Paper SAS1941-2015

# Real Time—Is That Your Final Decision?

Diane Hatcher, SAS Institute Inc.

## ABSTRACT

Streaming data is becoming more and more prevalent. Everything is generating data now—social media, machine sensors, and the Internet of Things. And you need to decide what to do with that data right now. And "right now" could mean 10,000 times or more per second. SAS® Event Stream Processing provides an infrastructure for capturing streaming data and processing it on the fly—including applying analytics and deciding what to do with that data. All in milliseconds. There are some basic tenets on how SAS provides this extremely high-throughput, low-latency technology to meet whatever streaming analytics your company might want to pursue.

## INTRODUCTION

The idea of processing events as they occur is not new. We already have ways of generating alerts when certain thresholds are crossed, or when specific data is available. Systems can track message queues or use web services for example, and take action when needed. But these systems are reactive in nature. They are focused on the consumption of previously existing information such as queues, listening for a response, or awaiting instruction. An event in this context is encapsulated as the result of an action or set of actions such as a purchase, a payment, a failure, or some other trigger.

Technology has advanced to a state where event data is constantly being generated with much more granular detail. An event can now be represented as something that happens within a system at a particular point in time, such as a click, a sensor reading, a tweet, or some other incremental, measurable activity within an ongoing process. Message queues and similar architectures have too much overhead to efficiently manage and track consumption of these granular events, or to deal with the velocity or volume of data that is in event streams.

Another distinguishing characteristic of event stream data is time. Time plays a critical role in defining context and is the single underlying construct across all types and formats of streaming data. One event by itself is typically not interesting. But when combined with other events that occurred before or just after it, a story emerges. For example, the story might be a possible machine failure, a case of fraud, a potential cyberattack, an opportunity to drive an existing sale, or a chance to provide a great customer experience.

This type of real-time analytics requires a specific infrastructure, one that can handle the need to continuously analyze an ever-changing set of data-driven events. The architecture must be able to capture events, assess them, make decisions about them, and share the outputs, all within the context of specific time windows. SAS® Event Stream Processing delivers the robust architecture needed to analyze streaming data so that organizations have immediate situational awareness and can proactively respond to changing conditions to drive more efficient operations and better customer interactions.

## WHAT ARE STREAMING ANALYTICS?

Streaming analytics is the process of manipulating data and applying analytics to events as they occur. These events are generated as a result of social networking activity, satellites, devices, networked machines, sensors or other systems at the edges of the Internet, or assets connecting an external customer or machine to an internal machine or to an organization. Continuous, real-time connectivity is essential to maximize the value of the use of this data. With streaming analytics, the goal is to analyze the data as close to the occurrence of the event as possible, before its value is lost due to information lag and before the volume of events overwhelms traditional analytics. This approach enables you to identify and examine patterns of interest as events occur, which enables immediate action on those events of interest as they happen.

Managing data in motion is different from managing data at rest. Event stream processing relies on the following principals: assessment, aggregation, correlation, and analytics to deal with data in motion.

- Assessment

  With the massive volumes of streaming data it is simply not practical to store it all. Much of what is generated is irrelevant for any action, analysis, or archive. Event stream processing can be used to standardize the incoming data, apply transformations, apply rules to determine if the data is relevant, and determine if any further downstream processing is needed. If not, the event can be discarded, without taking up additional processing bandwidth.

- Aggregation

  Event stream processing is used to continuously calculate metrics across sliding windows of time to understand real-time trends. This type of continuous aggregation would be difficult with traditional tools because of the time lag and I/O overhead associated with storing data on disk and then running a query against it.

- Correlation

  An individual event by itself is not interesting. It is a single data point, with no reference or context. But when you bring together and relate multiple events from a single stream or even multiple streams, then the operational context exists to assess events. Monitoring patterns and correlations (such as identifying that Event A was followed by B and then C within a given time window) provides more accurate understanding of a situation than does recognizing that Event B happened.

- Analysis

  One of the inherent characteristics of processing streaming data is gaining as much insight as possible while the data is still in motion. The ability to score or perform advanced analytics while events are happening ensures that operational processes are current with existing conditions. Adding “out of stream” analytics to events from data streams enables you to enrich the events and put the events in business context. This approach helps you to leverage the stream directly to improve organizational effectiveness and efficiency by driving additional revenue, improving a customer experience, or reducing costs and risk.

