of streaming data processing frameworks (Flink, Kafka Streams, etc.), integrate with your existing enterprise BI tools (e.g., Tableau, Thoughtspot, Looker, etc.), and power UI frameworks (Angular, React). This connectivity allows your organization to squeeze even more value out of its current technology investments.

4. **Higher data quality:** Traditional extract, transform, and load (ETL) processes put a drag on data speed, but streaming data applications can clean up and transform data mid-stream. This provides greater accuracy at network-level latency, greatly reducing the inaccurate or inconsistent data across different systems that can lead to poorly informed decisions.

5. **Fewer data systems:** End-to-end streaming data applications reduce the need for multiple data systems and integrations. This lowers not only the complexity for the application team, but also the cost of managing these systems, which typically require hiring expensive Subject Matter Experts (SMEs) for each data system or paying every data system software vendor for managed service.

6. **Greater efficiency and productivity:** With existing data stacks, most of an organization's time is spent on integrating databases, message brokers, BI tools, and analytics tools, leaving less time to focus on integrating business logic, which is what's actually providing value. Leveraging a streaming data application development platform can shrink time to value from months down to a matter of weeks. Once deployed, the application itself can save users time with more actionable information as well as opportunities for automation.

7. **Improved collaboration:** Because streaming data applications have no effective limit on the number of data sources they can process in real-time, they can facilitate data sharing across traditional business silos. Sharing data between functions like sales, marketing, data analytics, and customer services — and also across business units — allows for better decision-making and a more holistic view of value to the customer and their experience, which can lead to reduced costs and new revenue opportunities.

Nstream makes building scalable, streaming data applications possible

Nstream is the first (and only) full-stack platform for developing , managing, and operating streaming data applications at scale. It is the fastest way to build full-stack streaming data applications that ensure full visibility, context, and automation of streaming data value through to the application layer.

Choose your deployment. Nstream can operate wherever your data resides — be it on the cloud, your own data centers, or a mixture of both with customizable templates for easy deployment.

Modernize application architecture. Nstream provides a simple mechanism to express stateful microservices while exposing easy-to-use streaming APIs that can terminate data streams at the UI or dependent process.

Accelerate time to value. Nstream simplifies the development process with configurable data connectors, pre-built application components, and out-of-the-box visualizations. Its low-code development environment empowers more employee stakeholders to become citizen developers for their needs.

Operate at scale. Nstream applications can work with hundreds of millions of stateful entities, computing billions of contextual KPIs from millions of events per second — all at network latency.

The Nstream platform has everything you need to build streaming applications end-to-end, from build and testing to production deployment. All this is possible at a fraction of the cost of store-then-process architectures.

A new age for streaming data applications has arrived

Organizations need the ability to process streaming data in real-time in a highly scalable, cost-effective way. Stateful entities, streaming APIs, and real-time UIs are the foundation of a new kind of streaming data application that does exactly that.

With Nstream's low-code approach, developers can leverage unique streaming APIs and stateful objects to build applications with business logic to gain more and better insights. As a result, users can make data-driven decisions within minutes to reduce operational costs, improve productivity and agility, enhance the customer experience, and outpace competitors — rapidly.

To learn more about Nstream, visit Nstream.io/why-nstream.