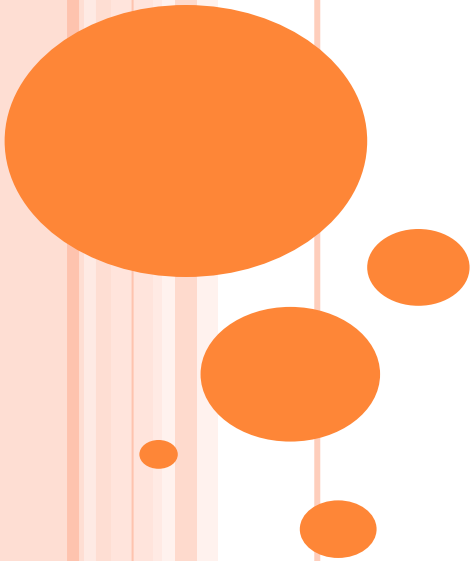


Apache Spark Streaming

07/25/18

Introduction

- Poll
- Role
  - Developers, Architects?
  - Data Analysts?
  - Data Scientists?
- Spark familiarity
  - Apache Spark
  - Spark Streaming

Who am I

- Software Architect
- Lead Editor for AI, ML & Data Engg @ InfoQ
- Books
  - Big Data Processing with Apache Spark (2018)
  - Spring Roo in Action (2012)
- Current Focus:
  - Big Data Application Architectures
  - Reactive Microservices & Service Mesh
  - Containers

Agenda

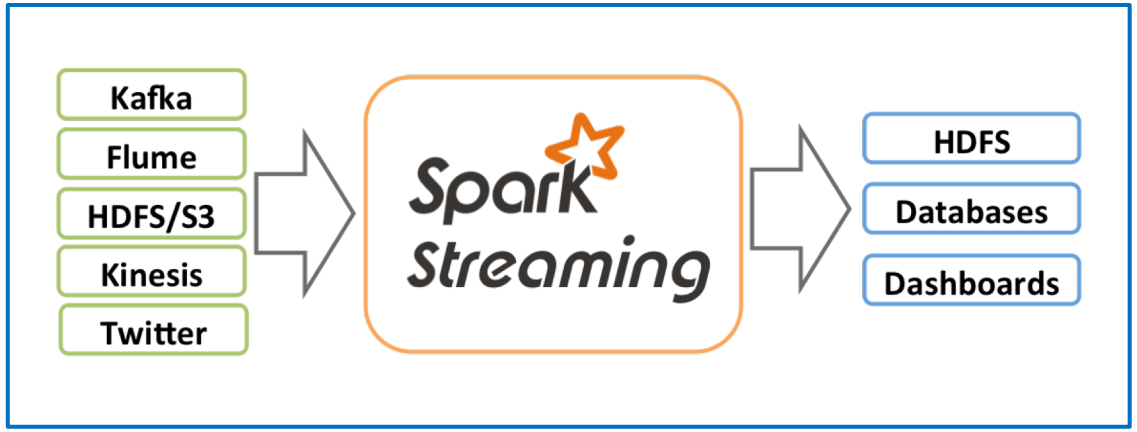
- Introduction
- Event Streaming & Streaming Data Analytics
- Spark Streaming
- Streaming API
- Sample Application
- Visualization
- Conclusions
- Q&A

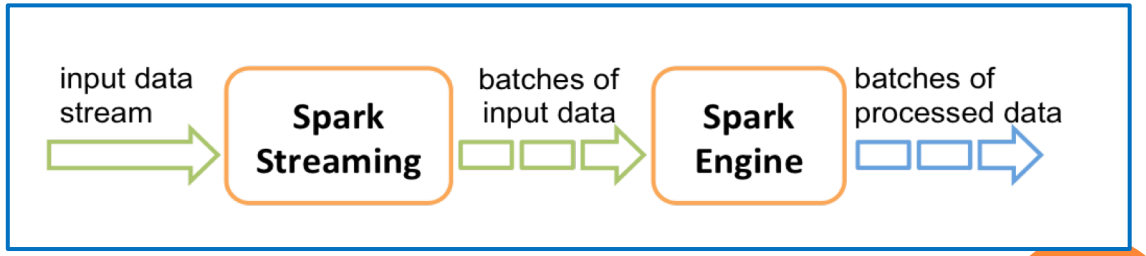
Reference material for this talk

- Big Data Processing with Apache Spark
- GitHub project
- Apache Spark article series on InfoQ
- Video stream analytics article