## HOW DOES SAS EVENT STREAM PROCESSING WORK?

### OVERVIEW

SAS Event Stream Processing includes a server, which provides the run-time environment for executing the event stream models against one or more input data streams. The run-time environment is built on C++. Both an interactive GUI and an optional XML interface are available for model development and manageability. With a publisher/subscriber interface to connect to streaming sources and output locations, SAS provides the most complete and integrated solution available in the market.

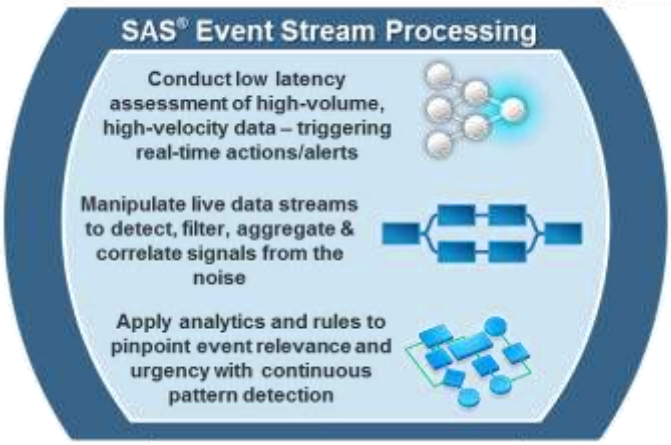


**Figure 1. SAS Event Stream Processing Core Elements**

There are three main elements to the architecture (as noted in Figure 1):

- Publishing interface for connecting to live data streams

  Connectors publish the event streams into source windows. Publish operations natively read event data from a specified source and inject that event data into a specific source window of a running event stream processor.

- SAS Event Stream Processing server

  The SAS Event Stream Processing server executes an event stream model that specifies how streaming data input from publishers is meaningful to output locations consumed by subscribers. The server supports continuous queries, which are defined in the model. A continuous query can be envisioned as a directed graph, which is a set of connected nodes (windows) that follow a direction down one or more parallel paths. These nodes represent the pattern definition, transformations, and analytics that are to be applied to the data, similar to a flow diagram.

  The server supports multiple languages to define expressions, patterns, and transformations. You can use embedded aggregate functions, or you can define your own procedural code using C++, SAS® DS2, or the DataFlux Expression Language. In addition, you can define temporal windows that capture streaming data over a period of time – static or rolling – and check for patterns or score an advanced analytic model based on the aggregate events that have been cached in the time window.

- Subscribing interface to share the results with other systems or information consumers

  Adapters and connectors subscribe to window event streams. Subscribe operations natively write output events from a window of a running event stream processor to the specified target.

Event stream processing engines with dedicated thread pools can be embedded within new or existing applications. SAS Event Stream Processing enables the continuous analysis of flowing data over periods of time where low latency and incremental results are important. SAS Event Stream Processing applications can analyze millions of events per second, with latencies in the milliseconds—even on small, commodity servers with 4-8 cores. Because of the lightweight architecture and thread pool management, it is easy to linearly scale the environment to run multiple SAS Event Stream Processing servers in a grid to distribute the load and extend in-memory resources.

SAS Event Stream Processing also offers high availability and provides this capability through a software-based approach for native failover. Native failover ensures that data streams are never lost because the servers can automatically and seamlessly switch over and reroute traffic to active servers when failures are detected, without clients even knowing this has occurred. SAS Event Stream Processing provides 1+N way failover with the use of message bus technology provided by Solace Systems, Tervela, or RabbitMQ.

## EVENT STREAM APPLICATION DEVELOPMENT

Conceptually, an event is something that happens at a determinable time that can be recorded as a collection of fields, which have a definable schema.

In designing an application, programmers must answer the following questions:

- What streams of data define an event? That is, how do the event streams need to be published into the application?
- What happens to that data? How are event streams transformed and modeled?
- What data streams out? What analysis conclusions are consumed by which subscribers and how do they need them?

The answers to these questions determine the structure of the SAS Event Stream Processing model. There are several parts to consider, as shown in Figure 2, below.

