

Building a SAM Application

Date of Publish: 2018-11-15



Contents

Building an Application.....	3
Launch the Stream Builder UI.....	3
Add a New Stream Application.....	3
Add a Source.....	6
Connect Components.....	10
Join Multiple Streams.....	10
Filter Events in a Stream.....	11
Use Aggregate Functions over Windows.....	16
Deploying a Stream App.....	18
Configure Deployment Settings.....	18
Deploy the App.....	19

Building an Application

Prerequisites

- You have integrated SAM
- You have set up appropriate environments and service pools

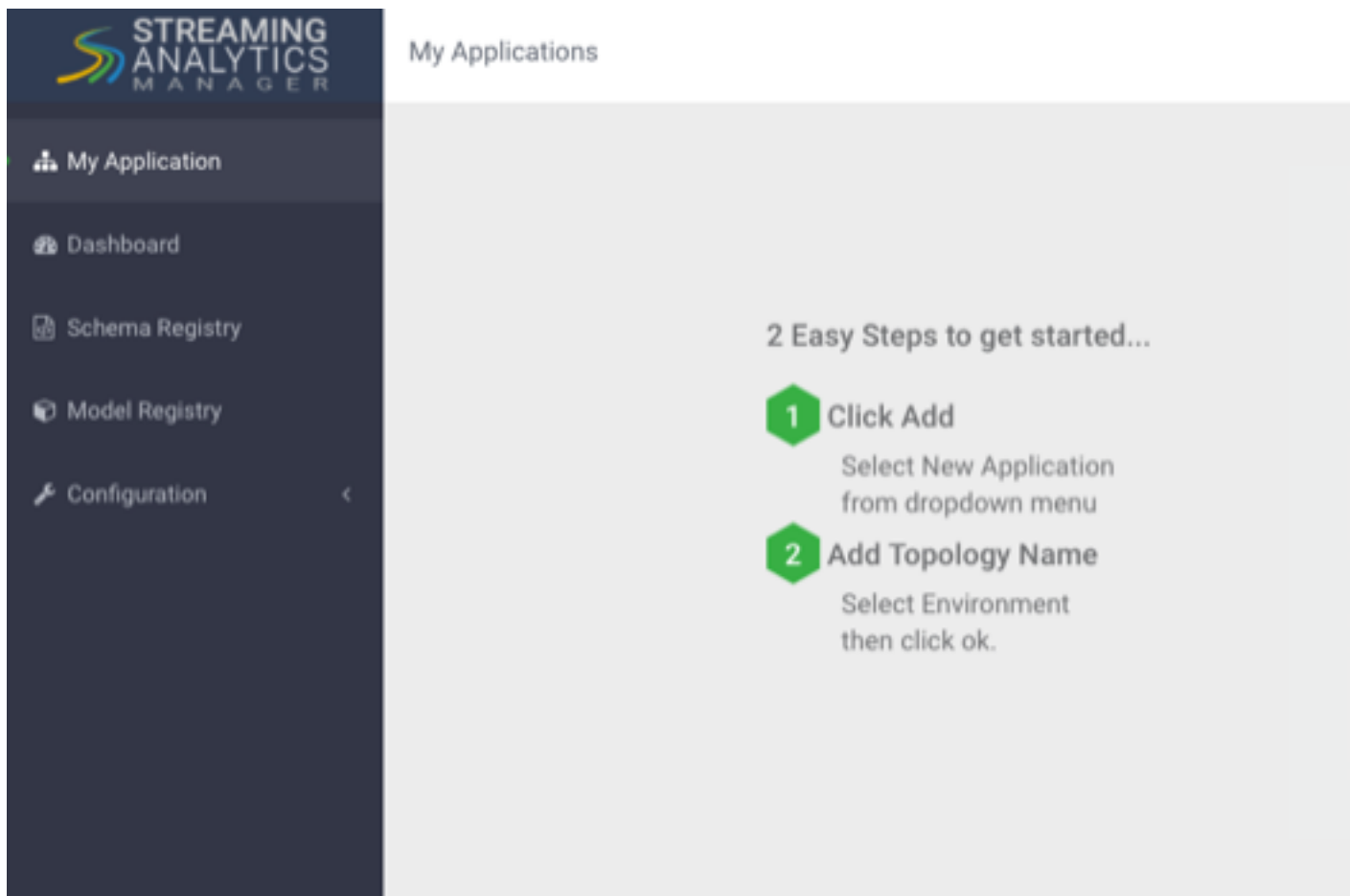
Launch the Stream Builder UI

Procedure

1. In Ambari, select **Streaming Analytics Manager** from the left-hand **Services** pane.
2. Under **Quick Links**, select **SAM UI**.