Event Streaming

- Continuous group of data records
- Generated from sources like:
  - sensors
  - server traffic
  - online searches
- Examples
  - user activity on websites
  - server logs
  - telemetry data





STREAMING USE CASES

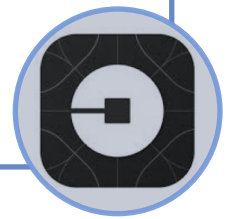
- Real-time online movie recommendation & data monitoring
- Processes billions of events received per day from different data sources

Netflix 

- User engagement insights
- Users across globe mark the pins in real-time

Pinterest 

- Streaming ETL pipeline for real-time telemetry
- Collect TB's of event data every day from mobile users

Uber 

Other real-world examples

- Supply chain analytics
- Real-time security intelligence operations to find threats
- Ad auction platform
- Real-time video analytics

Streaming Data Frameworks

Open Source

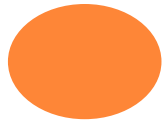
- Apache Samza
- Apache Spark
- Apache Storm
- Kafka (LinkedIn)
- Flume
- Twitter
- ZeroMQ

Commercial

- Esper
- SQLStream
- IBM InfoSphere Streams
- Oracle Event Processor
- SAP Event Stream Processor
- Microsoft StreamInsight
- Tibco StreamBase
- WSO2 CEP Server

