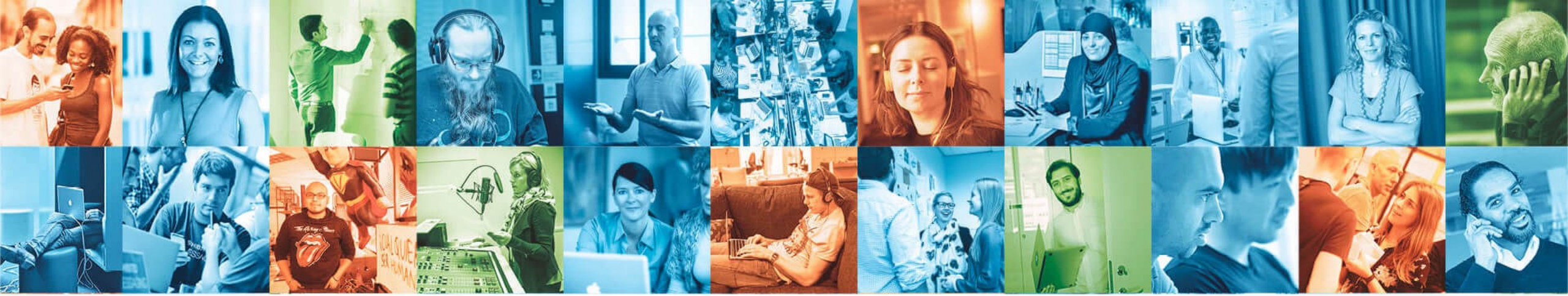


TRANSLATING 800 MILLION EVENTS FROM IP TO COORDINATES EACH DAY USING KAFKA STREAMS

Berlin Buzzwords 2018

Håkon Åmdal, 11.06.2018

SCHIBSTED MEDIA GROUP

A: I USE KAFKA, BUT I NEED TO KNOW IF KAFKA STREAMS IS SOMETHING FOR ME

A: I USE KAFKA, BUT I NEED TO KNOW IF KAFKA STREAMS IS SOMETHING FOR ME

B: I USE OTHER STREAMING FRAMEWORKS, LIKE FLINK, SPARK STREAMING, OR SAMZA, BUT I'M NOT PARTICULARLY FAMILIAR WITH KAFKA STREAMS

A: I USE KAFKA, BUT I NEED TO KNOW IF KAFKA STREAMS IS SOMETHING FOR ME

B: I USE OTHER STREAMING FRAMEWORKS, LIKE FLINK, SPARK STREAMING, OR SAMZA, BUT I'M NOT PARTICULARLY FAMILIAR WITH KAFKA STREAMS

C: I USE KAFKA STREAMS, AND I'M HERE TO LEARN HOW "YOU GUYS DID IT"

A: I USE KAFKA, BUT I NEED TO KNOW IF KAFKA STREAMS IS SOMETHING FOR ME

B: I USE OTHER STREAMING FRAMEWORKS, LIKE FLINK, SPARK STREAMING, OR SAMZA, BUT I'M NOT PARTICULARLY FAMILIAR WITH KAFKA STREAMS

C: I USE KAFKA STREAMS, AND I'M HERE TO LEARN HOW "YOU GUYS DID IT"

D: I'M NEW TO KAFKA

0. THE BACKGROUND STORY

1. THE CHALLENGE

2. THE TECHNOLOGY

3. THE APPLICATION

4. THE DEPLOYMENT

5. THE CONCLUSION

SCHIBSTED, THE DATA PLATFORM AND TARGETED ADVERTISING

HÅKON ÅMDAL

DATA ENGINEER IN SCHIBSTED

WORKS IN THE DATA PLATFORM TEAM

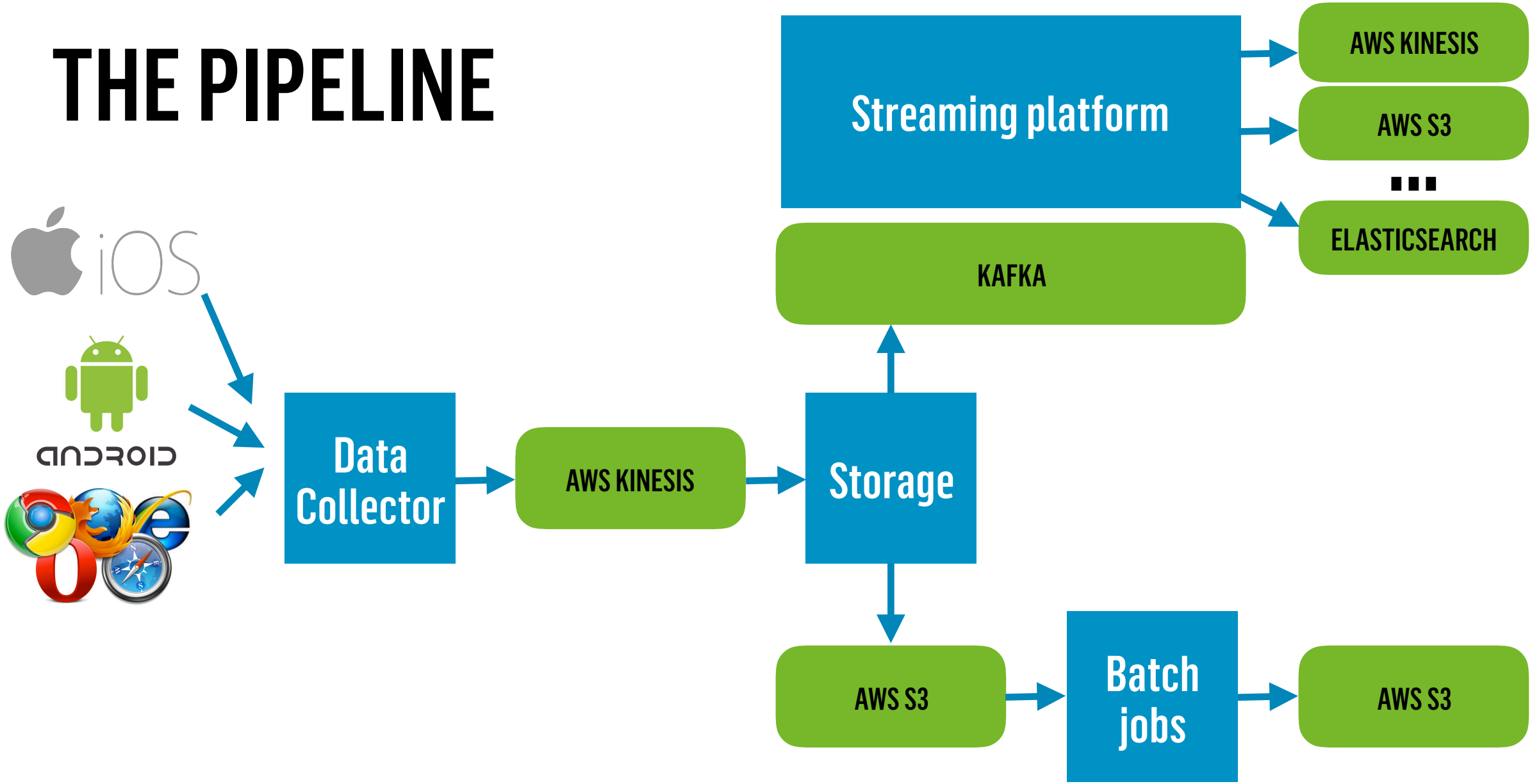
FIRST TIME AT BERLIN BUZZWORDS

(LOVES STREAM PROCESSING)