Results

The SAM Stream Builder UI displays. You can return at any time by clicking **My Applications** from the left-hand menu.



Add a New Stream Application

Procedure

1. Specify the name of the stream application and the environment you want to use.

**Note:**

The name of the stream app should not have any spaces.

The screenshot shows a dialog box titled "Add Stream" with a close button (X) in the top right corner. Below the title, there are two input fields. The first is labeled "NAME" and contains the text "Trucking-IOT-Stream-Analytics". The second is labeled "ENVIRONMENT" and has a dropdown menu with "Dev" selected. At the bottom right of the dialog, there are two buttons: "Cancel" and "Ok".

2. SAM displays the Stream Builder canvas. Builder components on the canvas palette are the building blocks you use to build stream apps. Refer to the *HDF Overview* for information about each component building block.

The screenshot displays the Hortonworks Streaming Analytics Manager interface. At the top, the text "My Applications" is followed by a box containing "Sample Application". To the right of this box is a blue arrow pointing left towards the text "Edit and n" and "ap". Below this is a search bar with a magnifying glass icon and a pencil icon. A vertical palette on the left side is highlighted with a blue border and contains the following categories and components:

- SOURCE**
 - Icon of a grid with data points
- EVENT HUBS**
 - Icon of a cloud with a gear
- HDFS**
 - Icon of a cube
- KAFKA**
 - Icon of a cluster of nodes
- PROCESSOR**
 - AGGREGATE**
 - Icon of a summation symbol (Σ)
 - BRANCH**
 - Icon of a branching arrow
 - JOIN**
 - Icon of a joining arrow
 - PMML**
 - Icon of a microchip
 - PROJECTION BOLT**
 - Icon of a bolt with a square head
 - RULE**
 - Icon of three interconnected nodes
 - ENRICH**
 - Icon of a circuit board

A blue arrow points from the text "Processor, source, and sink palette contains builder components" to the vertical palette.

Add a Source

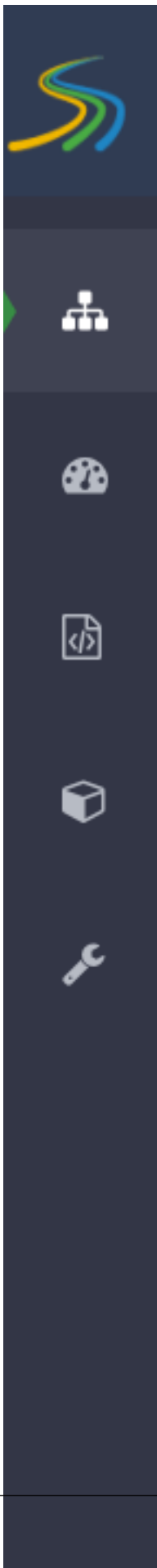
As described in the *HDF Overview*, Stream Builder offers four types of builder components: sources, processors, sinks, and custom components. Start building your application by adding a source.

Before you begin

You have configured Schema Registry and integrated with SAM.