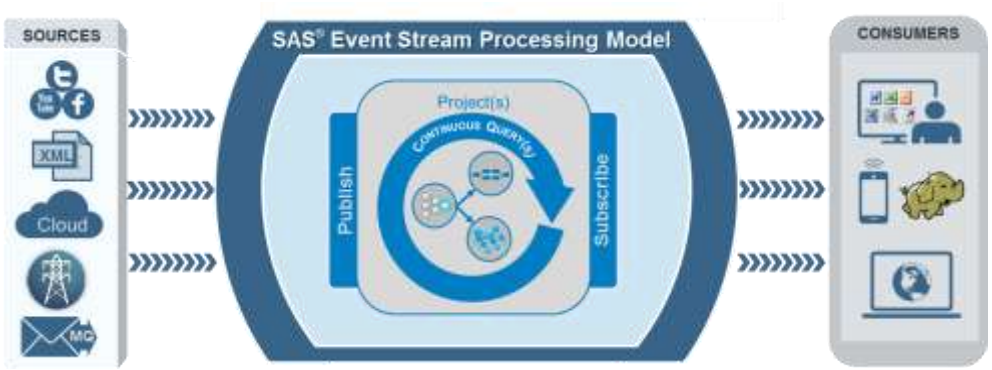


**Figure 2. SAS Event Stream Processing Application Architecture**

Each event stream application contains a single model definition. Within that definition, you can have one or more projects. Projects define the dedicated thread pools that are specified relative to size. Using a pool of threads in a project enables the event stream processing engine to use multiple in-memory processor cores for more efficient parallel processing. This gives the modeler the ability to optimize the performance of the environment across multiple projects.

Within each project, you can also have one or more continuous queries. Each continuous query consists of at least one source window and one or more derived windows. The source window defines the schema for the input stream. The derived windows determine the actions to take from the analysis. Derived windows can detect patterns in the data, transform the data, or perform computations based on the data. They can be connected to other derived windows.

Here are some examples of derived windows that you can use:

- **Aggregate:** computes aggregations of non-key fields.
- **Compute:** performs one-to-one transformation of input events to output events.
- **Copy:** defines the retention policy for data based on the parent window. The parent window can be the source window or a derived window. Retention can be defined in static or rolling time periods, or by number of events.
- **Counter:** counts the number of events streaming through your model and the rate they are being processed.
- **Filter:** defines which data is relevant to continue processing in-stream based on a Boolean filter function or expression.
- **Functional:** enables you to use different types of functions to manipulate or transform event data. When an event enters a functional window, the window looks for a function with a name that corresponds to each field in its schema.
- **Join:** joins two input windows. Supports both inner and outer joins.
- **Notification:** sends notifications through email (SMTP), text (SMS), or multimedia message (MMS).
- **Pattern:** defines connections between different patterns of interest, based on both time and calculation.
- **Procedural:** enables you to define custom functions to apply to input data.
- **Text Context:** enables the classification of terms from an unstructured string field.

### SAS® EVENT STREAM PROCESSING STUDIO

To facilitate the development and implementation of Event Stream Processing models, a studio environment has been developed, which provides a web-based, point-and-click interface that supports SAS Event Stream Processing model development and testing. SAS Event Stream Processing Studio enables you to visualize your SAS Event Stream Processing models in a process flow, as you add windows to your model (shown in Figure 3).