SCHIBSTED

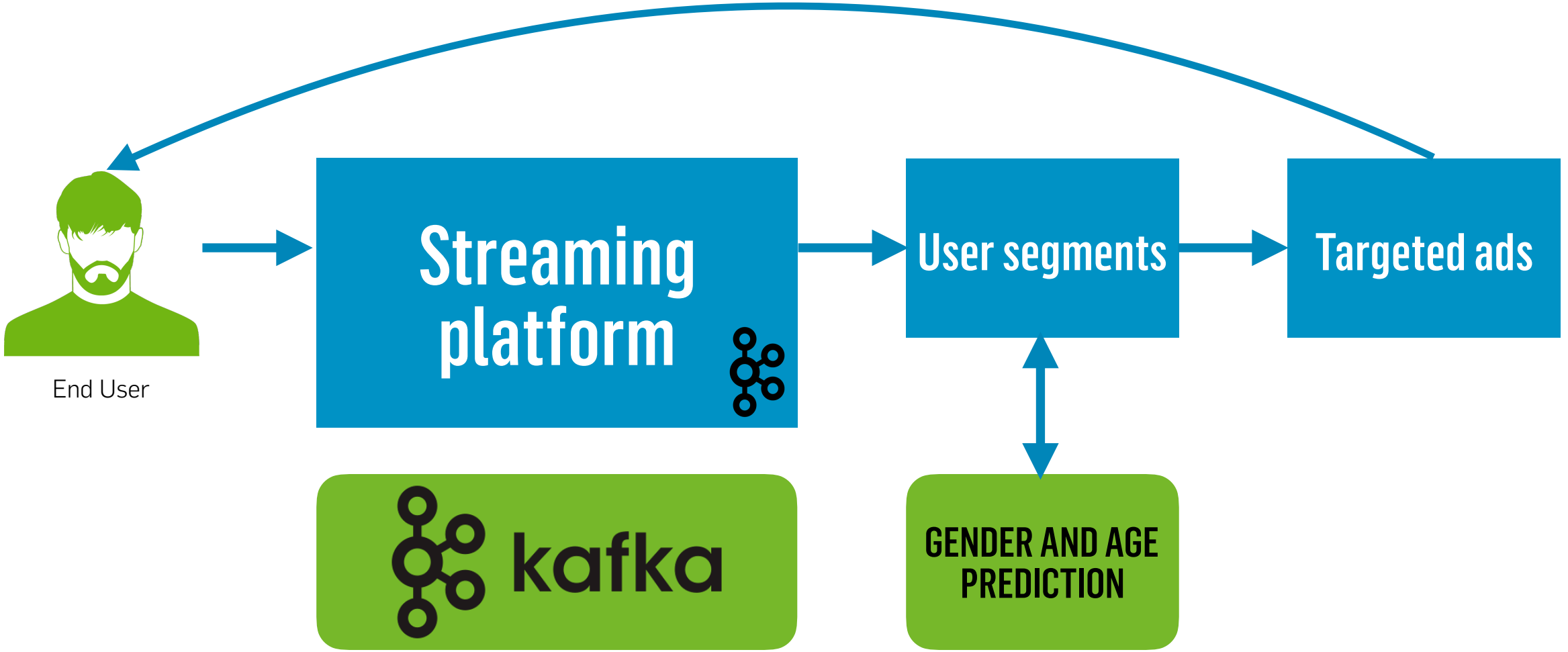

MEDIAHOUSES
Aftenposten
VG
bt:no
AFTONBLADET
SvD
etc...
MARKETPLACES
FINN
WILLHABEN.AT
blocket.se
leboncoin
vibbo
subito
DoneDeal.ie
bomnegócio.com
compre e venda perto de você
etc...
GROWTH
Compricer
Prisjakt
hitta.se
Lendo
Tripwell
let's deal
sh
shpock
etc...

THE PIPELINE

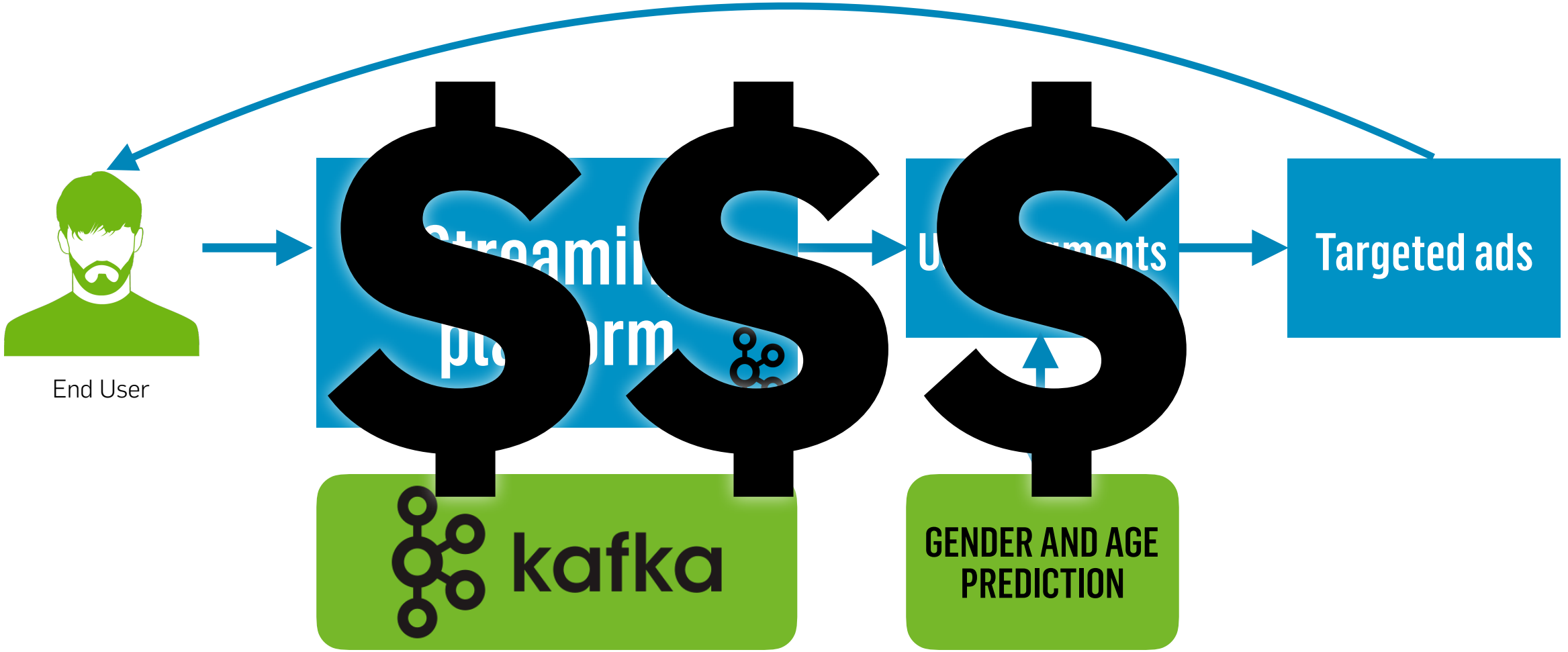


771.17M

TARGETED ADVERTISING IN SCHIBSTED



TARGETED ADVERTISING IN SCHIBSTED



0. THE BACKGROUND STORY

1. THE CHALLENGE

2. THE TECHNOLOGY

3. THE APPLICATION

4. THE DEPLOYMENT

5. THE CONCLUSION

TARGET USERS BASED ON LOCATION

THE LOCATION API

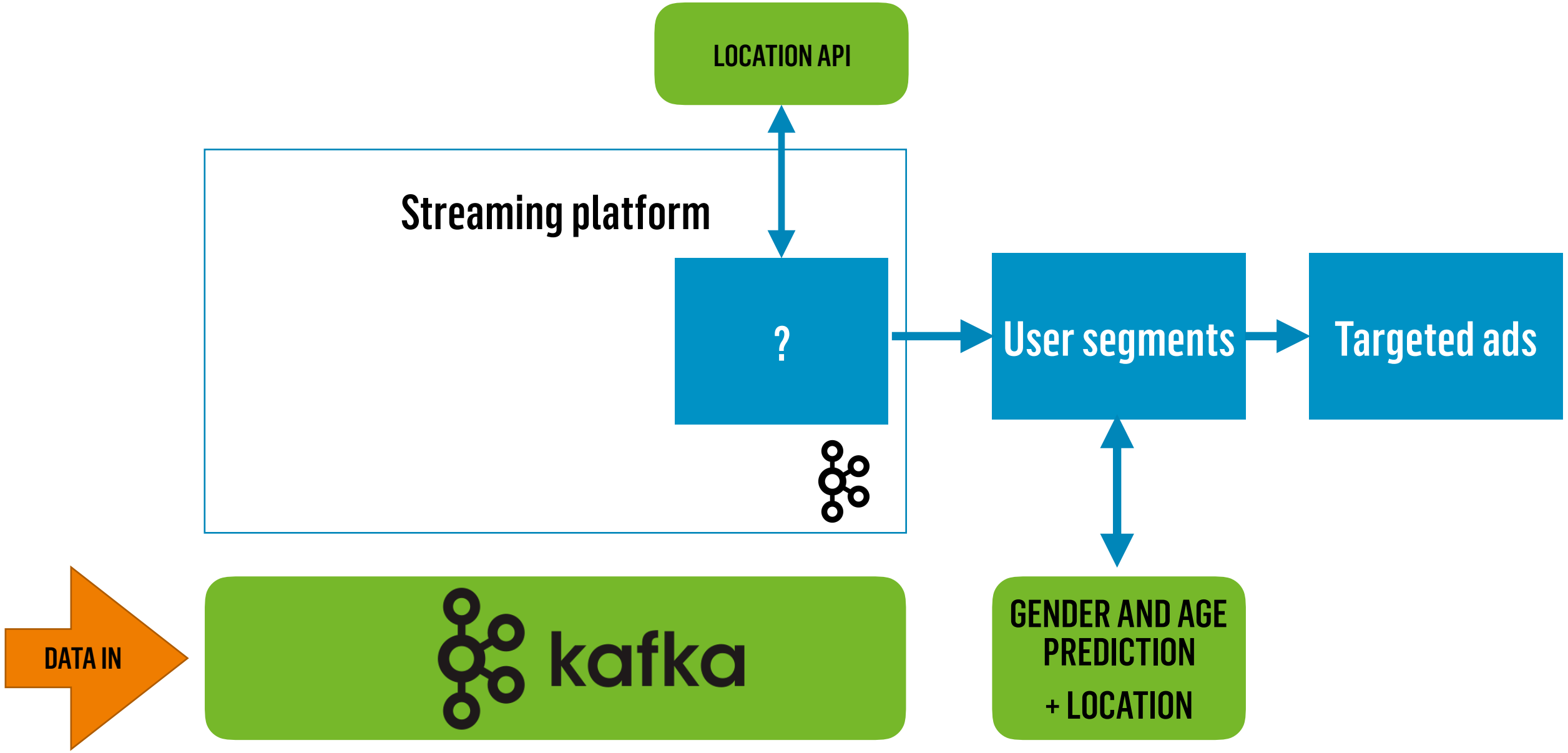
INPUT

```
{
    "ip": "158.36.105.93"
}
```

```
{
    "ip": "158.36.105.93",
    "coordinates": {
        "latitude": 59.398531,
        "longitude": 9.782724
    }
}
```

