Spotfire

Back to glossary

What is data streaming?

Data streaming is the process of transmitting a continuous flow of data (also known as streams) typically fed into stream processing software to derive valuable insights. A data stream consists of a series of data elements ordered in time. The data represents an “event” or a change in state that has occurred in the business and is useful for the business to know about and analyze, often in real-time. Some examples of data streams include sensor data, activity logs from web browsers, and financial transaction logs. A data stream can be visualized as an endless conveyor belt, carrying data elements and continuously feeding them into a data processor.

With the growth of the Internet of Things (IoT) and customer expectations, the significance of data streaming and stream processing has increased. Personal health monitors and home security systems are two examples of data streaming sources. A home security system includes multiple motion sensors to monitor different areas of the house. These sensors generate a stream of data transmitted continuously to a processing infrastructure that monitors any unexpected activity either in real-time, or saves the data to analyze for harder to detect patterns later. Health monitors are another example of data streaming sources including heartbeat, blood pressure, or oxygen monitors. These devices continuously generate data. Timely analysis of

this data is essential, as the safety of the person might depend on it.

What are the general characteristics of Visualize brilliance in action.

Sign-up for a free trial

Contact us

Time sensitive

Each element in a data stream carries a time stamp. The data streams are time sensitive and lose significance after a certain time. For example, the data from a home security system that indicates a suspicious movement should be analyzed and addressed within a short time period to remain relevant.

Continuous

There is no beginning or end to streaming data. Data streams are continuous and happen in real-time, but they aren't always acted upon in the moment, depending on system requirements.

Heterogeneous

The stream data often originates from thousands of different sources that can be geographically distant. Due to the disparity in the sources, the stream data might be a mix of different formats.

Imperfect

Due to the variety of their sources and different data transmission mechanisms, a data stream may have missing or damaged data elements. Also, the data elements in a stream might arrive out of order.

Volatile and unrepeatable

As data streaming happens in real-time, repeated transmission of a stream is quite difficult. While there are provisions for retransmission, the new data may not be the same as the last one. This makes the data streams highly volatile. However, many modern systems keep a record of their data streams so even if you couldn't access it at the moment, you can still analyze it later.

What is the significance of data streaming for business?

Data in the format of streams are highly significant in today's world. Numerous IoT devices and internet users generate huge volumes of continuous, real-time data every second. Processing this data in real-time is both a challenge and an opportunity for organizations.

The changing nature of data

Traditionally, organizations collect data over time, store it in data warehouses, and process them in batches. This saves scarce computing power. In recent years, data structure and processing technologies have greatly changed. IoT has introduced a wide range of sensors that generate stream data. Credit cards and online financial transactions also generate real-time data that needs to be analyzed and verified. Web browsers generate online transactions and activity logs. Data streaming and stream processing are essential to handle these types of data.

Large volumes of data

The amount of data that is generated every second is too huge to store in any data warehouse. Therefore, stream data is often assessed in the moment to determine if it is a key piece of real-time data or not necessary. As a result, systems can stream data and analyze it immediately to decide what gets stored and what does not — helping organizations reduce data loss, data storage, and save on infrastructure costs.

Examples of data streaming

Internet of Things: IoT includes a huge number of devices that collect data using sensors and transmit them in real-time to a data processor. IoT data generates stream data. Wearable health monitors like watches, home security systems, traffic monitoring systems, biometric scanners, connected home appliances, cybersecurity, and privacy systems generate and stream data in real-time.

Real-time stock market monitors: Real-time finance data is often transmitted in a stream format. Processing and analyzing financial data (like stock prices and market trends) helps organizations make crucial decisions fast.

Activity and transaction logs: The internet is also a major source of real-time stream data. When people visit websites or click on links, web browsers generate activity logs. Online financial transactions, like credit card purchases, also generate time-critical data that can be streamed and processed for real-time actions.

Process monitors: Every company generates billions of data points from their internal systems. By streaming this data and processing in real-time, businesses are able to monitor the system health and act

before things escalate. For example, manufacturing companies often have devices to monitor the health of the assembly line and to detect faults to assess risk in production. These devices can also stream time-critical data to monitor outages and even prevent them.

What is stream processing? How does it work?

In order to process streaming or live data, you need a process that is quite different from traditional batch processing. A stream processor collects, analyzes, and visualizes a continuous flow of data. And, of course, to process, you need to start with data streaming. Data streaming is at the beginning of stream processing. Stream processing is used to take in the data streams and derive insights from them, often in real-time. Due to the unique nature of the streaming data, a stream processor needs to meet the following requirements:

Low latency

A stream processor should work quickly on continuous streams of data. Processing speed is a primary concern due to two reasons. One, the data comes in as a continuous stream, and if the processor is slow and misses data, it cannot go back. Secondly, streaming data loses its relevance in a short time. Any processing delay causes deterioration of value in the data.