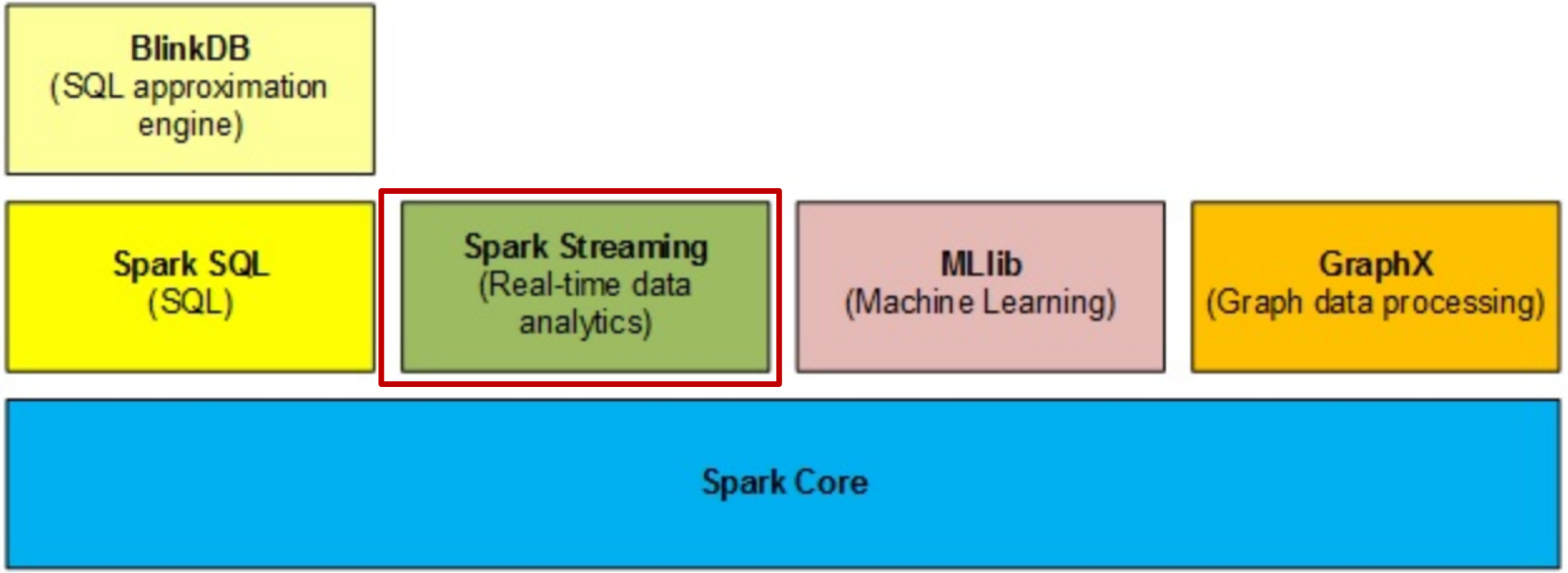
Spark Ecosystem

Data Analytics Area	Spark Library
Batch Data Processing	Spark Core
SQL over Spark	Spark SQL
Real-time Streaming Data Analytics	Spark Streaming
Machine Learning	Spark MLlib
Graph Data Analytics	Spark GraphX
Memory-centric Distributed Storage	Tachyon*



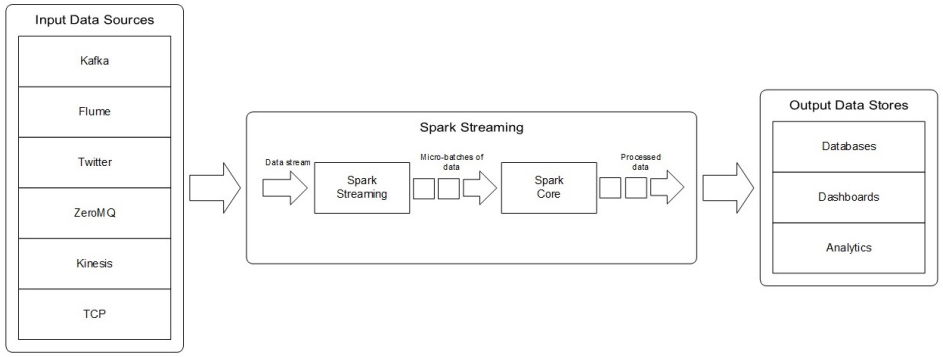
*Not part of Spark installation

Spark Ecosystem



SPARK STREAMING ARCHITECTURE

- Microbatch is small (seconds) batches of the live stream
- Microbatches are treated as RDD
- Same RDD Programming model and operations
- Results are returned as batches
- Recovery based on lineage & Checkpoint



Spark Streaming

- Extension of core Spark API
- Fault-tolerant stream processing of live data streams
- Run streaming computation as a series of small, deterministic batch jobs
- How it works:
  - Divide live stream into batches of N seconds
  - Treat batch of data as RDDs & process using RDD operations
  - Results of the RDD operations are returned in batches

http://spark.apache.org/docs/latest/streamingprogramming-guide.html

Main Components & API

Components

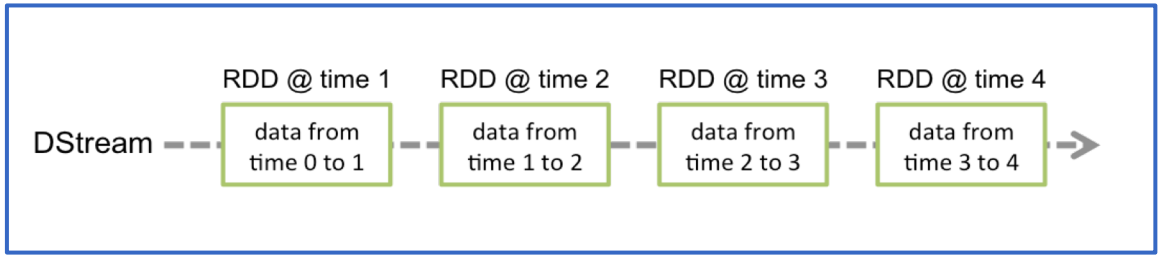
- DStream
- Streaming Context

API

- Scala
- Java
- Python

DStream

- Discretized Stream
- Represents a continuous stream of data
- Sequence of RDD objects



Operations

- map
- flatMap
- filter
- count
- reduce
- countByValue
- reduceByKey
- join
- updateStateByKey