OUTPUT

```
{
    "geoLocation": {
        "components": {
            "POSTCODE": "NO-P-0168",
            "LEVEL_2": "NO-0301",
            "LEVEL_1": "NO-03",
            "COUNTRY": "578"
        },
        "coordinates": {
            "latitude": 59.92660140991211,
            "longitude": 10.733699798583984
        }
    }
}
```


LOCATION API
Streaming platform
?
User segments
Targeted ads
DATA IN
kafka
GENDER AND AGE PREDICTION
+ LOCATION

REQUIREMENTS

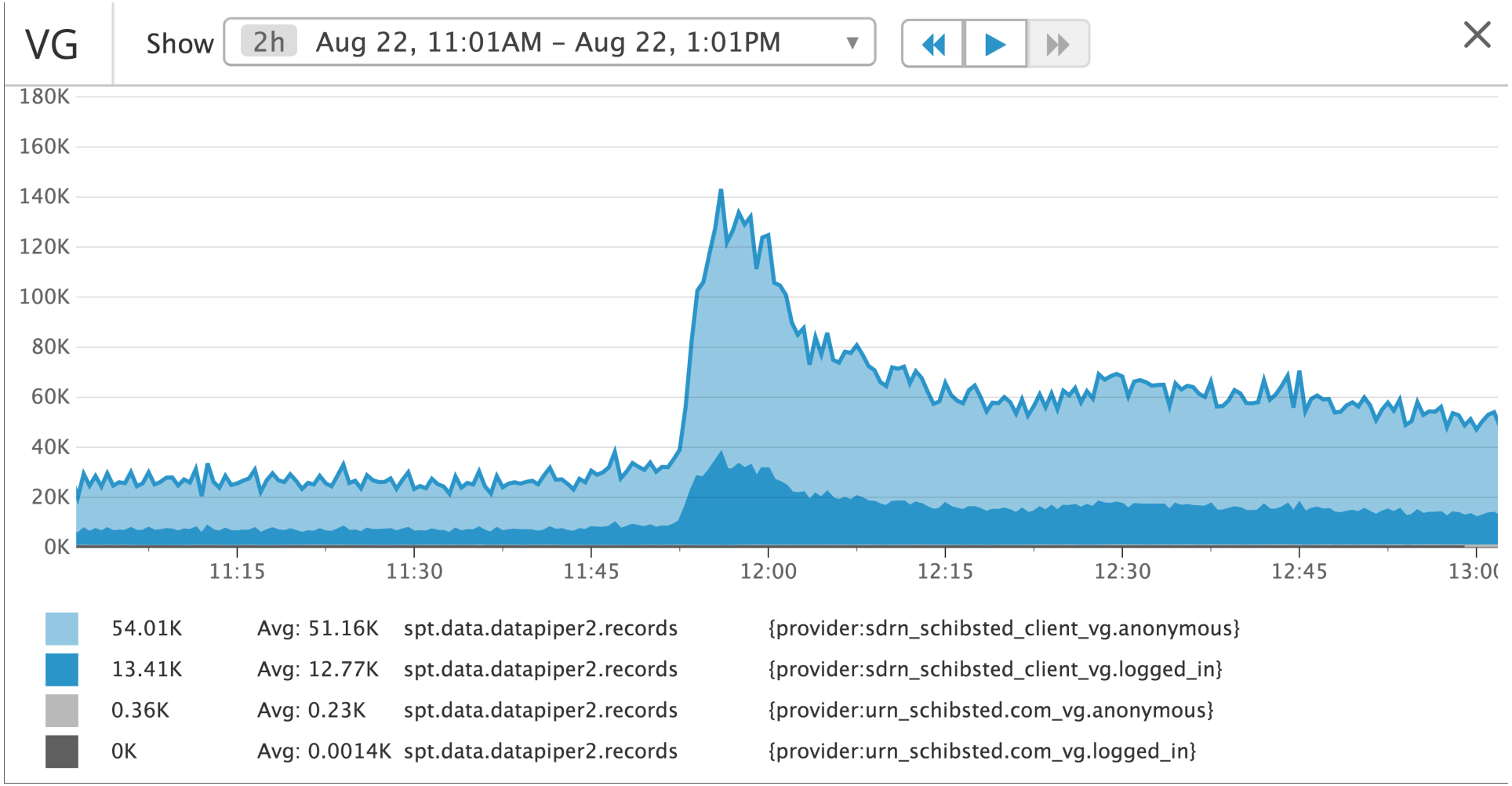
Latency - Events must arrive in time

Completeness - All events must have location*

