Apache Big Data Seville 2016

Apache Bahir
Writing Applications using Apache Bahir

Luciano Resende
IBM | Spark Technology Center

About Me

Luciano Resende (lresende@apache.org)

- Architect and community liaison at IBM – Spark Technology Center
- Have been contributing to open source at ASF for over 10 years
- Currently contributing to : Apache Bahir, Apache Spark, Apache Zeppelin and Apache SystemML (incubating) projects

Origins of the Apache Bahir Project

MAY/2016: Established as a top-level Apache Project.

- PMC formed by Apache Spark committers/pmc, Apache Members
- Initial contributions imported from Apache Spark

AUG/2016: Flink community join Apache Bahir

- Initial contributions of Flink extensions
- In October 2016 Robert Metzger elected committer

The Apache Bahir name

Naming an Apache Project is a science !!!

- We needed a name that wasn't used yet
- Needed to be related to Spark

We ended up with : Bahir

- A name of Arabian origin that means Sparkling,
  - Also associated with a guy who succeeds at everything

Why Apache Bahir

It's an Apache project

- And if you are here, you know what it means

What are the benefits of curating your extensions at Apache Bahir

- Apache Governance
- Apache License
- Apache Community
- Apache Brand

Why Apache Bahir

Flexibility

- Release flexibility
  - Bounded to platform or component release

Shared infrastructure

- Release, CI, etc

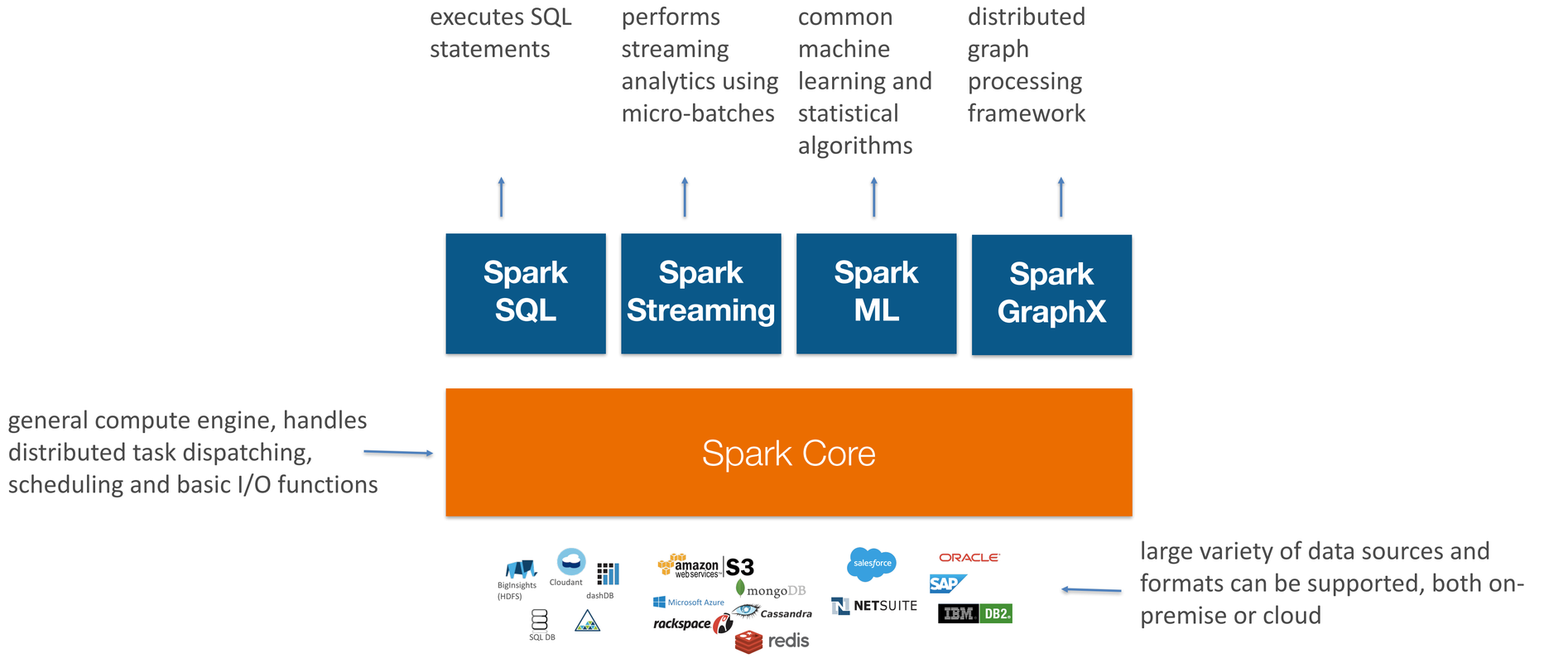
Shared knowledge

- Collaborate with experts on both platform and component areas

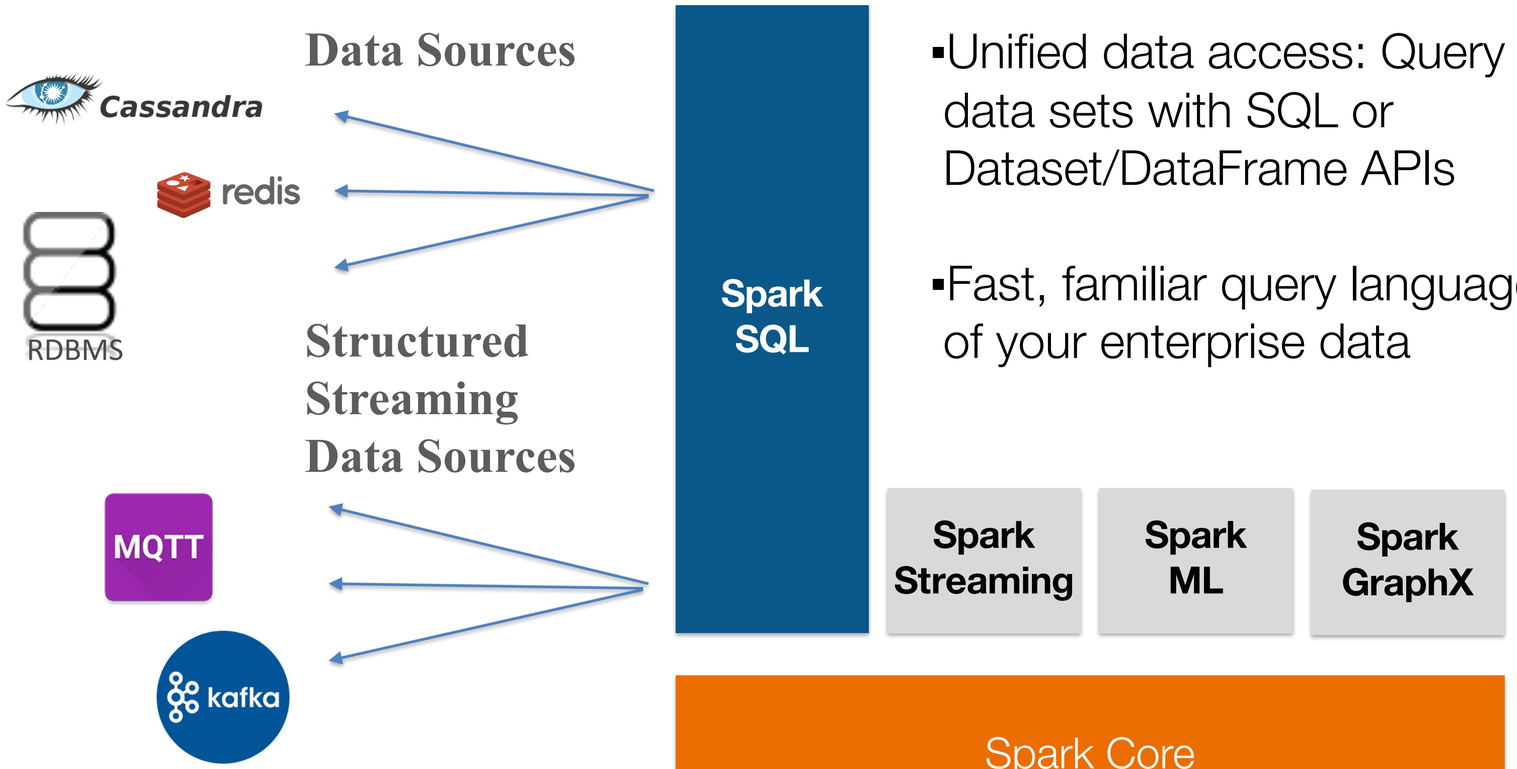
Apache Spark

Apache Spark - Introduction

What is Apache Spark ?



Apache Spark – Spark SQL



- Unified data access: Query structured data sets with SQL or Dataset/DataFrame APIs
- Fast, familiar query language across all of your enterprise data

Apache Spark – Spark SQL

You can run SQL statement with SparkSession.sql(…) interface:

```
val spark = SparkSession.builder()
  .appName("Demo")
  .getOrCreate()
spark.sql("create table T1 (c1 int, c2 int) stored as parquet")
val ds = spark.sql("select * from T1")
```

You can further transform the resultant dataset:

```
val ds1 = ds.groupBy("c1").agg("c2"-> "sum")
val ds2 = ds.orderBy("c1")
```

The result is a DataFrame / Dataset[Row]

ds.show() displays the rows

Apache Spark – Spark SQL

You can read from data sources using SparkSession.read.format(…)

```
val spark = SparkSession.builder()
  .appName(“Demo”)
  .getOrCreate()

case class Bank(age: Integer, job: String, marital: String, education: String,  balance: Integer)

// loading csv data to a Dataset of Bank type
val bankFromCSV = spark.read.csv(“hdfs://localhost:9000/data/bank.csv").as[Bank]
// loading JSON data to a Dataset of Bank type
val bankFromJSON = spark.read.json(“hdfs://localhost:9000/data/bank.json").as[Bank]
// select a column value from the Dataset
bankFromCSV.select(‘age).show() will return all rows of column “age” from this dataset.
```

Apache Spark – Spark SQL

You can also configure a specific data source with specific options

```
val spark = SparkSession.builder()
  .appName(“Demo”)
  .getOrCreate()
case class Bank(age: Integer, job: String, marital: String, education: String,  balance: Integer)
// loading csv data to a Dataset of Bank type
val bankFromCSV = sparkSession.read
  .option("header", ”true") // Use first line of all files as header
  .option("inferSchema", ”true") // Automatically infer data types
  .option("delimiter", " ")
  .csv("/users/lresende/data.csv”)
  .as[Bank]
bankFromCSV.select(‘age).show()  // will return all rows of column “age” from this dataset.
```