Procedure

1. Drag a source builder component, Kafka for example, onto the canvas. This creates a Kafka tile component:



My Applications / IOT-Trucking-Ref-App

SEARCH [icon] EDIT [icon]

SOURCE

[icon] EVENT HUBS

[icon] HDFS

[icon] KAFKA

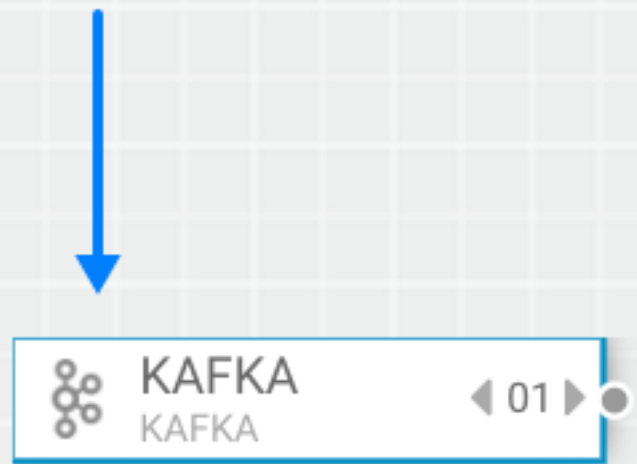
PROCESSOR

[icon] AGGREGATE

[icon] BRANCH

[icon] JOIN

Kafka source tile



Click the arrows to increase or decrease the number of builder component instances for performance and scalability needs

2. Double-click the tile to begin configuring Kafka. After you specify a Kafka topic name, SAM communicates with Schema Registry and displays the schema:

TruckGeoEvent

Kafka connection settings are populated by SAM based on the Kafka service in Environment the Service Pool

REQUIRED OPTIONAL NOTES

CLUSTER NAME *

streamanalytics

SECURITY PROTOCOL *

PLAINTEXT

BOOTSTRAP SERVERS *

secure-fenton-hdf5.field.hortonworks.com:6667,secure

KAFKA TOPIC *

truck_events_avro

CONSUMER GROUP ID *

truck_geo_event_1|

After you select a Kafka topic, SAM fetches the topic schema from Schema Registry

3. Add the additional components you want to use to develop your stream app.

Results

When you have added and correctly configured your stream app components, the component tile displays a green dot on the left. You cannot connect a source to different processors or sinks until it is correctly configured.

Connect Components

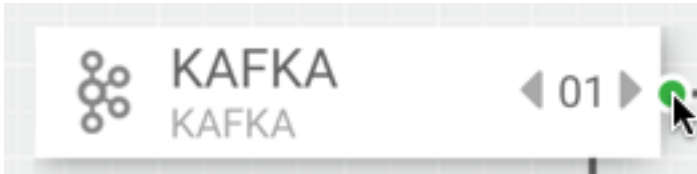
Once you have added and configured your source, add additional processors and sinks to the canvas. To pass a stream of events from one component to the next, create a connection between the two components. In addition to defining data flow, connections allow you to pass a schema from one component to another.

Before you begin

You have added and configured at least one source.

Procedure

1. Click the green dot to the left of your source component.



2. Drag your cursor to the component tile to which you want to connect.

Join Multiple Streams

Joining multiple streams is an important SAM capability. You accomplish this by adding the Join processor to your stream application.

Procedure

1. Drag a Join processor onto your canvas and connect it to a source.
2. Double click the Join tile to open the **Configuration** dialog.
3. Configure the Join processors according to your streaming application requirements.

