Rethinking Stream Processing with Apache Kafka: Applications vs. Clusters, Streams vs. Databases

Michael G. Noll
Confluent

Strata Data Conference, London, May 2017
Apache Kafka: birthed as a messaging system, now a **streaming platform**
SECURITY GUARD

APP

MACHINE
SECURITY GUARD

RECEPTIONIST

ROOM SERVICE

MICROSERVICES

APP
1 MACHINE

APP
1 MACHINE

APP
1 MACHINE
SECURITY GUARD

SECURITY TEAM

DISTRIBUTED APPS

1 APP ACROSS MANY MACHINES
AS APP DEVS WE WANT TO
BUILD APPS
(NOT INFRA)
WE WANT OUR APPS TO BE

ELASTIC
SCALABLE
FAULT-TOLERANT
STATEFUL
DISTRIBUTED
WE WANT OUR APPS TO BE

ELASTIC

SCALABLE

FAULT-TOLERANT

STATEFUL

DISTRIBUTED

SHARED PROCESSING CLUSTERS?

SHARED DATABASES?

(STATE)
TASTE, NO ALCOHOL (SAFE DRIVING)
TASTE,

NO SUGAR
(HEALTHY TEETH)
Kafka Streams API

Java library, 0.10+

to build real-time applications that power your core business
"ENRICH MY LIBRARY BECOMES A LIBRARY THAT ENRICHES MY APPS"
FEATURES (FOR YOUR APP)

"CLUSTER TO GO"
RUNS EVERYWHERE
EXACTLY-ONCE PROCESSING
EVENT-TIME

"DATABASE TO GO"
S/M/L/XXL USES
INTERACTIVE QUERIES
JOINS, AGGREGATIONS, WINDOWING

... AND MORE...
(Does NOT run inside the Kafka brokers!)
(Does NOT run inside the Kafka brokers!)
DEMO TIME!

CONFLUENT "KAFKA MUSIC" APP

http://docs.confluent.io/current/cp-docker-images/docs/tutorials/kafka-streams-examples.html
Before

With Kafka’s Streams API
KStream<Integer, Integer> input = builder.stream("numbers-topic");

// Stateless computation
KStream<Integer, Integer> doubled = input.mapValues(v -> v * 2);

// Stateful computation
KTable<Integer, Integer> sumOfOdds = input
    .filter((k, v) -> v % 2 != 0)
    .selectKey((k, v) -> 1)
    .groupByKey()
    .reduce((v1, v2) -> v1 + v2, "sum-of-odds");

class PrintToConsoleProcessor implements Processor<K, V> {
    @Override
    public void init(ProcessorContext context) {}

    @Override
    void process(K key, V value) {
        System.out.println("Got value " + value);
    }

    @Override
    void punctuate(long timestamp) {}

    @Override
    void close() {}
"HELLO, KAFKA STREAMS"
"ALL STREAMS LEAD TO KAFKA"

HELLO, 1
KAFKA, 1
STREAMS, 1
ALL, 1
STREAMS, 2
...
WORD COUNT APP: DEVELOPING

OPERATING SYSTEMS

Linux

MacOS

Windows
WORD COUNT APP: DEVELOPING

```java
KStream<String, String> textLines = builder.stream("TextLinesTopic");
KTable<String, Long> wordCounts = textLines
  .flatMapValues(value -> Arrays.asList(value.toLowerCase().split("\\W+")))
  .groupBy((key, word) -> word)
  .count("Counts");
wordCounts.to(Serdes.String(), Serdes.Long(), "WordsWithCountsTopic");
```
STREAMS ARE EVERYWHERE!
STREAMS ARE EVERYWHERE!

TABLES ARE EVERYWHERE, TOO!
STREAMS + TABLES

Both are needed all the time

- Customer 360°
- Tracking shipments
- Shopping carts

...
STREAMS + TABLES

Both are needed all the time

- Customer 360°
- Tracking shipments
- Shopping carts

...
https://kafka.apache.org/documentation/streams#streams_duality
STREAM ➔ TABLE

CONTINUOUSLY UPDATING

EXAMPLE:
CUSTOMER 360°
*Cough* WORD COUNT

MATERIALIZED VIEW
TABLE → STREAM

MUTATIONS

INSERT INSERT DELETE UPDATE ...
0 1 2 3 4 5 6 7 8

EXAMPLES:
CHANGE DATA CAPTURE,
ALERTING ON CUSTOMER EVENTS
STREAM + TABLE

"ALL PASSENGERS"

TRANSFORM

"PASSENGERS IN TRANSIT TO EUROPE"
WORD COUNT APP: DEVELOPING

```java
KStream<String, String> textLines = builder.stream("TextLinesTopic");
KTable<String, Long> wordCounts = textLines
    .flatMapValues(value -> Arrays.asList(value.toLowerCase().split("\\W+")))
    .groupBy((key, word) -> word)
    .count("Counts");
wordCounts.to(Serdes.String(), Serdes.Long(), "WordsWithCountsTopic");
```
KStream<