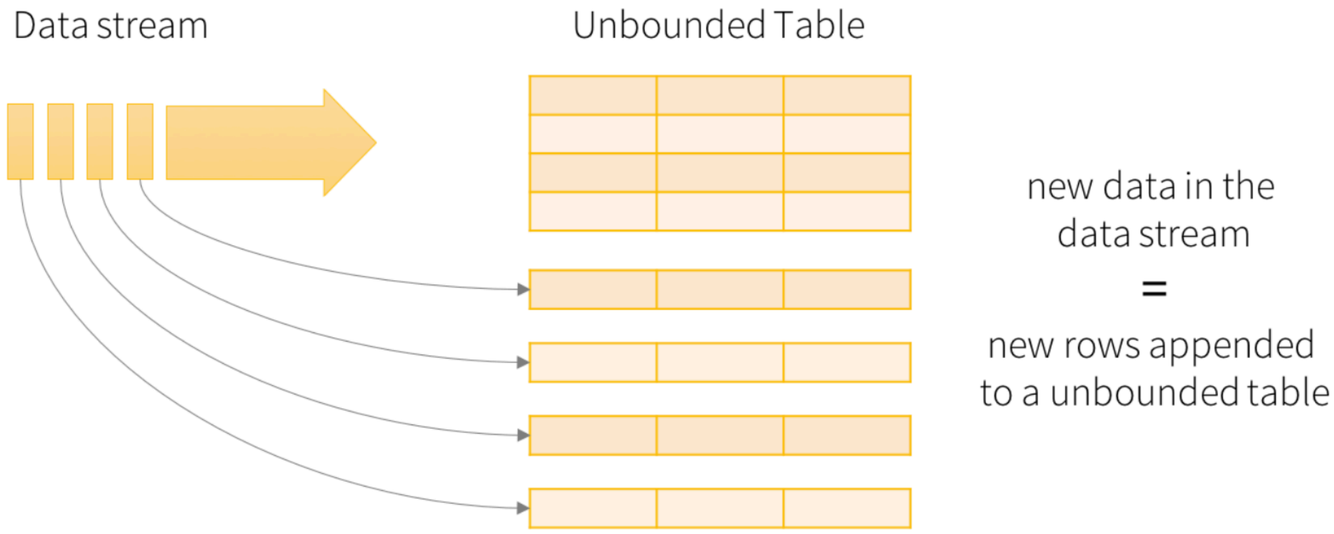
Apache Spark – Spark SQL

Data Sources under the covers

- Data source registration (e.g. spark.read.datasource)
- Provide BaseRelation implementation
  - That implements support for table scans:
    - TableScans, PrunedScan, PrunedFilteredScan, CatalystScan
- Detailed information available at
  - http://www.spark.tc/exploring-the-apache-spark-datasource-api/

Apache Spark – Spark SQL Structured Streaming

Unified programming model for streaming, interactive and batch queries



Considers the data stream as unbounded table

Image source: https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html

Apache Spark – Spark SQL Structured Streaming

SQL regular APIs

```
val spark = SparkSession.builder()
  .appName(“Demo”)
  .getOrCreate()
val input = spark.read
    .schema(schema)
    .format(”csv")
    .load(”input-path")
val result = input
    .select(”age”)
    .where(”age > 18”)
result.write
    .format(”json”)
    . save(” dest-path”)
```

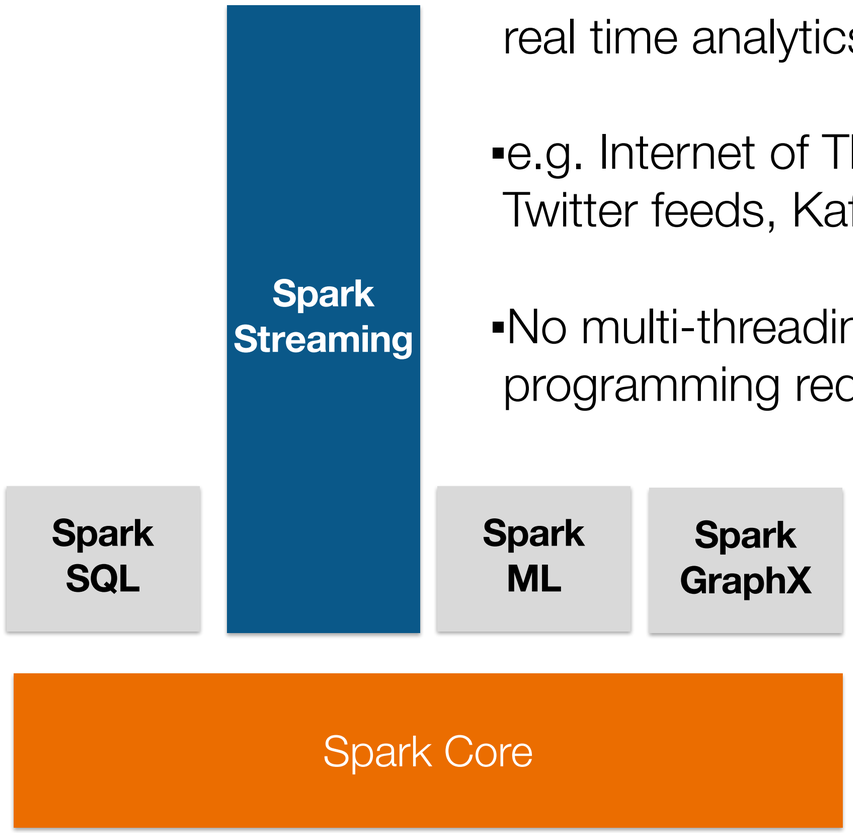
Structured Streaming APIs

```
val spark = SparkSession.builder()
  .appName(“Demo”)
  .getOrCreate()
val input = spark.readStream
    .schema(schema)
    .format(”csv")
    .load(”input-path")
val result = input
    .select(”age”)
    .where(”age > 18”)
result.write
    .format(”json”)
    . startStream(” dest-path”)
```