Streaming Context

- Similar to SparkContext
- Entry point for all streaming functionality
- Used to create a DStream object

```
import org.apache.spark._
import org.apache.spark.streaming._

val conf = new
SparkConf().setAppName(appName).setMast
er(master)
val ssc = new StreamingContext(conf,
Seconds(1))
```

Steps in a Streaming Program

1. Initialize StreamingContext
2. Specify input data sources by creating input **DStreams**
3. Define computations using Sparking Streaming **Transformations API**
4. Start receiving data and process using **start** method
5. Wait for streaming data processing to be stopped using **awaitTermination** method

Sample App Details

Use Case

- Real-time video capture & motion detection

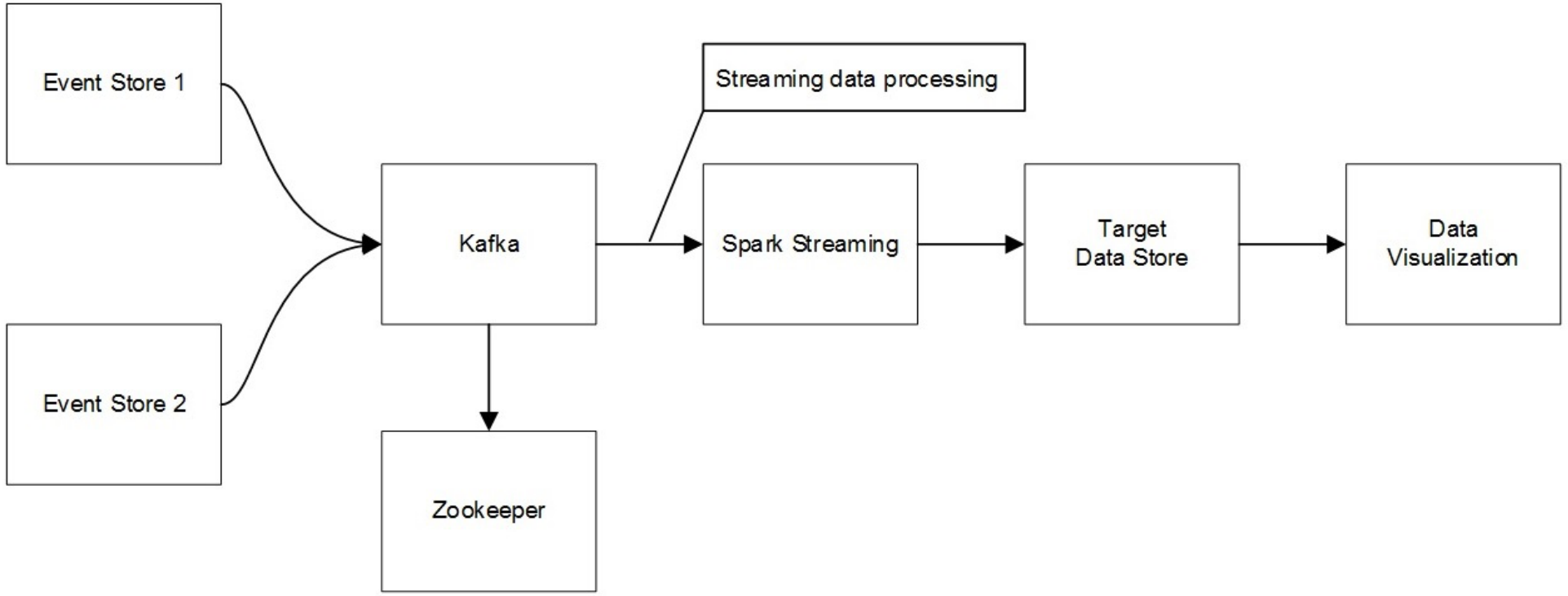
Technologies

- Zookeeper
- Apache Kafka
- Spark Streaming

TECHNOLOGIES

Technology	Version	URL
Zookeeper	3.4.6	https://zookeeper.apache.org/doc/r3.4.6/
Kafka	2.10	http://kafka.apache.org/downloads.html
Spark Streaming	1.4.1	https://spark.apache.org/releases/spark-release-1-4-1.html
JDK	1.7	http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html
Maven	3.3.3	http://archive.apache.org/dist/maven/maven-3/3.3.3/