CHALLENGE: EVENT VOLUME

771.17M

CHALLENGE: EVENT SPIKES



LOCATION API CHALLENGES

Bulking requests

Back pressure

Slow

Fails

0. THE BACKGROUND STORY

1. THE CHALLENGE

2. THE TECHNOLOGY

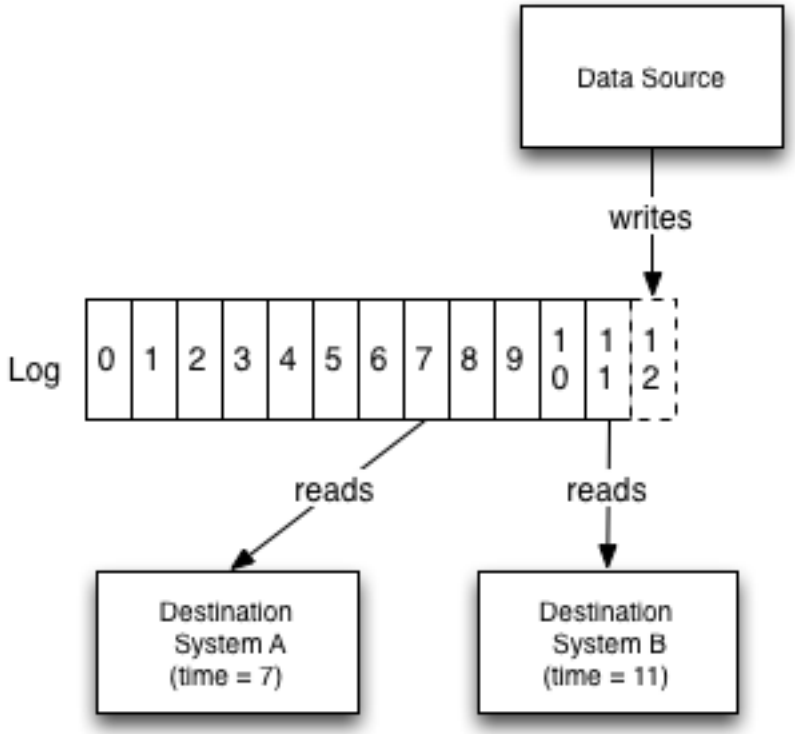
3. THE APPLICATION

4. THE DEPLOYMENT

5. THE CONCLUSION

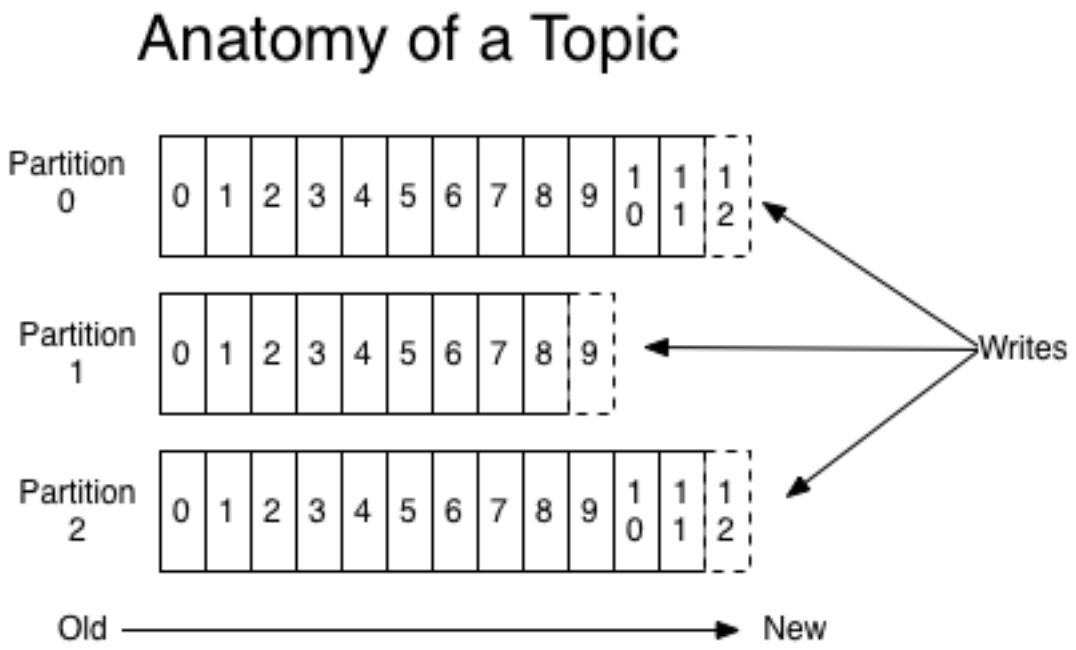
kafka

APACHE kafka



https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying

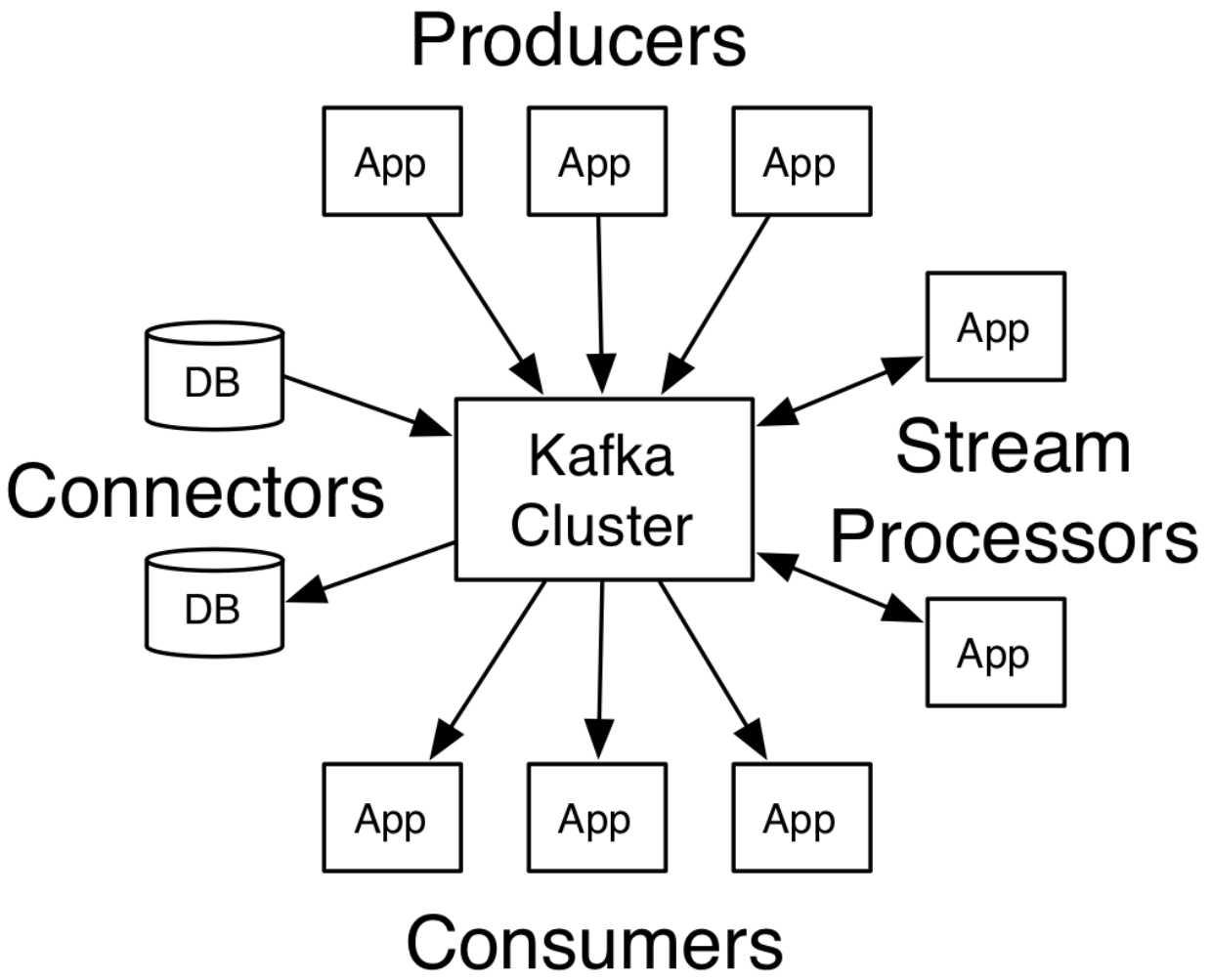
APACHE kafka



http://kafka.apache.org/documentation.html

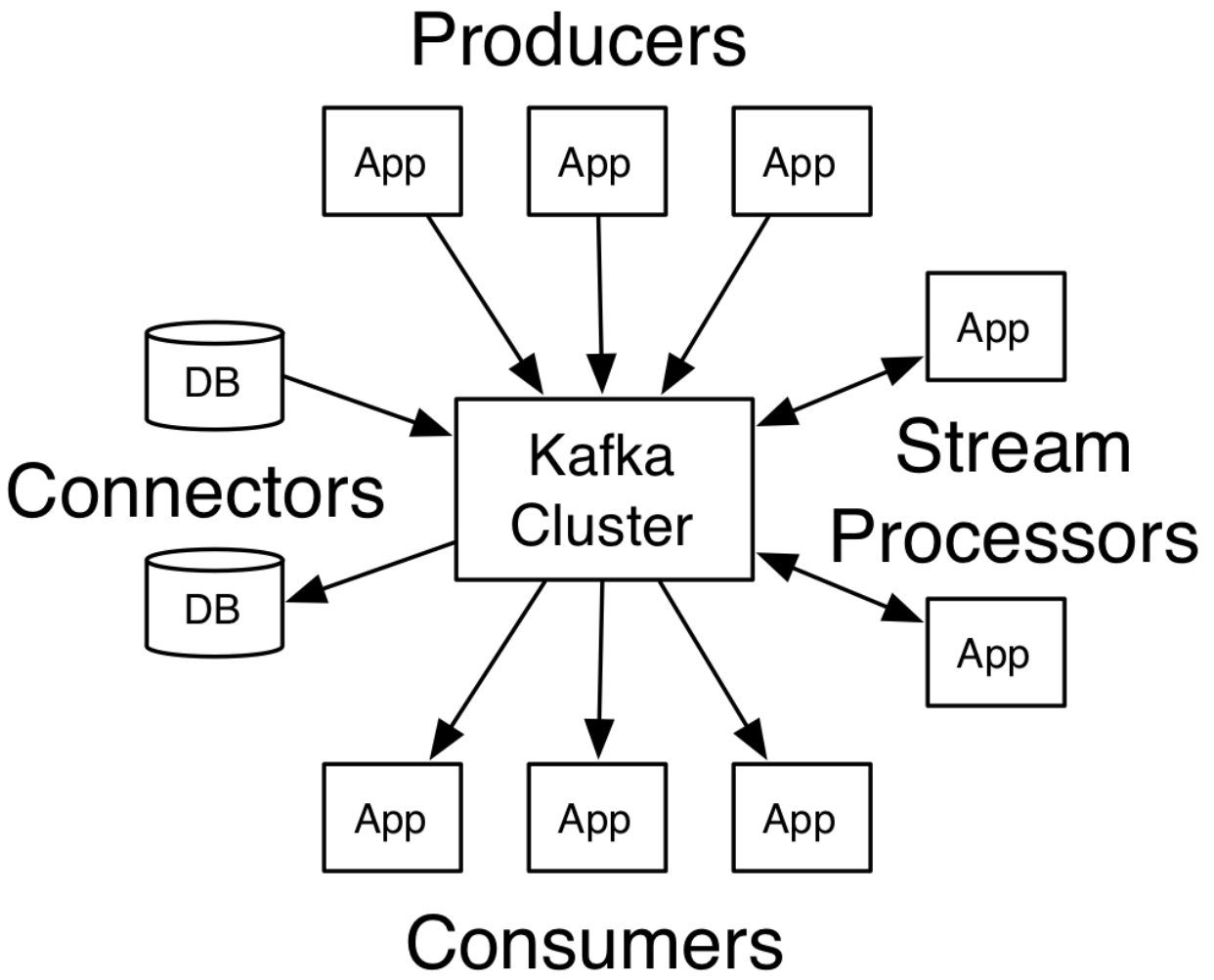
(KEY[ANY], VALUE[ANY], TIMESTAMP)