Apache Spark – Spark SQL Structured Streaming

Structured Streaming is an ALPHA feature

Apache Spark – Spark Streaming



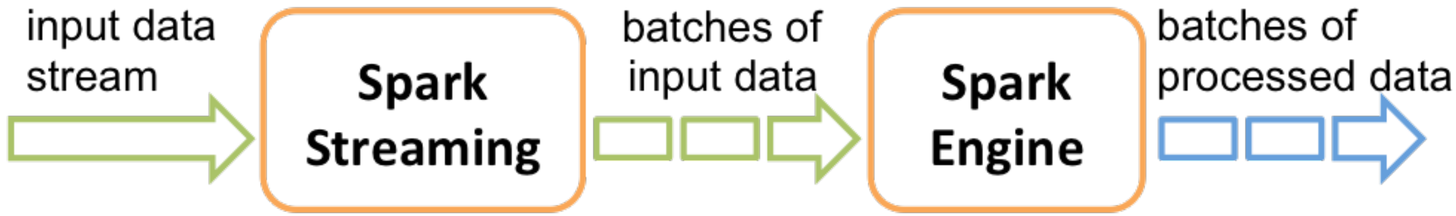
- Micro-batch event processing for near-real time analytics
- e.g. Internet of Things (IoT) devices, Twitter feeds, Kafka (event hub), etc.
- No multi-threading or parallel process programming required

Apache Spark – Spark Streaming

Also known as discretized stream or Dstream

Abstracts a continuous stream of data

Based on micro-batching



Apache Spark – Spark Streaming

```
 val sparkConf = new SparkConf()
   .setAppName("MQTTWordCount")

val ssc = new StreamingContext(sparkConf, Seconds(2))
val lines = MQTTUtils.createStream(ssc, brokerUrl, topic, StorageLevel.MEMORY_ONLY_SER_2)

val words = lines.flatMap(x => x.split(" "))
val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)

wordCounts.print()

ssc.start()
ssc.awaitTermination()
```

Apache Spark extensions in Bahir

MQTT – Enables reading data from MQTT Servers using Spark Streaming or Structured streaming.

- http://bahir.apache.org/docs/spark/current/spark-sql-streaming-mqtt/
- http://bahir.apache.org/docs/spark/current/spark-streaming-mqtt/

Twitter – Enables reading social data from twitter using Spark Streaming.

- http://bahir.apache.org/docs/spark/current/spark-streaming-twitter/

Akka – Enables reading data from Akka Actors using Spark Streaming.

- http://bahir.apache.org/docs/spark/current/spark-streaming-akka/

ZeroMQ – Enables reading data from ZeroMQ using Spark Streaming.

- http://bahir.apache.org/docs/spark/current/spark-streaming-zeromq/

Apache Spark extensions coming soon to Bahir

WebHDFS – Enables reading data from remote HDFS file system utilizing Spark SQL APIs

- https://issues.apache.org/jira/browse/BAHIR-67

CounchDB / Cloudant– Enables reading data from CounchDB NoSQL document stores using Spark SQL APIs

Apache Spark extensions in Bahir

Adding Bahir extensions into your application

- Using SBT
  - libraryDependencies += "org.apache.bahir" %% "spark-streaming-mqtt" % "2.1.0-SNAPSHOT”
- Using Maven
  - <dependency>
    <groupId>org.apache.bahir</groupId>
    <artifactId>spark-streaming-mqtt_2.11 </artifactId>
    <version>2.1.0-SNAPSHOT</version>
    </dependency>

Apache Spark extensions in Bahir

Submitting applications with Bahir extensions to Spark

- Spark-shell
  - bin/spark-shell --packages org.apache.bahir:spark-streaming_mqtt_2.11:2.1.0-SNAPSHOT …..
- Spark-submit
  - bin/spark-submit --packages org.apache.bahir:spark-streaming_mqtt_2.11:2.1.0-SNAPSHOT …..