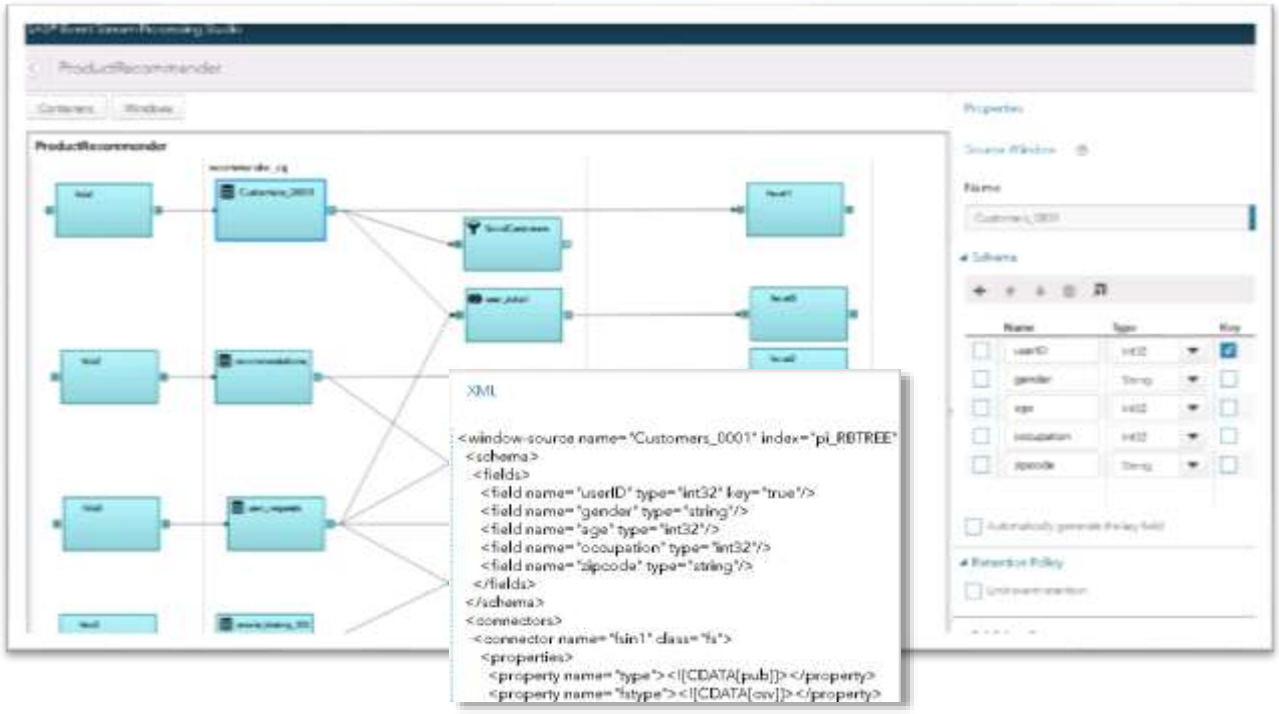


**Figure 3. Screenshot of SAS Event Stream Processing Studio.**

From the visual interface, you have access to many of the key windows, and as windows are added and joined to each other in the model flow, the appropriate XML code is automatically generated. You can publish the resulting XML code to the server for execution, and you can import existing SAS Event Stream Processing models in XML to create the process flow diagram.

## HOW DOES EVENT STREAM PROCESSING LEVERAGE SAS ANALYTICS?

One of the key advantages of SAS Event Stream Processing is its ability to leverage SAS analytics. The value of operating on streaming data increases significantly when you can interrogate the events in a business context. There are two key methods for doing this:

- enriching events with additional intelligence already gathered by your organization
- applying advanced analytics such as scoring models or other business rules to the streaming data

The results from such analysis can then be fed directly into operations or other data stores to drive new insights and models, which can then serve as new elements within SAS Event Stream Processing detection (as depicted in Figure 4 below).

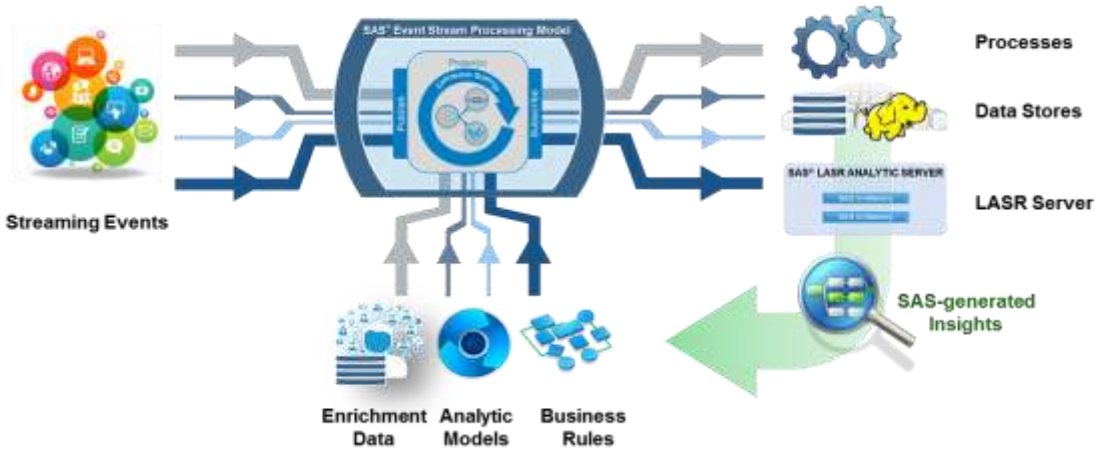


**Figure 4. Injecting Data Enrichment, Analytic Models, and Business Rules**

## ENRICHING EVENTS

While events in context provide insight over a specific time window, there might be times when the event is tied to a longer term process or relationship that is beyond the containment of a live event window. For example, if an event is being triggered by a specific customer activity, such as when a customer enters the retail outlet WiFi, it's possible that this is a customer who has some previous history with the store. As a result of that history, there might be additional intelligence about that customer. The history might include buying behavior, product preferences, or other segmentation characteristics.