APACHE kafka



http://kafka.apache.org/documentation.html

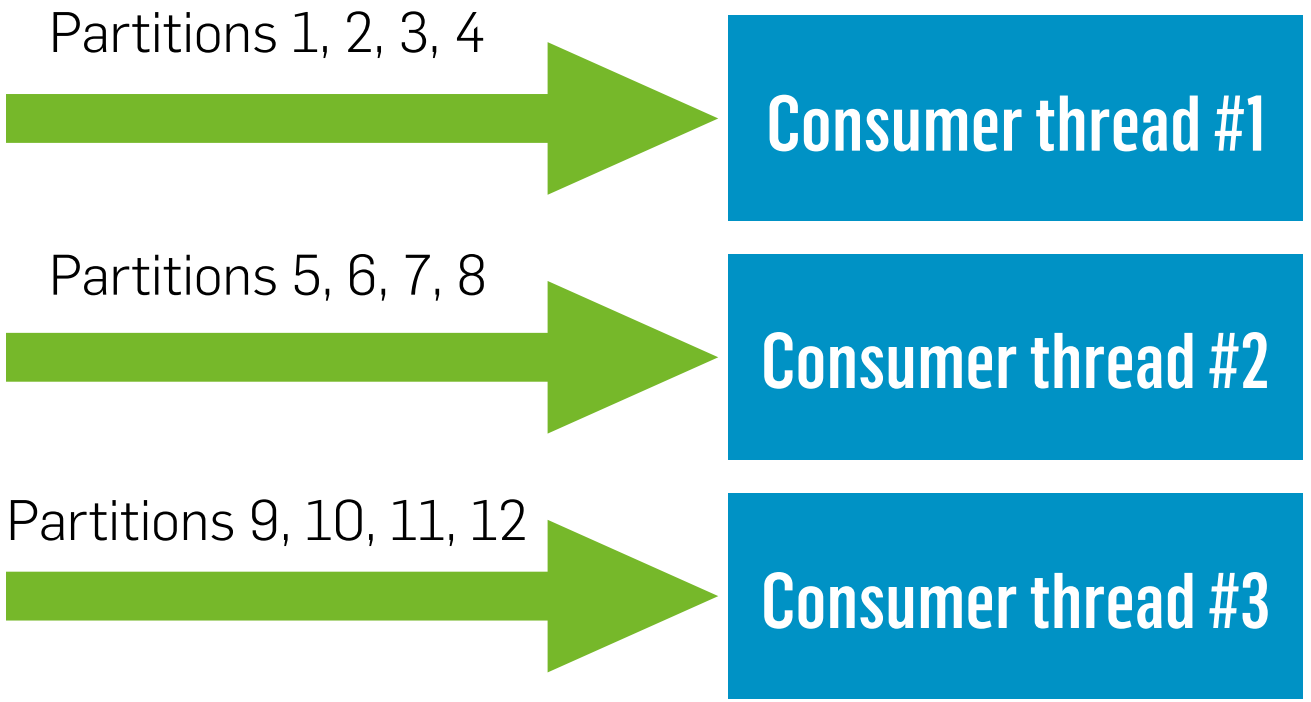
APACHE kafka



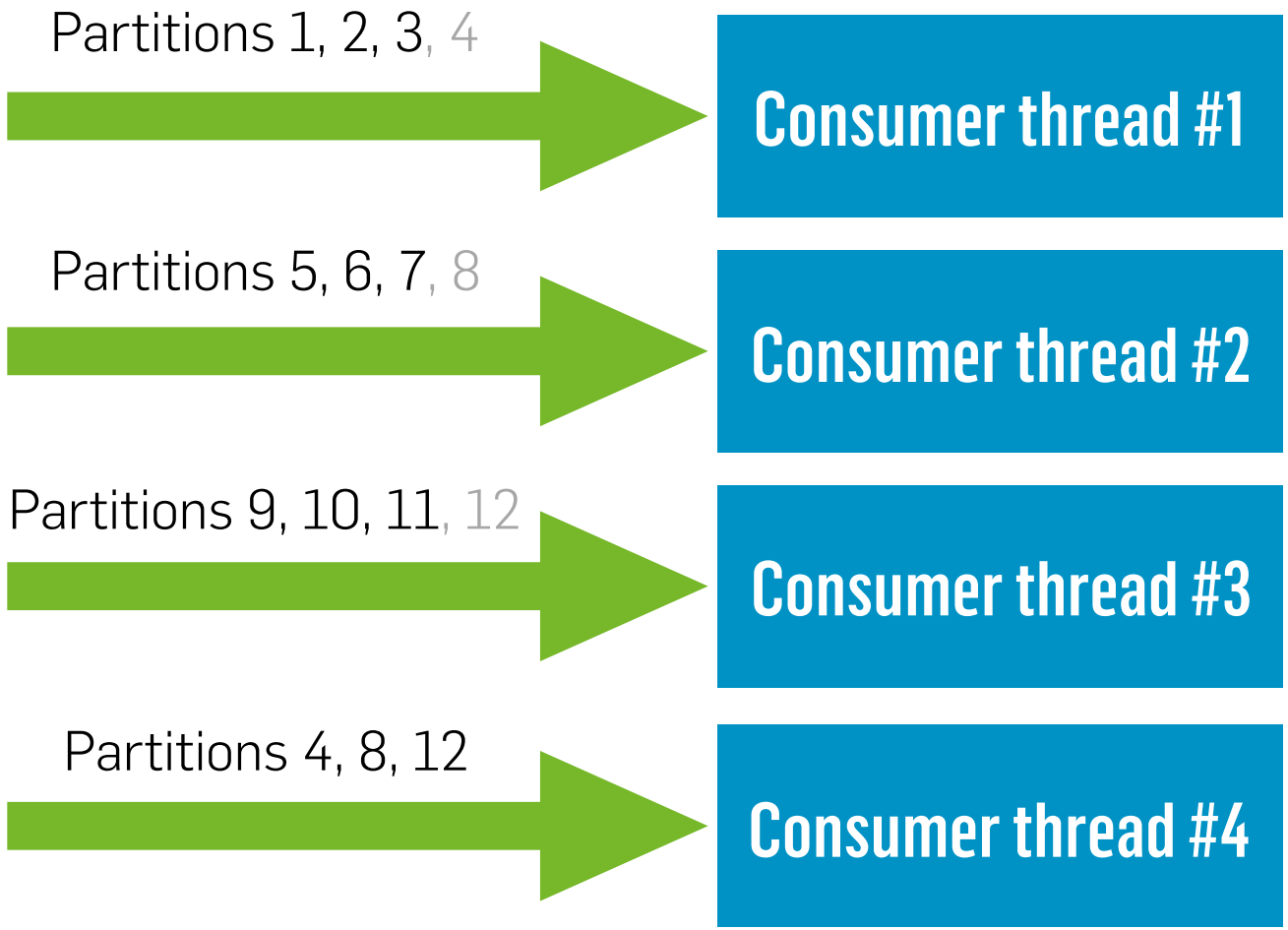
Kafka Streams

http://kafka.apache.org/documentation.html

KAFKA CONSUMER GROUPS



KAFKA CONSUMER GROUPS



0. THE BACKGROUND STORY

1. THE CHALLENGE

2. THE TECHNOLOGY

3. THE APPLICATION

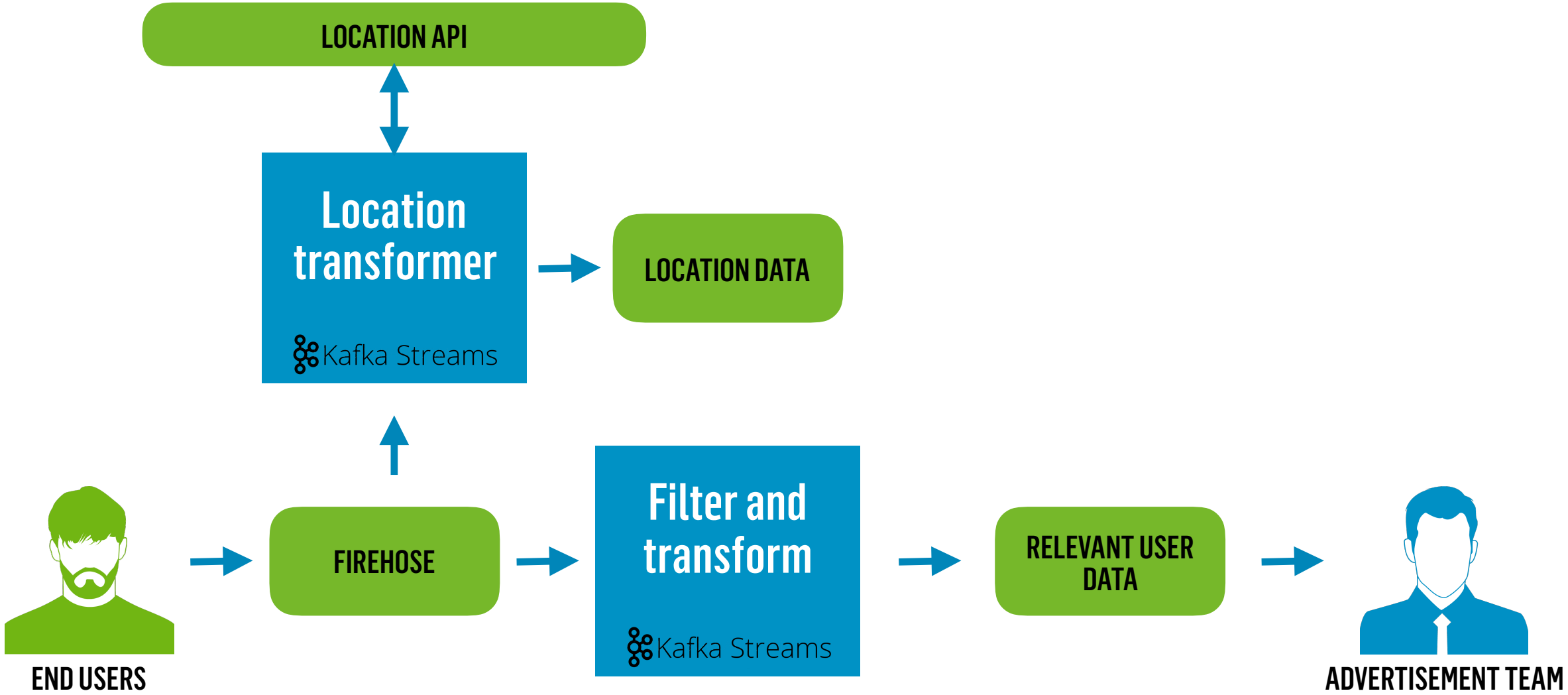
4. THE DEPLOYMENT

5. THE CONCLUSION

ORIGINAL


FIREHOSE
Filter and transform
Kafka Streams
RELEVANT USER DATA
END USERS
ADVERTISEMENT TEAM