SPARK STREAMING APPLICATION RUNTIME



ZOOKEEPER

```
//
// Start ZooKeeper
//

export KAFKA_HOME=/dev/tools/kafka_2.11-0.10.2.1

export PATH=$PATH:$KAFKA_HOME/bin

cd $KAFKA_HOME
bin/zookeeper-server-start.sh config/zookeeper.properties
```

KAFKA

```
//
// Start Kafka Server
//

export KAFKA_HOME=/dev/tools/kafka_2.11-0.10.2.1

export PATH=$PATH:$KAFKA_HOME/bin

cd $KAFKA_HOME
bin/kafka-server-start.sh config/server.properties
```

Topic Management

```
// Kafka Topic management commands
export KAFKA_HOME=/dev/tools/kafka_2.11-0.10.2.1
export PATH=$PATH:$KAFKA_HOME/bin

cd $KAFKA_HOME

// List topics
bin/kafka-topics.sh --zookeeper localhost:2181 --list

// Create a new topic
bin/kafka-topics.sh --create --zookeeper localhost:2181 -
-topic video-stream-event --replication-factor 1 --
partitions 3
```

Console Message Producer

```
// Kafka console producer

export KAFKA_HOME=/dev/tools/kafka_2.10-0.10.2.1

export PATH=$PATH:$KAFKA_HOME/bin

cd $KAFKA_HOME
bin/kafka-console-producer.sh --broker-list
localhost:9092 --topic clickstream
```

Console Message Consumer

```
// Kafka console consumer (in a new terminal window)

export KAFKA_HOME=/dev/tools/kafka_2.10-0.10.2.1
export PATH=$PATH:$KAFKA_HOME/bin

cd $KAFKA_HOME
bin/kafka-console-consumer.sh --zookeeper localhost:2181
--topic clickstream  --from-beginning
```