Apache Flink

Apache Flink extensions in Bahir

Flink platform extensions added recently

- https://github.com/apache/bahir-flink

First release coming soon

- Release discussions have started
- Finishing up some basic documentation and examples
- Should be available soon

Apache Flink extensions in Bahir

ActiveMQ – Enables reading and publishing data from ActiveMQ servers

- https://github.com/apache/bahir-flink/blob/master/flink-connector-activemq/README.md

Flume– Enables publishing data to Apache Flume

- https://github.com/apache/bahir-flink/tree/master/flink-connector-flume

Redis – Enables reading data to Redis and publishing data to Redis PubSub

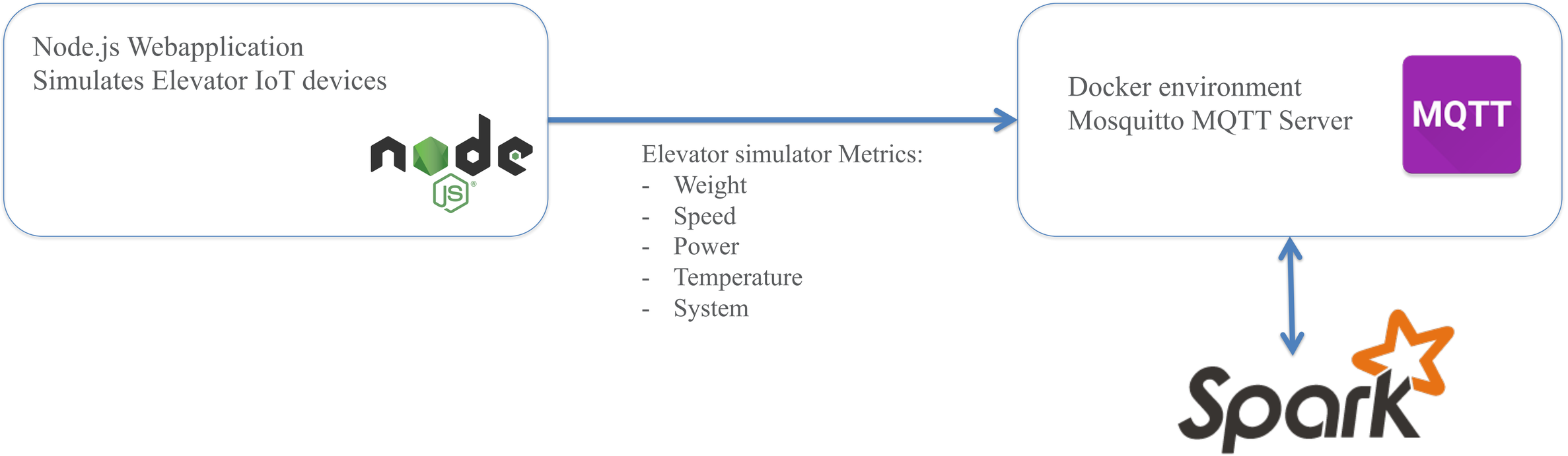
- https://github.com/apache/bahir-flink/blob/master/flink-connector-redis/README.md

Live Demo

IoT Simulation using MQTT

The demo environment

https://github.com/lresende/bahir-iot-demo



Join the Apache Bahir community !!!

References

Apache Bahir

http://bahir.apache.org

Documentation for Apache Spark extensions

http://bahir.apache.org/docs/spark/current/documentation/

Source Repositories

https://github.com/apache/bahir

https://github.com/apache/bahir-flink

https://github.com/apache/bahir-website

Image source: http://az616578.vo.msecnd.net/files/2016/03/21/635941249931013850155786752 9_thank-you-1400x800-c-default.gif