ADDING LOCATION DATA



SETTING UP THE APPLICATION

```
val config = Map(
  StreamsConfig.APPLICATION_ID_CONFIG -> "LocationEnricher",
  StreamsConfig.BOOTSTRAP_SERVERS_CONFIG -> "kafkabroker:9092"
)

val builder = new KStreamBuilder()
// TODO: Create topology

val streams: KafkaStreams = new KafkaStreams(builder, new StreamsConfig(config.asJava))
streams.start()

Runtime.getRuntime.addShutdownHook(new Thread(() => {
  streams.close(10, TimeUnit.SECONDS)
}))
```

READING THE INPUT #1

```
val builder = new KStreamBuilder()

val stringSerde: Serde[String] = Serdes.String()
val jsonSerde: Serde[JsonNode] = new JsonNodeSerde

val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, jsonSerde, "Firehose")
```

READING THE INPUT #1

```
class JsonSerde extends Serde[JsonNode] {

  val mapper = new ObjectMapper with ScalaObjectMapper
  mapper.registerModule(DefaultScalaModule)

  override def deserializer() = new Deserializer[JsonNode] {

    override def deserialize(topic: String, data: Array[Byte]): JsonNode = mapper.readValue(data)

    override def configure(configs: util.Map[String, _], isKey: Boolean): Unit = {/* ... */}
    override def close(): Unit = {/* ... */}
  }
  override def serializer() = new Serializer[JsonNode] {

    override def serialize(topic: String, data: JsonNode) = mapper.writeValueAsBytes(data)

    override def configure(configs: util.Map[String, _], isKey: Boolean): Unit = {/* ... */}
    override def close(): Unit = {/* ... */}
  }
  override def configure(configs: util.Map[String, _], isKey: Boolean): Unit = {/* ... */}
  override def close(): Unit = {/* ... */}
}
```

READING THE INPUT #1

```
val builder = new KStreamBuilder()

val stringSerde: Serde[String] = Serdes.String()
val jsonSerde: Serde[JsonNode] = new JsonNodeSerde

val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, jsonSerde, "Firehose")
```

```
2017-09-03 14:17:52,865 ERROR [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:122] - Uncaught exception: Thread
  Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1 stopped unexpectedly after Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
org.apache.kafka.streams.errors.StreamsException: Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:46)
    at org.apache.kafka.streams.processor.internals.RecordQueue.addRawRecords(RecordQueue.java:84)
    at org.apache.kafka.streams.processor.internals.PartitionGroup.addRawRecords(PartitionGroup.java:117)
    at org.apache.kafka.streams.processor.internals.StreamTask.addRecords(StreamTask.java:464)
    at org.apache.kafka.streams.processor.internals.StreamThread.addRecordsToTasks(StreamThread.java:650)
    at org.apache.kafka.streams.processor.internals.StreamThread.runLoop(StreamThread.java:556)
    at org.apache.kafka.streams.processor.internals.StreamThread.run(StreamThread.java:527)
Caused by: com.fasterxml.jackson.core.JsonParseException: Unrecognized token 'Hoo': was expecting ('true', 'false' or 'null')
 at [Source: [B@22fe4830; line: 1, column: 7]
    at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1702)
    at com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:558)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._reportInvalidToken(UTF8StreamJsonParser.java:3528)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleUnexpectedValue(UTF8StreamJsonParser.java:2686)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._nextTokenNotInObject(UTF8StreamJsonParser.java:878)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:772)
    at com.fasterxml.jackson.databind.ObjectMapper._initForReading(ObjectMapper.java:3850)
    at com.fasterxml.jackson.databind.ObjectMapper._readMapAndClose(ObjectMapper.java:3799)
    at com.fasterxml.jackson.databind.ObjectMapper.readTree(ObjectMapper.java:2420)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:26)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:25)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:65)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:55)
    at org.apache.kafka.streams.processor.internals.SourceNode.deserializeValue(SourceNode.java:56)
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:44)
    ... 6 common frames omitted
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:133] - Restarting stream processing (attempt 1)
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
  [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from RUNNING to PENDING_SHUTDOWN.
2017-09-03 14:17:52,866 INFO [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:900] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
  Informed thread to shut down
2017-09-03 14:17:52,866 WARN [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:978] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
  Unexpected state transition from DEAD to PENDING_SHUTDOWN.
2017-09-03 14:18:52,866 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
  [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from PENDING_SHUTDOWN to NOT_RUNNING.
2017-09-03 14:18:52,866 WARN [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:141] - Timeout waiting for stream processing to shut down
2017-09-03 14:18:52,867 INFO [kafka-streams-close-thread] o.a.k.streams.KafkaStreams [KafkaStreams.java:514] - stream-client [Jord-e4d7678b-9804-4570-8735-672aeaddb284] Stopped Kafka
  Streams process.
```

```
2017-09-03 14:17:52,865 ERROR [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:122] - Uncaught exception: Thread
 Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1 stopped unexpectedly after Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
org.apache.kafka.streams.errors.StreamsException: Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:46)
    at org.apache.kafka.streams.processor.internals.RecordQueue.addRawRecords(RecordQueue.java:84)
    at org.apache.kafka.streams.processor.internals.PartitionGroup.addRawRecords(PartitionGroup.java:117)
    at org.apache.kafka.streams.processor.internals.StreamTask.addRecords(StreamTask.java:464)
    at org.apache.kafka.streams.processor.internals.StreamThread.addRecordsToTasks(StreamThread.java:650)
    at org.apache.kafka.streams.processor.internals.StreamThread.runLoop(StreamThread.java:556)
    at org.apache.kafka.streams.processor.internals.StreamThread.run(StreamThread.java:527)
Caused by: com.fasterxml.jackson.core.JsonParseException: Unrecognized token 'Hoo': was expecting ('true', 'false' or 'null')
 at [Source: [B@22fe4830; line: 1, column: 7]
    at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1702)
    at com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:558)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._reportInvalidToken(UTF8StreamJsonParser.java:3528)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleUnexpectedValue(UTF8StreamJsonParser.java:2686)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._nextTokenNotInObject(UTF8StreamJsonParser.java:878)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:772)
    at com.fasterxml.jackson.databind.ObjectMapper._initForReading(ObjectMapper.java:3850)
    at com.fasterxml.jackson.databind.ObjectMapper._readMapAndClose(ObjectMapper.java:3799)
    at com.fasterxml.jackson.databind.ObjectMapper.readTree(ObjectMapper.java:2420)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:26)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:25)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:65)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:55)
    at org.apache.kafka.streams.processor.internals.SourceNode.deserializeValue(SourceNode.java:56)
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:44)
    ... 6 common frames omitted
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:133] - Restarting stream processing (attempt 1)
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from RUNNING to PENDING_SHUTDOWN.
2017-09-03 14:17:52,866 INFO [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:900] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Informed thread to shut down
2017-09-03 14:17:52,866 WARN [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:978] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Unexpected state transition from DEAD to PENDING_SHUTDOWN.
2017-09-03 14:18:52,866 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from PENDING_SHUTDOWN to NOT_RUNNING.
2017-09-03 14:18:52,866 WARN [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:141] - Timeout waiting for stream processing to shut down
2017-09-03 14:18:52,867 INFO [kafka-streams-close-thread] o.a.k.streams.KafkaStreams [KafkaStreams.java:514] - stream-client [Jord-e4d7678b-9804-4570-8735-672aeaddb284] Stopped Kafka
 Streams process.
```

```
2017-09-03 14:17:52,865 ERROR [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:122] - Uncaught exception: Thread
 Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1 stopped unexpectedly after Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
org.apache.kafka.streams.errors.StreamsException: Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:46)
    at org.apache.kafka.streams.processor.internals.RecordQueue.addRawRecords(RecordQueue.java:84)
    at org.apache.kafka.streams.processor.internals.PartitionGroup.addRawRecords(PartitionGroup.java:117)
    at org.apache.kafka.streams.processor.internals.StreamTask.addRecords(StreamTask.java:464)
    at org.apache.kafka.streams.processor.internals.StreamThread.addRecordsToTasks(StreamThread.java:650)
    at org.apache.kafka.streams.processor.internals.StreamThread.runLoop(StreamThread.java:556)
    at org.apache.kafka.streams.processor.internals.StreamThread.run(StreamThread.java:527)
Caused by: com.fasterxml.jackson.core.JsonParseException: Unrecognized token 'Hoo': was expecting ('true', 'false' or 'null')
 at [Source: [B@22fe4830; line: 1, column: 7]
    at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1702)
    at com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:558)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._reportInvalidToken(UTF8StreamJsonParser.java:3528)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleUnexpectedValue(UTF8StreamJsonParser.java:2686)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._nextTokenNotInObject(UTF8StreamJsonParser.java:878)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:772)
    at com.fasterxml.jackson.databind.ObjectMapper._initForReading(ObjectMapper.java:3850)
    at com.fasterxml.jackson.databind.ObjectMapper._readMapAndClose(ObjectMapper.java:3799)
    at com.fasterxml.jackson.databind.ObjectMapper.readTree(ObjectMapper.java:2420)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:26)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:25)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:65)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:55)
    at org.apache.kafka.streams.processor.internals.SourceNode.deserializeValue(SourceNode.java:56)
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:44)
    ... 6 common frames omitted
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:133] - Restarting stream processing (attempt 1)
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from RUNNING to PENDING_SHUTDOWN.
2017-09-03 14:17:52,866 INFO [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:900] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Informed thread to shut down
2017-09-03 14:17:52,866 WARN [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:978] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Unexpected state transition from DEAD to PENDING_SHUTDOWN.
2017-09-03 14:18:52,866 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from PENDING_SHUTDOWN to NOT_RUNNING.
2017-09-03 14:18:52,866 WARN [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:141] - Timeout waiting for stream processing to shut down
2017-09-03 14:18:52,867 INFO [kafka-streams-close-thread] o.a.k.streams.KafkaStreams [KafkaStreams.java:514] - stream-client [Jord-e4d7678b-9804-4570-8735-672aeaddb284] Stopped Kafka
 Streams process.
```

```
2017-09-03 14:17:52,865 ERROR [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:122] - Uncaught exception: Thread
 Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1 stopped unexpectedly after Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
org.apache.kafka.streams.errors.StreamsException: Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:46)
    at org.apache.kafka.streams.processor.internals.RecordQueue.addRawRecords(RecordQueue.java:84)
    at org.apache.kafka.streams.processor.internals.PartitionGroup.addRawRecords(PartitionGroup.java:117)
    at org.apache.kafka.streams.processor.internals.StreamTask.addRecords(StreamTask.java:464)
    at org.apache.kafka.streams.processor.internals.StreamThread.addRecordsToTasks(StreamThread.java:650)
    at org.apache.kafka.streams.processor.internals.StreamThread.runLoop(StreamThread.java:556)
    at org.apache.kafka.streams.processor.internals.StreamThread.run(StreamThread.java:527)
Caused by: com.fasterxml.jackson.core.JsonParseException: Unrecognized token 'Hoo': was expecting ('true', 'false' or 'null')
 at [Source: [B@22fe4830; line: 1, column: 7]
    at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1702)
    at com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:558)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._reportInvalidToken(UTF8StreamJsonParser.java:3528)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleUnexpectedValue(UTF8StreamJsonParser.java:2686)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._nextTokenNotInObject(UTF8StreamJsonParser.java:878)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:772)
    at com.fasterxml.jackson.databind.ObjectMapper._initForReading(ObjectMapper.java:3850)
    at com.fasterxml.jackson.databind.ObjectMapper._readMapAndClose(ObjectMapper.java:3799)
    at com.fasterxml.jackson.databind.ObjectMapper.readTree(ObjectMapper.java:2420)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:26)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:25)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:65)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:55)
    at org.apache.kafka.streams.processor.internals.SourceNode.deserializeValue(SourceNode.java:56)
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:44)
    ... 6 common frames omitted
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:133] - Restarting stream processing (attempt 1)
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from RUNNING to PENDING_SHUTDOWN.
2017-09-03 14:17:52,866 INFO [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:900] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Informed thread to shut down
2017-09-03 14:17:52,866 WARN [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:978] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Unexpected state transition from DEAD to PENDING_SHUTDOWN.
2017-09-03 14:18:52,866 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from PENDING_SHUTDOWN to NOT_RUNNING.
2017-09-03 14:18:52,866 WARN [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:141] - Timeout waiting for stream processing to shut down
2017-09-03 14:18:52,867 INFO [kafka-streams-close-thread] o.a.k.streams.KafkaStreams [KafkaStreams.java:514] - stream-client [Jord-e4d7678b-9804-4570-8735-672aeaddb284] Stopped Kafka
 Streams process.
```

```
2017-09-03 14:17:52,865 ERROR [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:122] - Uncaught exception: Thread
 Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1 stopped unexpectedly after Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
org.apache.kafka.streams.errors.StreamsException: Failed to deserialize value for record. topic=PulseEvent, partition=6, offset=284
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:46)
    at org.apache.kafka.streams.processor.internals.RecordQueue.addRawRecords(RecordQueue.java:84)
    at org.apache.kafka.streams.processor.internals.PartitionGroup.addRawRecords(PartitionGroup.java:117)
    at org.apache.kafka.streams.processor.internals.StreamTask.addRecords(StreamTask.java:464)
    at org.apache.kafka.streams.processor.internals.StreamThread.addRecordsToTasks(StreamThread.java:650)
    at org.apache.kafka.streams.processor.internals.StreamThread.runLoop(StreamThread.java:556)
    at org.apache.kafka.streams.processor.internals.StreamThread.run(StreamThread.java:527)
Caused by: com.fasterxml.jackson.core.JsonParseException: Unrecognized token 'Hoo': was expecting ('true', 'false' or 'null')
 at [Source: [B@22fe4830; line: 1, column: 7]
    at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1702)
    at com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:558)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._reportInvalidToken(UTF8StreamJsonParser.java:3528)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleUnexpectedValue(UTF8StreamJsonParser.java:2686)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser._nextTokenNotInObject(UTF8StreamJsonParser.java:878)
    at com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:772)
    at com.fasterxml.jackson.databind.ObjectMapper._initForReading(ObjectMapper.java:3850)
    at com.fasterxml.jackson.databind.ObjectMapper._readMapAndClose(ObjectMapper.java:3799)
    at com.fasterxml.jackson.databind.ObjectMapper.readTree(ObjectMapper.java:2420)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:26)
    at com.schibsted.spt.data.yggdrasil.common.serdes.JsonNodeSerde$JsonNodeDeserializer.deserialize(JsonNodeSerde.scala:25)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:65)
    at org.apache.kafka.common.serialization.ExtendedDeserializer$Wrapper.deserialize(ExtendedDeserializer.java:55)
    at org.apache.kafka.streams.processor.internals.SourceNode.deserializeValue(SourceNode.java:56)
    at org.apache.kafka.streams.processor.internals.SourceNodeRecordDeserializer.deserialize(SourceNodeRecordDeserializer.java:44)
    ... 6 common frames omitted
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:133] - Restarting stream processing (attempt 1)
2017-09-03 14:17:52,865 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from RUNNING to PENDING_SHUTDOWN.
2017-09-03 14:17:52,866 INFO [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:900] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Informed thread to shut down
2017-09-03 14:17:52,866 WARN [kafka-streams-close-thread] o.a.k.s.p.i.StreamThread [StreamThread.java:978] - stream-thread [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1]
 Unexpected state transition from DEAD to PENDING_SHUTDOWN.
2017-09-03 14:18:52,866 INFO [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] o.a.k.streams.KafkaStreams [KafkaStreams.java:229] - stream-client
 [Jord-e4d7678b-9804-4570-8735-672aeaddb284] State transition from PENDING_SHUTDOWN to NOT_RUNNING.
2017-09-03 14:18:52,866 WARN [Jord-e4d7678b-9804-4570-8735-672aeaddb284-StreamThread-1] c.s.s.d.y.Yggdrasil [Yggdrasil.scala:141] - Timeout waiting for stream processing to shut down
2017-09-03 14:18:52,867 INFO [kafka-streams-close-thread] o.a.k.streams.KafkaStreams [KafkaStreams.java:514] - stream-client [Jord-e4d7678b-9804-4570-8735-672aeaddb284] Stopped Kafka
 Streams process.
```