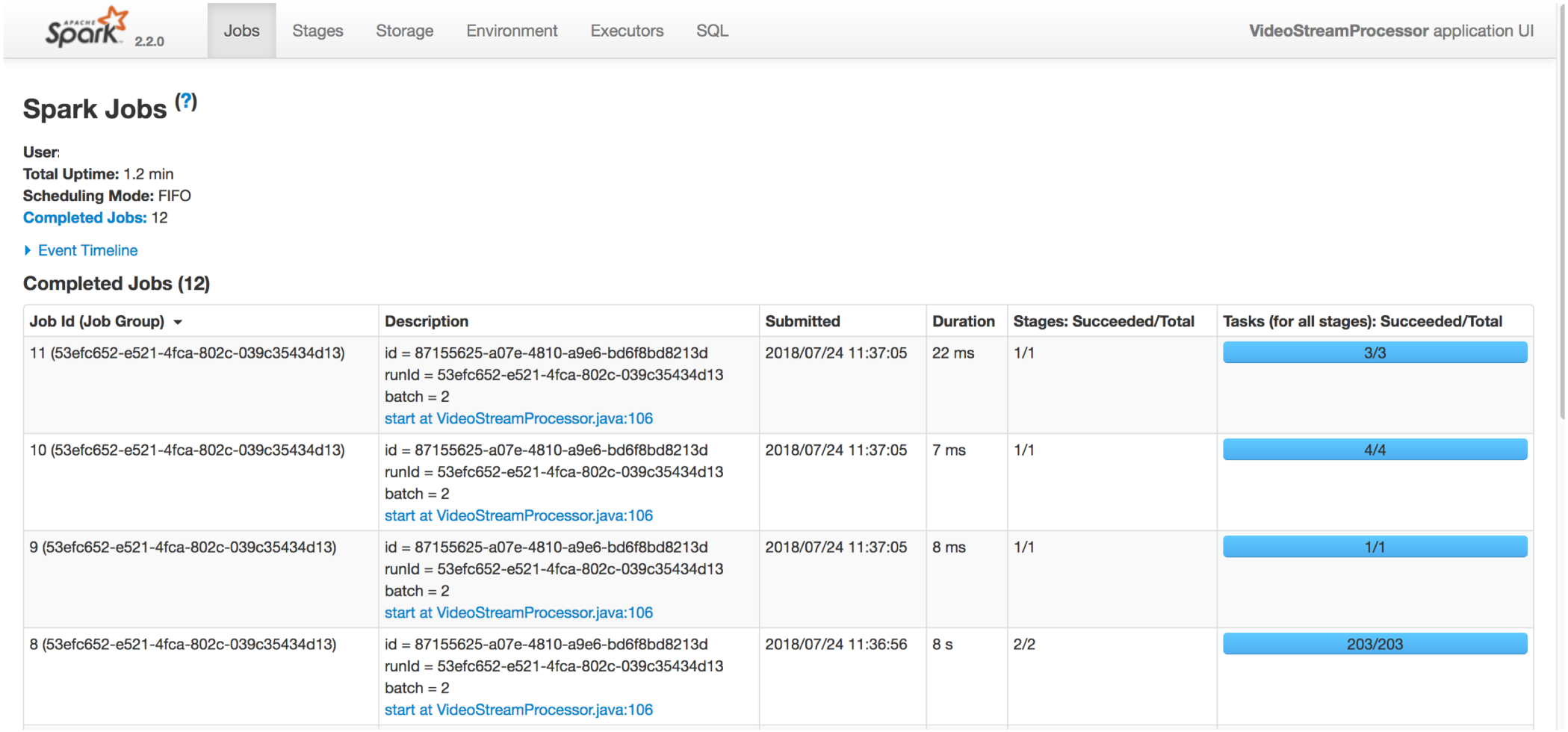
STREAM DATA PROCESSOR

```
cd /dev/projects/video-stream-analysis-master
cd video-stream-processor
export JAVA_HOME=/usr
export PATH=$PATH:$JAVA_HOME/bin

export MAVEN_HOME=/dev/tools/apache-maven-3.5.0
export PATH=$PATH:$MAVEN_HOME/bin

export OPENCV_HOME=/dev/tools/OpenCV/opencv-
3.2.0/build/lib
export PATH=$PATH:$OPENCV_HOME

mvn clean package
mvn exec:java -
Dexec.mainClass="com.iot.video.app.spark.processor.VideoS
```

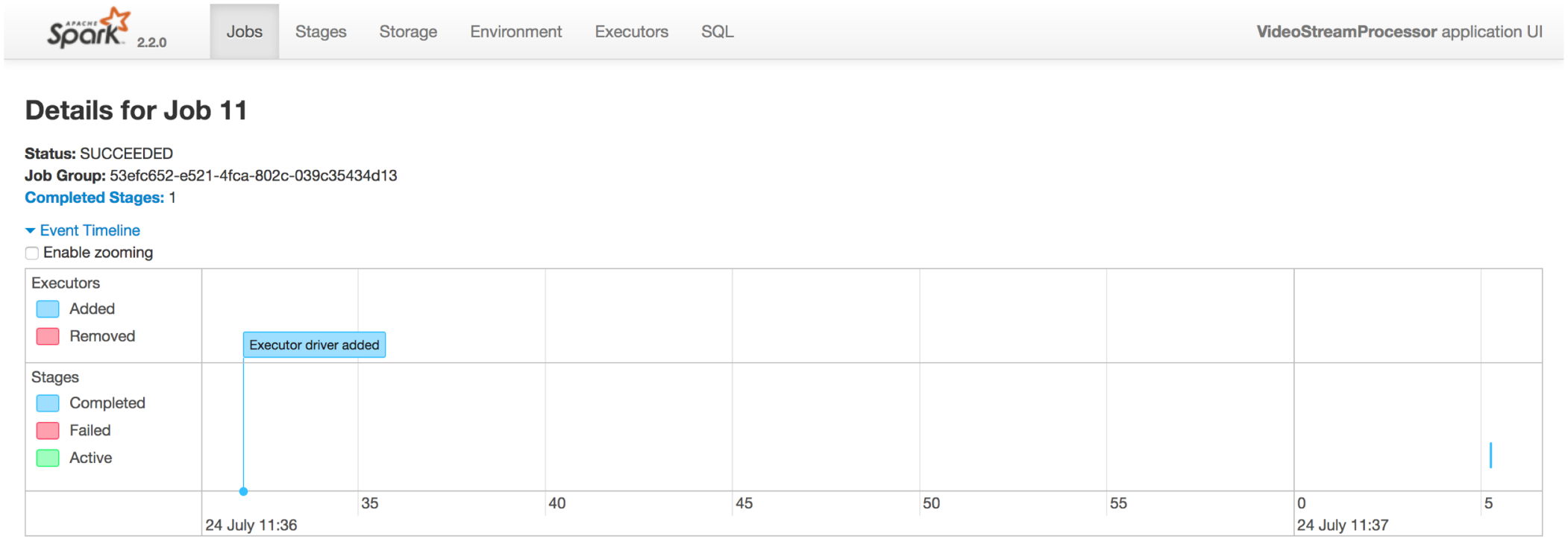
Stream Data Collector

```
//
// Stream data collector
//

cd video-stream-collector
mvn clean package

mvn exec:java -
Dexec.mainClass="com.iot.video.app.kafka.collector.VideoS
treamCollector" -Dexec.cleanupDaemonThreads=false
```