KStreamBuilder builder = new KStreamBuilder();
KStream<String, String> textLines = builder.stream("TextLinesTopic");
KTable<String, Long> wordCounts = textLines
.flatMapValues(value -> Arrays.asList(value.toLowerCase().split("\\W+"))
 .groupBy((key, word) -> word)
 .count("Counts");
wordCounts.to(Serdes.String(), Serdes.Long(), "WordsWithCountsTopic");
public static void main(final String[] args) throws Exception {
    Properties config = new Properties();
    config.put(StreamsConfig.APPLICATION_ID_CONFIG, "wordcount-example");
    config.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "kafka-broker1:9092");
    config.put(StreamsConfig.KEY_SERDE_CLASS_CONFIG, Serdes.String().getClass().getName());
    config.put(StreamsConfig.VALUE_SERDE_CLASS_CONFIG, Serdes.String().getClass().getName());

    KStreamBuilder builder = new KStreamBuilder();
    KStream<String, String> textLines = builder.stream("TextLinesTopic");
    KTable<String, Long> wordCounts = textLines
        .flatMapValues(value -> Arrays.asList(value.toLowerCase().split("\\W")))
        .groupBy((key, word) -> word)
        .count("Counts");
    wordCounts.to(Serdes.String(), Serdes.Long(), "WordsWithCountsTopic");
public static void main(final String[] args) throws Exception {
    Properties config = new Properties();
    config.put(StreamsConfig.APPLICATION_ID_CONFIG, "wordcount-example");
    config.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "kafka-broker1:9092");
    config.put(StreamsConfig.KEY_SERDE_CLASS_CONFIG, Serdes.String().getClass().getName());
    config.put(StreamsConfig.VALUE_SERDE_CLASS_CONFIG, Serdes.String().getClass().getName());

    KStreamBuilder builder = new KStreamBuilder();
    KStream<String, String> textLines = builder.stream("TextLinesTopic");
    KTable<String, Long> wordCounts = textLines
        .flatMapValues(value -> Arrays.asList(value.toLowerCase().split("\W+")))
        .groupBy((key, word) -> word)
        .count("Counts");
    wordCounts.to(Serdes.String(), Serdes.Long(), "WordsWithCountsTopic");

    KafkaStreams streams = new KafkaStreams(builder, config);
    streams.start();
}
WORD COUNT APP: TESTING

UNIT TESTING
INTEGRATION TESTING
SYSTEM TESTING
**WORD COUNT APP: TESTING**

<table>
<thead>
<tr>
<th>IN-PROCESS</th>
<th>KAFKA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CONFLUENT SCHEMA REGISTRY</td>
</tr>
<tr>
<td></td>
<td>WHATEVER ELSE YOU NEED</td>
</tr>
</tbody>
</table>
WORD COUNT APP: TESTING

GitHub

Jenkins

TeamCity

Travis CI

circleci

...and more...
WORD COUNT APP: DEPLOYING

VERY SMALL MINIMUM FOOTPRINT

EQUALLY VAILABLE FOR S/M/L/XXL USE CASES
WORD COUNT APP: DEPLOYING RUNS EVERYWHERE (JVM)
WORD COUNT APP: DEPLOYING
RUNS EVERYWHERE (JVM)

...and more...
Is Kafka's Streams API ready for production?

Of course, why do you ask?

It is only one year old.

Kafka solves all the hard problems for it.

You use that answer for everything, don’t you?
Kafka Streams API in the wild

First release of Kafka’s Streams API (0.10.0.0)

2016

In production at LINE Corp., Japan

220+ million active users, processing millions of msg/s

“Applying Kafka Streams for internal message delivery pipeline”

2017

today
EXAMPLE: DEPLOY WITH docker
REMEMBER?

DEMO TIME!

CONFLUENT "KAFKA MUSIC" APP

docker • confluent
WORD COUNT APP: DEPLOYING
RUNS EVERYWHERE (JVM)

...and more...
SAY "WORDCOUNT" ONE MORE TIME...
FAULT TOLERANCE
EXACTLY-ONCE PROCESSING

Great for: Everyone 😁

Notably: Finance, ...

(+ WordCount)

*Available in Apache Kafka 0.11 (June 2017)
ELASTICITY

1. APP
2. APP
3. APP

SHRINK

confluent
SCALABILITY (EX: JOINS)

1. APP
2. APP
3. APP

λM/sec

LOOKUP
EXTERNAL DATABASE

confluent
SCALABILITY (EX.: JOINS)

\[ \lambda M/\text{sec} \]

1. APP
2. APP
3. APP
SCALABILITY (EX: JOINS)
INTERACTIVE QUERIES

(E.G. FOR MICROSERVICES)

APP

QUERY

"LATEST COUNT FOR HELLO?"

OTHER APP
$ curl -sXGET http://localhost:7070/kafka-music/charts/top-five
[
{
   "artist": "Subhumans",
   "album": "Live In A Dive",
   "name": "All Gone Dead",
   "plays": 126
},
{
   "artist": "Wheres The Pope?",
   "album": "PSI",
   "name": "Fear Of God",
   "plays": 115
},
...
]
INTERACTIVE QUERIES

1

APP

OTHER APP

APP

INTERACTIVE QUERIES

OTHER APP

...and more...
REMEMBER Z?
RECAP
AS APP DEVs WE WANT TO

BUILD APPS

(NOT INFRA)
Kafka Streams API

To build awesome applications that power your core business

(We're hiring)

https://kafka.apache.org/documentation/streams
https://www.confluent.io/downloads/
http://docs.confluent.io/current/streams/
Kafka Summit San Francisco

August 28, 2017
www.kafka-summit.org
Discount code: kafcom17
Use the Apache Kafka community discount code to get $50 off

Questions? We’re at booth #317 in the Exhibition Hall.