READING THE INPUT #2

```
val builder = new KStreamBuilder()

val stringSerde: Serde[String] = Serdes.String()
val jsonSerde = new JsonNodeSerde
val byteSerde: Serde[Array[Byte]] = Serdes.ByteArray()

val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
```

READING THE INPUT #2

```
val builder = new KStreamBuilder()

val stringSerde: Serde[String] = Serdes.String()
val jsonSerde = new JsonNodeSerde
val byteSerde: Serde[Array[Byte]] = Serdes.ByteArray()

val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
  .mapValues[Try[JsonNode]](value => Try(jsonSerde.deserializer.deserialize(null, value))
```

READING THE INPUT #2

```
val builder = new KStreamBuilder()

val stringSerde: Serde[String] = Serdes.String()
val jsonSerde = new JsonNodeSerde
val byteSerde: Serde[Array[Byte]] = Serdes.ByteArray()

val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
  .mapValues[Try[JsonNode]](value => Try(jsonSerde.deserializer.deserialize(null, value)))
  .filter((key, value) => value.isSuccess)
  .mapValues(value => value.get)
```

READING THE INPUT #2

```
val builder = new KStreamBuilder()

val stringSerde: Serde[String] = Serdes.String()
val jsonSerde = new JsonNodeSerde
val byteSerde: Serde[Array[Byte]] = Serdes.ByteArray()

val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
  .flatMapValues[JsonNode] { (value: Array[Byte]) =>
    Try(jsonSerde.deserializer.deserialize(null, value)).toOption.toIterable.asJava
  }
```

TO REQUEST OBJECTS

```
case class LocationAPIRequest(ip: String, coordinates: Option[Coordinates])
```

TO REQUEST OBJECTS

```
case class LocationAPIRequest(ip: String, coordinates: Option[Coordinates])

def jsonToRequest(json: JsonNode): Option[LocationAPIRequest] = {..}
```

TO REQUEST OBJECTS

```
case class LocationAPIRequest(ip: String, coordinates: Option[Coordinates])

def jsonToRequest(json: JsonNode): Option[LocationAPIRequest] = {..}


val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
  .flatMapValues[JsonNode](…)

val locationAPIRequests = input
```

