WHITE PAPER

StreamSets
A SOFTWARE AG COMPANY

10 Best Practices
For Modern Data Integration

www.streamsets.com

Introduction

Before big data and streaming data, data movement was simple. Once you built a pipeline, it operated consistently because data moved like trains on a track, from static, structured databases to data warehouses or back and forth between databases and applications.

Modern data — and modern data infrastructure — have upended the traditional approach to data movement. The data supply chain is constantly evolving with new systems, services, and apps. And more and more apps and services are moving to the cloud to take advantage of elasticity and scale, while some data needed for business insights remain trapped in legacy systems.

As a result, big data processing operations are more like a busy city traffic grid than that orderly railroad line. Your data movement system has to:

- Accommodate streaming, batch, micro-batch, and CDC processing
- Support structured, semi-structured, and unstructured sources
- Enable streaming into multiple applications and evolving destinations
- Handle schema and semantics changes without affecting downstream analysis
- Respond to change from sources and applications controlled by different departments and external entities
- Unlock data from even the most secure and hard-to-access legacy systems — like your mainframe

To move data at the speed of business and unlock the flexibility of modern data architectures, modern data integration systems must be architected to handle change with the ability to monitor and manage performance continually.

These 10 best practices will help you develop a data integration practice to support continuous deployment and operations for modern data integration.

StreamSets
A SOFTWARE AG COMPANY

1. Eliminate hand coding as much as possible

Writing custom code to ingest data from sources into your data stores is still common. While sometimes necessary because of use-case requirements, this practice doesn't scale. Custom code creates brittle data pipelines where minor changes to the data schema can cause the pipeline to drop data or fail altogether. Constant maintenance of custom-coded pipelines is a huge problem with data engineering talent in short supply and more line of business users (read: not coders) involved in the process today than ever. Also, since instrumentation has to be explicitly designed — and often isn't — dataflows can become black boxes offering no visibility to pipeline health. Finally, sub-par coding can lead to tighter coupling between components, making it difficult to upgrade your infrastructure which stifles organizational agility.

Today, modern data integration systems can create code-free plug-and-play connectivity between data source types, intermediate processing systems (such as Kafka and other message queues), and your data platforms and object stores. The benefits of such systems are flexibility instead of brittleness, visibility instead of opacity, and the ability to upgrade data processing components independently. If you're concerned about customization or extensibility, these tools usually augment their built-in connectors with support for powerful expression languages or the ability to plug in custom code in your choice of programming languages and complex SQL queries.

2.

Be intent driven (aka - minimize schema specification)

While it is a standard requirement in the traditional data world, developing full schema specifications before big data and cloud deployments leads to wasted engineering time and resources. Applications ingesting data often use only a few key fields for analysis. Additionally, data sources often have less control over schemas that change over time and require ongoing maintenance.

Rather than relying on full schema specification, data integration systems should be intent-driven. Intent-driven systems let you specify conditions for and transformations on only those fields that matter for downstream analysis. This minimalist approach reduces the work and time required to develop and implement pipelines. It also makes dataflows more reliable, robust, and easy to diagnose.

3.

Plan for both: streaming and batch

Despite all the commotion about streaming analytics, enterprise data still lives in a batch-oriented world based on applications and source databases developed over the past 30 years. While you're building for cybersecurity, IoT, and other modern applications that capitalize on streaming data, you must also account for the fact that this data often needs to be joined with or analyzed against historical and other batch sources. For example, adding the decades of valuable transactional, customer, and sales data stored in mainframes to cloud analytics platforms can take analytics insights to the next level.

Rather than provisioning for a streaming-only framework, practical use cases demand that you incorporate streaming into the batch-driven data architecture while maintaining or improving the performance and reliability of the overall data operations.

4.

Sanitize raw data upon ingest

The original mantra for early Hadoop users was, "Store only immutable raw data in your data store." This approach spawned the "data swamp" metaphor from Gartner and others. Having data team members prune the data for each consumption activity is a common approach, but it's an inefficient use of resources. Moreover, storing raw inputs invariably leads to personal data and other sensitive information in your data platform, increasing security and compliance risks.

With modern data integration systems, you can and should sanitize your data upon ingest. Basic sanitization includes simple "row in, row out" transformations that normalize or standardize data formats. More advanced sanitization includes ranking, sorting, rolling averages, and other time-series computations, the results of which can be leveraged broadly by data scientists and business analysts.

Sanitizing data as close to the source as possible makes data scientists much more productive. They can focus on use-case-specific "data wrangling" rather than reinventing generic transformations that should be centralized and automated.

5.

Protect your data pipelines from unexpected shifts

Data drift — unexpected and undocumented changes to data structure, semantics, and infrastructures — is a fact of modern data architectures. And if the way you ingest data is dependent on how it's processed or consumed, you create a brittle pipeline; every time you change the way you store or consume data, you risk breaking the data flow.

Data drift erodes data fidelity, the reliability of your data operations, and ultimately, the productivity of your data scientists and engineers. It increases costs, delays time-to-analysis, and leads to poor decision-making based on polluted or incomplete data.