SAS Event Stream Processing enables you to capture this pre-existing intelligence in the SAS Event Stream Processing model by using join events with existing data to enrich the live event data. Using connectors, you link existing data that is stored in a database, XML, CSV, JSON, or other sources, and load that data into your SAS Event Stream Processing project. That historical data then becomes available to join with streaming events, providing much richer context to the live event.

## APPLYING ADVANCED ANALYTICS

SAS Event Stream Processing provides the ability to execute SAS analytics in the stream using the procedural window. The procedural window enables you to specify one or more input data streams and the code to execute against each. This code is referred to as an input handler and can be written in C++, SAS DS2, or XML. The following code is an example of how to define a DS2 input handler that calculates the total cost of a stock transaction by multiplying stock price by number of shares (size):

```
input schema:
      "ID*:int32,symbol:string,size:int32,price:double"

output (procedural schema):
      "ID*:int32,symbol:string,size:int32,price:double,cost:double"

ds2_options cdump;
data esp.out;
      dcl double cost;
      method run();
            set esp.in;
            cost = price * size; /* compute the total cost */
      end;
enddata;
```

Although the example above is a simple calculation, it is possible to leverage any SAS DS2 code within the event stream. SAS DS2 code can be associated with predictive models, forecasts, data mining, or other advanced algorithms. Note however, that these advanced analytic models are developed against a larger set of data that includes history and is tested against validation samples. Thus, traditional algorithm development remains. These advanced analytics are developed outside of event streams using solutions such as SAS® Enterprise Miner™ and are simply deployed within the event stream model.

Moreover, such traditional, in-depth analysis can identify new patterns or measures that need to be monitored and thus encoded into the streaming analytics for detection. As such, there is a feedback loop from out-of-stream analytics to in-stream analytic scoring, so that models are enhanced over time as events evolve.

## CONCLUSION

Organizations are just beginning to take a deeper look at capturing data about streaming events. Being able to act as soon as events are generated elevates operational responsiveness and organizational effectiveness. The challenge is how to move beyond the limited value that simple data assessment provides. As context is applied to events in real time, the opportunity to act in a relevant and precise manner is vastly improved, as depicted in Figure 5. Those that take advantage of the power of event streams to immediately take the best action have a competitive advantage.

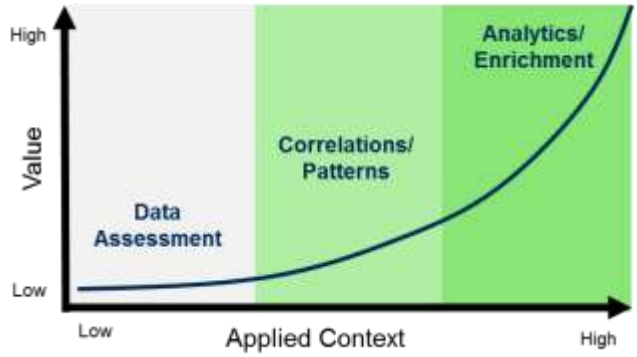


**Figure 5. Value Increases as Applied Context Increases**

SAS Event Stream Processing can help you realize the value of streaming data by providing the flexibility and performance to apply rich context to the events as they occur. SAS Event Stream Processing provides the full range of contextual needs to immediately act with confidence by supporting the following capabilities:

- integration with SAS Analytics
- visual interface to streamline development and implementation of SAS Event Stream Processing models
- extremely high throughput and resiliency
- highly flexible architecture to integrate with streaming data sources and output channels

## REFERENCES

Baulier, J., S. McCormick, and M. Dobeck. October 2014. SAS Institute white paper. “Making Meaningful Predictions in the Fast Lane: The Role of Event Stream Processing in Connected Vehicles.” http://www.sas.com/en_us/whitepapers/iia-making-meaningful-predictions-in-fast-lane-107420.html.

## ACKNOWLEDGMENTS

Special thanks to Fiona McNeill and Steve Sparano for helping with reviewing and editing this paper. Also, thanks to Michael Harvey for providing content.

## RECOMMENDED READING

- SAS White Paper. “Understanding Data Streams in IoT.” http://www.sas.com/en_us/whitepapers/understanding-data-streams-in-iot-107491.html. (Registration is free)

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Diane Hatcher
SAS Institute Inc.
919-531-0503
Diane.Hatcher@sas.com
http://www.sas.com/en_us/home.html