Example

JOIN

Join stream_1 on field driverId

CONFIGURATION NOTES

Input

kafka_stream_1

eventTime*
STRING

eventSource*
STRING

truckId*
INTEGER

driverId*
INTEGER

driverName*
STRING

routeId*
INTEGER

route*
STRING

eventType*
STRING

latitude*
DOUBLE

longitude*
DOUBLE

correlationId*
LONG

JOIN TYPE

INNER

SELECT STREAM

kafka_stream_2

WINDOW INTERVAL TYPE*

Time

WINDOW INTERVAL*

05

Seconds

OUTPUT FIELDS*

× eventTime × eventSource × truckId × driverId

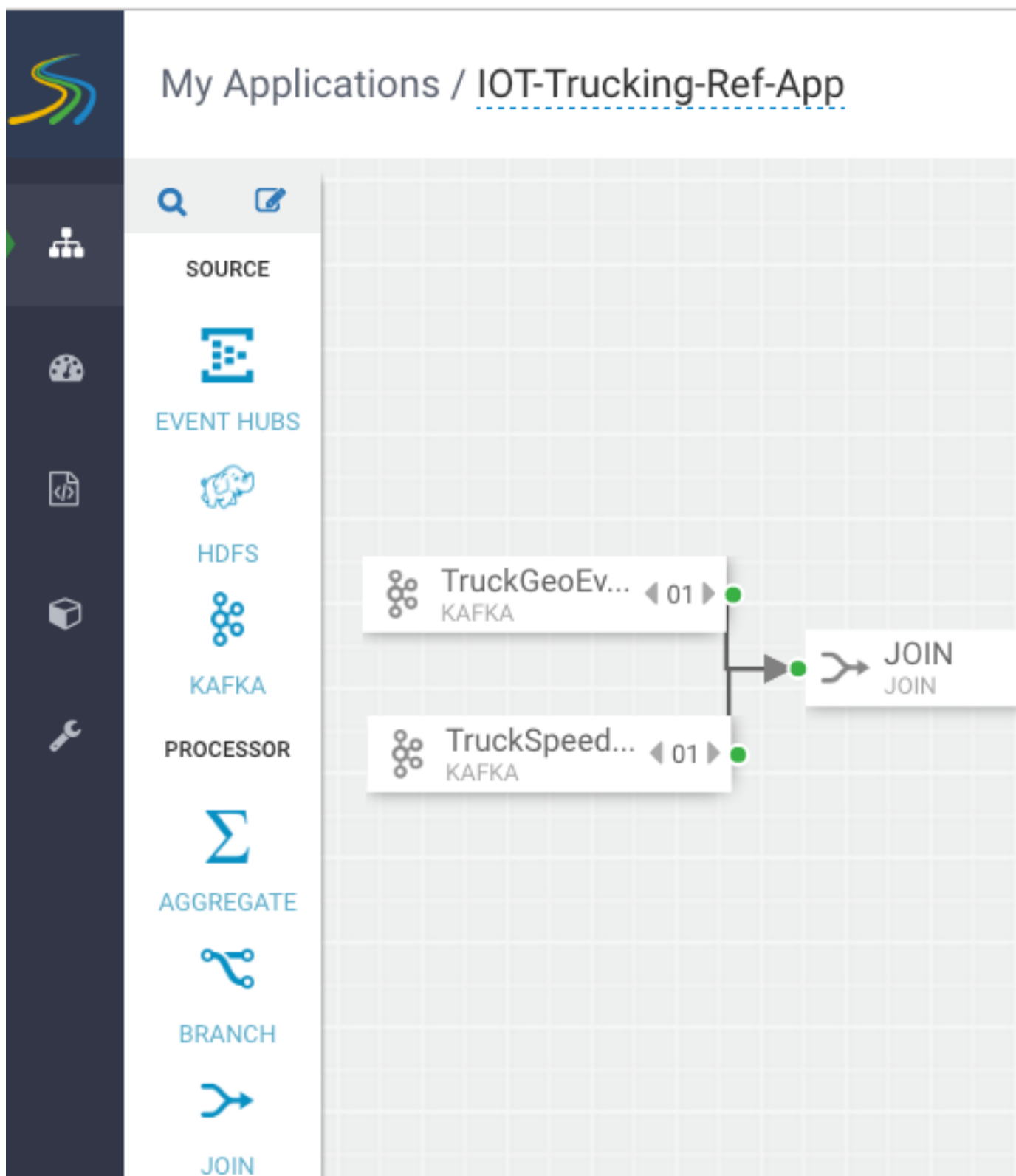
× latitude × longitude × correlationId × geo

Filter Events in a Stream

You can use SAM to filter events in the stream. You accomplish this by using Rule processor, which translates rules into SQL queries that operate on the stream of data.