If your end goal is to operate continuously in the face of constant change and gain the freedom to innovate, then you need a data integration tool with dynamic data pipelines that keep running through any change, let you ingest more and more data without building more infrastructure, and allow different teams to innovate at their pace without any repercussions to the data engineering team.

6.

Architect for hybrid, cloud, and multi-cloud

Cloud managed services give users access to powerful tools that mask complexity and alleviate the burden of managing infrastructure and scaling. But data integration design patterns look fundamentally different when architected in the cloud. Data engineers often land raw data into object stores without knowing the analytics use case. The analytics value chain needs to be reimagined since recreating architectures exactly as they are on-premises is cost-prohibitive.

Legacy tools for data integration often lack the level of customization and interoperability to take full advantage of cloud services. Modern data integration systems should allow users to design pipelines and port them to the environment that makes the best sense for the business — shifting environments at will, with no pipeline rework. They can optimize for cost and efficiency in the cloud and still accomplish the goals of the data transformation. This can help reduce redundancy and provide insight into future cloud modernization.

7.

Ensure visibility into how systems are connected and data flows across the enterprise

Data constantly flows across the enterprise, with LOBs creating new integrations as part of their ongoing operations. But when data is used in multiple systems, how can you be sure it's being pulled from the best source? Can you trust data if you can't trace its provenance? And how can you maintain governance? In the rare case that manual documentation happens, it becomes obsolete almost immediately.

Organizations should endeavor to capture details of every aspect of the overall dataflow architecture, including automatically seeing when a new integration point is being created and if there is a more direct route for the data. And you should know where data comes from to help understand and explain outcomes, for example, in AI/ML models. Use a single management console for visibility into data connections and flows across a hybrid landscape, including volume and throughput of data and exactly what data is moving between components to reduce operational gaps and increase transparency, governance, and control.

8.

Don’t just count packages, inspect contents

To manage and solve for data drift, you must understand the actual content as it flows through your infrastructure. Otherwise, you leave yourself at risk of unannounced changes in data format or meaning. A significant change in data values might indicate a change in the real world that is interesting to the business —- or it might mean undetected data drift polluting your downstream analysis.

An additional benefit of data introspection is that it allows you to identify personal or otherwise sensitive data transiting your infrastructure. Many industries and geographies have strict requirements for personal data storage, such as the EU’s GDPR and California’s CCPA and CPRA.

Continually monitoring incoming data for patterns helps companies comply by providing real-time detection and tracking of any personal data they are collecting and storing.

9.

Take a DataOps approach to data movement

The DevOps sensibility of an agile workflow with tight linkages between those who design a system and those who run it is well-suited to data operations. Data pipelines must frequently change in a world with a continual evolution of data sources, consumption use cases, and data-processing systems.

Traditional data integration systems date back to when the waterfall development methodology was king, and tools from that era tend to focus almost exclusively on the design-time problem. This is also true of the early big data ingest developer frameworks such as Apache Sqoop and Apache Flume. Fortunately, modern dataflow tools provide an integrated development environment (IDE) for continual use through the evolving dataflow lifecycle. As modern dataflows continue to grow and become more complex, you must continually ensure that they function reliably and meet internal SLAs. This means adding tools that provide real-time visibility into the state of traffic flows with the ability to receive alerts and notifications, so you can act on issues that could violate contracts around data delivery, completeness, and integrity.

10.

Engineer for complex design patterns

Not only have data pipelines become complex, but they now span a range of deployment alternatives. Industry surveys confirm that enterprises expect to deploy data across multiple clouds while retaining on-premises data operations. Since each deployment option has its advantages, it is a mistake to expect a single approach to work now and forever. Realistically, business requirements will dictate an enterprise architecture combining many.

Regardless of where you are in your journey, it is best to assume a world where you have data stored in many different environments. Building an architecture based on complete "workload portability" means you can move data to the point of analysis based on the best price and performance characteristics for the job and do so with minimal friction. Also, you should assume that the constellation that describes your multi-cloud will change over time as cloud offerings and your business needs evolve.

Modernize Your Data Integration Practice

The good news is that as data has exploded in variety, volume, and velocity and its infrastructure continues to evolve, so do the solutions that keep your data moving smoothly. Built to eliminate data integration friction in complex hybrid and multi-cloud environments, StreamSets helps with all 10 of the best practices above — and then some. Learn more at streamsets.com.

About StreamSets

StreamSets, a Software AG company, eliminates data integration friction in complex hybrid and multi-cloud environments to keep pace with need-it-now business data demands. Our platform lets data teams unlock data—without ceding control—to enable a data-driven enterprise. Resilient and repeatable pipelines deliver analytics-ready data that improve real-time decision-making and reduce the costs and risks associated with data flow across an organization. That's why the largest companies in the world trust StreamSets to power millions of data pipelines for modern analytics, data science, smart applications, and hybrid integration.

To learn more, visit **www.streamsets.com** and follow us on **LinkedIn**.