Scalability

Streaming data doesn't always have the same volume. For example, sensors may often generate low volumes of data, but occasionally, there might be a spike in the data. Since the volume of data is unpredictable, the processor should scale up to handle large volumes of data if required.

Availability

A stream processor cannot afford long downtimes. The stream data is continuous and arrives in real-time. A processor must be fault-tolerant, meaning it should be able to continue to function even if some of its components fail. A stream processor should also be able to collect, process, and immediately pass the insights to an upper layer for presentation.

What are the major components of a stream processor?

In general, there are two use cases in stream processing:

Datastream management

In datastream management, the objective of the stream processing is to create a summary of the incoming data or to build models. For example, from a continuous stream of facial data, a stream processor might be able to create a list of facial features. Another example of this use case is the internet activity logs. From the constant stream of user click data, the stream processor tries to calculate the user's preferences and tastes.

Complex event processing

Complex event processing is the use case that applies to most IoT data streams. In this use case, the data stream consists of event streams. The job of the stream processor is to extract significant events, derive meaningful insights, and quickly pass the information to a higher layer so that prompt action can be taken in real-time.

Some of the stream processors handle only one of the above use cases, while some advanced processors handle both. Irrespective of the use case, an end-to-end architecture of the stream processor should have the following functionalities:

Data generation

The data generation system denotes the various sources of raw data — like sensors, transaction monitors, and web browsers. They continuously produce data for the stream processing system to consume.

Data collection and aggregation

Each of the above data generation sources is associated with a client, which receives data from the source. These are known as source clients. An aggregator collates the data from several source clients, sending the data in motion to a centralized data buffer.

Messaging buffering

The message buffers take the stream data from an aggregation agent and store them temporarily before passing into a logic processor. There are two main types of message buffers: topic-based and queue-based. In the topic-based buffers, the incoming data is stored in the form of records called topics. One or more data producers can contribute to a topic. The queue-based message buffer is more of a point-to-point buffering system, reading from a single producer and delivering to a single data consumer.

Message broker

The data collection, aggregation, and message buffering systems

together form a message broker system. The functionality of the message broker is to aggregate the stream data from various sources, format it, and pass it on to a continuous logic processing system.

Continuous logic processing

This is the core of the stream processing architecture. The continuous logic processing subsystem runs various predefined queries on the incoming data streams to derive useful insights. The queries may be as simple as ones stored in an XML file. These queries are continuously run on the incoming data. This subsystem may define a declarative command language for the users to easily create these queries. The continuous logic processing system often runs on distributed machines for scalability and fault tolerance. Over the years, the logic processing system has evolved to support dynamic modification of queries and programming APIs for easier querying.

Storage and presentation

These are two supporting systems in stream processing. The storage system keeps a summary of the input data stream and can be used for future references. It also stores the results of the queries that are run on the continuous data stream. The presentation system is used for visualizing the data to the consumers. The presentation system may include a higher level of analytical system or alerts to the end-users.

Stream processing Vs. batch processing

Traditionally, organizations collect data from various sources — like sales records, customer interactions, and reviews — and store them in a data warehouse. This data is then processed in batches in a data analytics system. Batch processing often works on historical data,

which is not time critical. While historical data is important, it doesn't provide the real-time insights that many organizations strive for today. Stream processing makes insights on real-time data available and needs to be processed quickly.

With batch processing, there is a flexibility to store and process the data at your convenience, while stream processing requires real time or near real-time action. Batch processing handles data that isn't time critical and doesn't need to be as fast as stream processing. Compared to stream processing, batch processing often requires a bigger infrastructure to store data while it is waiting to be analyzed. However, most organizations need a combination of stream and batch processing to be successful in today's market.

Benefits of data streaming and processing

Stream processing and high returns

Organizations can derive immense value from data in general. Real-time stream processing techniques help organizations gain an advantage by analyzing time-sensitive data so they can react and respond quickly to potential issues. For example, stream analysis helps financial organizations monitor real-time stock prices and make time-critical decisions. It helps them be informed about real-time market trends. Robust visualization systems, along with a real-time stream processing infrastructure, allow organizations to improve their response time to crucial events.

Reduce infrastructure cost

In traditional data processing, data is often stored in huge volumes in data warehouses. The cost of these storage systems and hardware are

often a burden to the organizations. With stream processing, data isn't stored in huge volumes, so processing systems have lesser hardware costs.