Procedure

1. Drag the Rule processor to the canvas and connect it to the Join processors.



2. Double click the Rule processor, click the + **Add New Rules** button, and create a new rule:

Add New Rule

RULE NAME*

Violation Event

DESCRIPTION*

Events that are infractions from drivers and trucks

CREATE QUERY*

eventType



NO

QUERY PREVIEW:

```
eventType <> 'Normal'
```

3. Click **Ok** to save the new rule.

Example

EventType

CONFIGURATION NOTES

Input

- eventTime*
STRING
- eventSource*
STRING
- truckId*
INTEGER
- driverId*
INTEGER
- driverName*
STRING
- routeId*
INTEGER
- route*
STRING
- eventType*
STRING
- latitude*
DOUBLE
- longitude*
DOUBLE
- correlationId*
LONG

[+Add New Rules](#)

Name	Condition
Violation Event	eventType

A rule that i
that looks
stream wi
equal to Nor
a V

Use Aggregate Functions over Windows

Windowing is the ability to split an unbounded stream of data into finite sets based on specified criteria such as time or count, so that you can perform aggregate functions (such as sum or average) on the bounded set of events. In SAM, you accomplish this using the Aggregate processor. The Aggregate processor supports two window types, tumbling and sliding windows. You can create a window based on time or count.

Procedure


1. Drag the Aggregate processor to the canvas and connect it to the stream application you are building.
2. Double click the Aggregate tile to configure it according the your stream application requirements.

Example

DriverAvgSpeed

CONFIGURATION NOTES

The fields to group by



Input

- truckId*
INTEGER
- driverId*
INTEGER
- driverName*
STRING
- routeId*
INTEGER
- route*
STRING
- eventType*
STRING
- latitude*
DOUBLE
- longitude*
DOUBLE
- correlationId*
LONG
- geoAddress*
STRING
- speed*
INTEGER

SELECT KEYS*

× driverId × driverName × route

WINDOW INTERVAL TYPE*

Time

WINDOW INTERVAL*

3

SLIDING INTERVAL

3

TIMESTAMP FIELD

processingTime ×

Output Fields

Deploying a Stream App

Configure Deployment Settings

Before deploying the application, it is important to configure deployment settings such as JVM size, number of ackers, and number of workers.

Because this topology uses a number of joins and windows, you should increase the JVM heap size for the workers. Click the gear icon on the top right corner of the canvas, and increase the number of workers (e.g.: 5) and increase the JVM heap memory (-Xmx3072m).

Topology Configuration

NUMBER OF WORKERS

5

NUMBER OF ACKERS

1

TOPOLOGY MESSAGE TIMEOUT (SECONDS)

40

WORKER JVM OPTIONS

-Xmx3072m

HBase config

HBASE ROOT DIRECTORY *

hdfs://localhost:9000/tmp/hbase

Cancel

Deploy the App

After the app's deployment settings has been configured, click the Deploy button on the lower right of the canvas. During the deployment process, Streaming Analytics Manager completes the following tasks:

Procedure

1. Construct the configurations for the different big data services used in the stream app.

2. Create a deployable jar of the streaming app.
3. Upload and deploy the app jar to streaming engine server.

Results

The stream app is deployed to a Storm cluster based on the Storm Service defined in the Environment associated with the app.

My Applications / IOT-Trucking-Ref-App

SOURCE

- EVENT HUBS
- HDFS
- KAFKA**

PROCESSOR

- AGGREGATE
- BRANCH
- JOIN**
- PMML

PROJECTION

RULE

SINK

```
graph LR; S1[TruckGeoEv... KAFKA < 01 >] --> J[JOIN JOIN < 01 >]; S2[TruckSpeed... KAFKA < 01 >] --> J;
```

After the application has been deployed successfully, Streaming Analytics Manager notifies you and updates the status to Active, as shown in the following diagram.

The screenshot displays the Hortonworks Streaming Analytics Manager interface for an application named "IOT-Trucking-Ref-App". On the left is a vertical navigation sidebar with icons for various components: SOURCE, EVENT HUBS, HDFS, KAFKA, PROCESSOR, AGGREGATE, BRANCH, JOIN, PMML, PROJECTION, and RULE. The main workspace shows a data flow diagram on a grid background. It consists of three nodes: "TruckGeoEv... KAFKA" (SOURCE), "TruckSpeed... KAFKA" (SOURCE), and "JOIN JOIN" (PROCESSOR). Both source nodes have a "01" label and a right-pointing arrow. Arrows from both source nodes converge into the "JOIN JOIN" node, which also has a "01" label and a right-pointing arrow. The "JOIN JOIN" node icon depicts two paths merging into one.