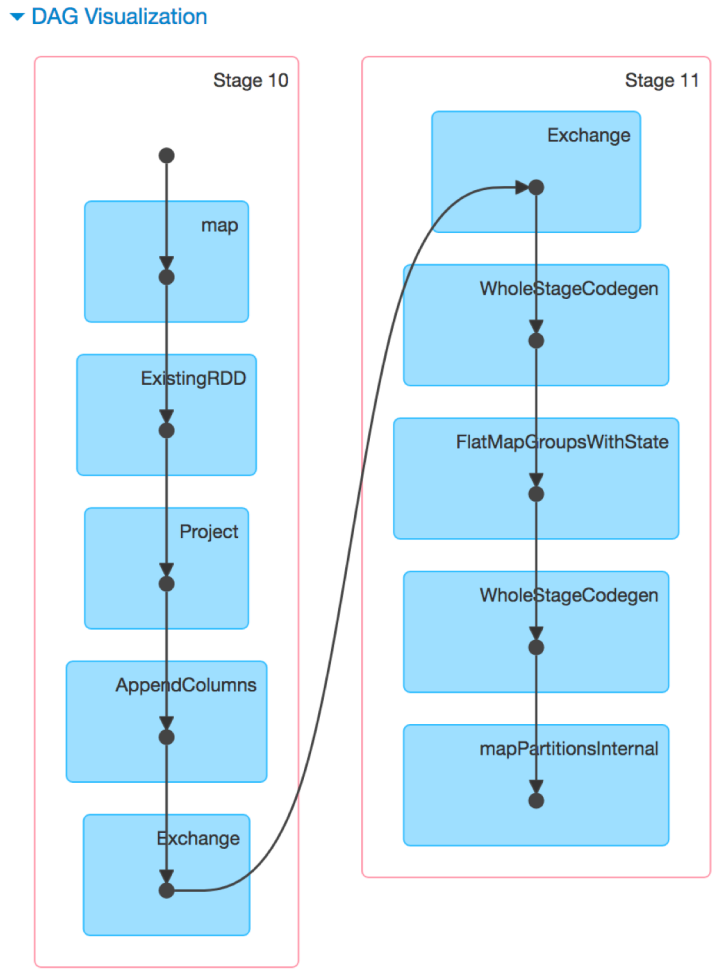
VISUALIZATION



VISUALIZATION



VISUALIZATION



Q&A

Resources

- “Big Data Processing with Apache Spark — article series
- Apache Spark homepage
- Spark Streaming homepage
- Spark Streaming Programming Guide
- Spark Streaming Scala code examples
- Spark Streaming Java code examples
- Tagging and Processing Data in Real-Time Using Spark Streaming (Spark Summit 2015 presentation)