Reduce preventable losses

Real-time data streams enable organizations to monitor their business ecosystem continuously. They keep organizations informed about possible security breaches, manufacturing issues, customer dissatisfaction, financial meltdowns, or an imminent social image disruption. With continuous data streaming and processing, organizations can avoid such preventable issues.

Increase competitiveness and customer satisfaction

With real-time data processing, organizations can proactively solve possible issues before they materialize. This gives them time and edge over competitors. Data streaming and processing also increases customer satisfaction as customer issues can be addressed in real-time. With continuous, real-time data processing, there is no delay caused by data sitting in the warehouses waiting to be processed.

Challenges for data streaming and processing

Data streaming and processing systems deal with highly volatile, real-time, and continuous data. The stream data is often heterogeneous and incomplete. The very nature of the stream data poses many challenges to data streaming and processing.

Data volume and diversity

Data streaming deals with huge volumes of continuous, real-time data. Data loss and damaged data packets are common challenges in data streaming. Stream data is often heterogeneous, originating from diverse geographical locations and applications. It is a challenge for data streaming and processing applications to handle this data due to its very nature.

Timeliness

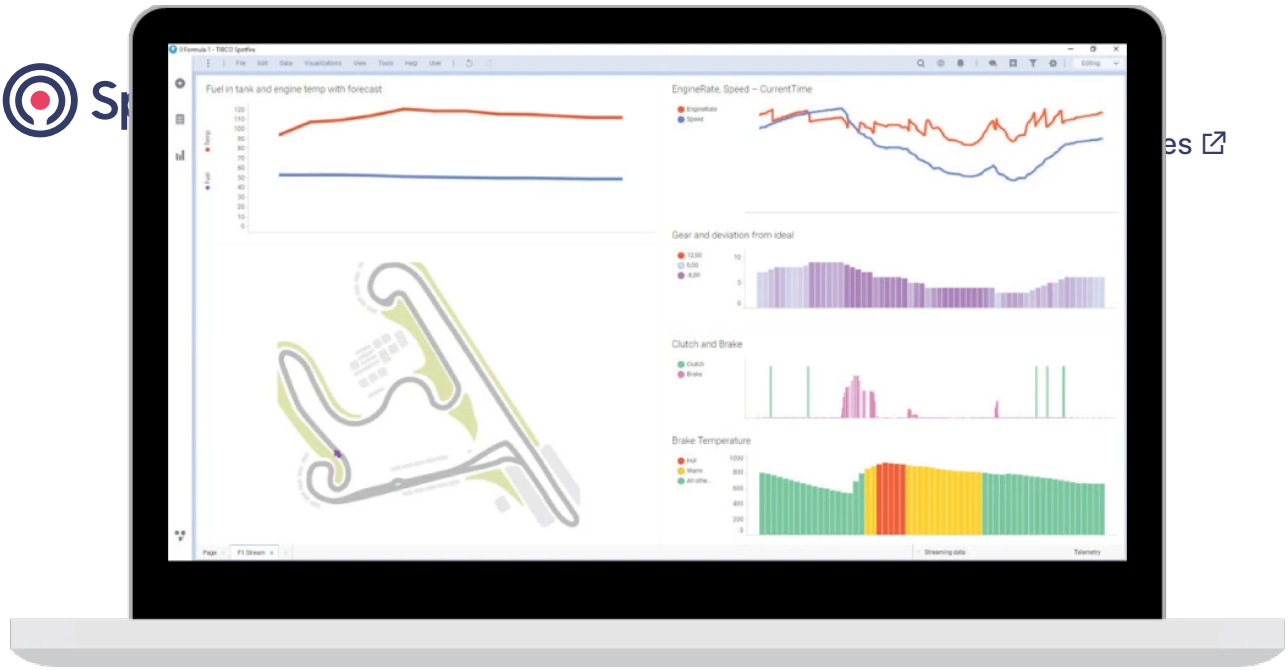
The relevance of stream data diminishes over time. Data streaming and processing system should be fast enough to analyze the data while it's still relevant. This time-critical nature of the stream data demands a high performing, fault-tolerant system.

Elasticity

Stream data volume is increasing every day. To maintain a certain level of quality of service, the stream processing systems need to adapt to the load dynamically. Stream data sources may not always transmit high volumes of data. In such cases, the processing systems must only use minimum resources. When the demand increases, the system should dynamically allocate more resources. This need for elasticity is another challenge from stream processing systems.

Fault tolerance

Stream processing happens continuously in real-time. The streamed data cannot be repeated or perfectly retransmitted. As a result, stream processing systems cannot afford downtime. Contrary to traditional batch processing systems, there is not much time between data collection and processing. Systems must be available all the time and should function all the time. If any part of the system falters, it should not affect the rest of the processing system.



© 2026 Cloud Software Group, Inc. All rights reserved.