TO REQUEST OBJECTS

```
case class LocationAPIRequest(ip: String, coordinates: Option[Coordinates])

def jsonToRequest(json: JsonNode): Option[LocationAPIRequest] = {..}


val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
  .flatMapValues[JsonNode](…)

val locationAPIRequests = input
  .mapValues[Option[LocationAPIRequest]](jsonToRequest)
```

TO REQUEST OBJECTS

```
case class LocationAPIRequest(ip: String, coordinates: Option[Coordinates])

def jsonToRequest(json: JsonNode): Option[LocationAPIRequest] = {..}


val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
  .flatMapValues[JsonNode](…)

val locationAPIRequests = input
  .mapValues[Option[LocationAPIRequest]](jsonToRequest)
  .peek((key: String, value: Option[LocationAPIRequest]) => {
    if (value.isDefined) {
      metrics.increment("events_processed")
    } else {
      metrics.increment("events_discarded")
    }
  })
```

TO REQUEST OBJECTS

```
case class LocationAPIRequest(ip: String, coordinates: Option[Coordinates])

def jsonToRequest(json: JsonNode): Option[LocationAPIRequest] = {..}


val input: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "Firehose")
  .flatMapValues[JsonNode](…)

val locationAPIRequests: KStream[String, LocationAPIRequest] = input
  .mapValues[Option[LocationAPIRequest]](jsonToRequest)
  .peek((key: String, value: Option[LocationAPIRequest]) => {
    if (value.isDefined) {
      metrics.increment("events_processed")
    } else {
      metrics.increment("events_discarded")
    }
  })
  .flatMapValues[LocationAPIRequest](_.toIterable.asJava)}
```

PRODUCING THE OUTPUT

```
val locationAPIRequests: KStream[String, LocationAPIRequest] = {…}

val locationAPITransformerSupplier = {
  new LocationAPITransformerSupplier[String]()
}

locationAPIRequests
  .transform(locationAPITransformerSupplier)
```

PRODUCING THE OUTPUT

```
val locationAPIRequests: KStream[String, LocationAPIRequest] = {…}

val locationAPITransformerSupplier = {
  new LocationAPITransformerSupplier[String]()
}

locationAPIRequests
  .transform(locationAPITransformerSupplier)
  .peek((key: String, value: Option[_]) => {
    if (value.isDefined) {
      metrics.increment("coordinates_found_for_ip")
    } else {
      metrics.increment("coordinates_not_found_for_ip")
    }
  })
```

PRODUCING THE OUTPUT

```
val locationAPIRequests: KStream[String, LocationAPIRequest] = {…}

val locationAPITransformerSupplier = {
  new LocationAPITransformerSupplier[String]()
}

locationAPIRequests
  .transform(locationAPITransformerSupplier)
  .peek((key: String, value: Option[_]) => {
    if (value.isDefined) {
      metrics.increment("coordinates_found_for_ip")
    } else {
      metrics.increment("coordinates_not_found_for_ip")
    }
  })
  .mapValues[JsonNode](_.map(fromResponseToJson).orNull)
```

PRODUCING THE OUTPUT

```
val locationAPIRequests: KStream[String, LocationAPIRequest] = {…}

val locationAPITransformerSupplier = {
  new LocationAPITransformerSupplier[String]()
}

locationAPIRequests
  .transform(locationAPITransformerSupplier)
  .peek((key: String, value: Option[_]) => {
    if (value.isDefined) {
      metrics.increment(“coordinates_found_for_ip”)
    } else {
      metrics.increment(“coordinates_not_found_for_ip”)
    }
  })
  .mapValues[JsonNode](_.map(fromResponseToJson).orNull)
  .to(stringSerde, jsonSerde, "LocationData")
```

PRODUCING THE OUTPUT

```
val locationAPIRequests: KStream[String, LocationAPIRequest] = {…}

val locationAPITransformerSupplier = {
  new LocationAPITransformerSupplier[String]()
}

locationAPIRequests
  .transform(locationAPITransformerSupplier)
  .peek((key: String, value: Option[_]) => {
    if (value.isDefined) {
      metrics.increment("coordinates_found_for_ip")
    } else {
      metrics.increment("coordinates_not_found_for_ip")
    }
  })
  .mapValues[JsonNode](_.map(fromResponseToJson).orNull)
  .to(stringSerde, jsonSerde, "LocationData")
```

RECAP: KAFKA STREAMS DSL

```
builder
  .stream(stringSerde, byteSerde, "Firehose")
  .flatMapValues[JsonNode] { (value: Array[Byte]) =>
      Try(jsonSerde.deserializer.deserialize(null, value)).toOption.toIterable.asJava
  }
  .flatMapValues[LocationAPIRequest](jsonToRequest(_).toIterable.asJava)
  .transform(locationAPITransformerSupplier)
  .mapValues[JsonNode](_.map(fromReverseLocationAPIResponseToJson).orNull)
  .to(stringSerde, jsonSerde, "LocationData")
```

PROCESSOR API

```
public interface Transformer<K, V, R> {

    void init(final ProcessorContext context);


    R transform(final K key, final V value);


    R punctuate(final long timestamp);


    void close();

}
```

LOCATION API TRANSFORMER

```
class LocationAPITransformer extends
  Transformer[String, LocationAPIRequest,
    KeyValue[String, Option[LocationAPIResponse]]] {

  var context: ProcessorContext = _
  val buffer = new ArrayBuffer[(String, LocationAPIRequest)]()
  val locationAPIClient: LocationAPIClient = {...}

  override def init(context: ProcessorContext): Unit = {
    this.context = context
    this.context.schedule(500)
  }

  override def transform(key: String, value: LocationAPIRequest) = {...}

  override def close(): Unit = {}
}
```

LOCATION API TRANSFORMER

```
override def transform(key: String, value: LocationAPIRequest) = {
  buffer += ((key, value))
  null
}
```

LOCATION API TRANSFORMER

```
override def punctuate(timestamp: Long) = {
  val requests = buffer.map(_._2)
    .grouped(250)
    .map(locationAPIClient.bulkLookup(_))




}
```

LOCATION API TRANSFORMER

```
override def punctuate(timestamp: Long) = {
  val requests = buffer.map(_._2)
    .grouped(250)
    .map(locationAPIClient.bulkLookup(_))

  val forwardedKeys = (for {
    request <- requests
    queryResponse <- Try(Await.result(request, Duration.Inf)) getOrElse Nil
    (key, _) <- buffer.find(kv => kv._2 == queryResponse.query)
  } yield {
    context.forward(key, Some(queryResponse.geoLocation))
    key
  }).toSet




}
```

LOCATION API TRANSFORMER

```
override def punctuate(timestamp: Long) = {
  val requests = buffer.map(_._2)
    .grouped(250)
    .map(locationAPIClient.bulkLookup(_))

  val forwardedKeys = (for {
    request <- requests
    queryResponse <- Try(Await.result(request, Duration.Inf)) getOrElse Nil
    (key, _) <- buffer.find(kv => kv._2 == queryResponse.query)
  } yield {
    context.forward(key, Some(queryResponse.geoLocation))
    key
  }).toSet

  // Forward unsuccessful events
  for { (key, value) <- buffer if !forwardedKeys.contains(key) } context.forward(key, None)


}
```

LOCATION API TRANSFORMER

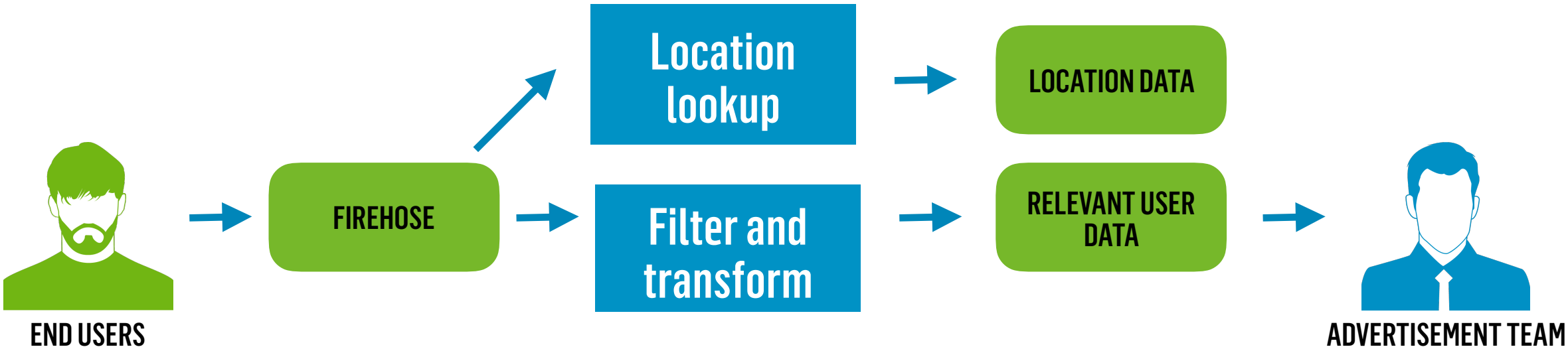
```
override def punctuate(timestamp: Long) = {
  val requests = buffer.map(_._2)
    .grouped(250)
    .map(locationAPIClient.bulkLookup(_))

  val forwardedKeys = (for {
    request <- requests
    queryResponse <- Try(Await.result(request, Duration.Inf)) getOrElse Nil
    (key, _) <- buffer.find(kv => kv._2 == queryResponse.query)
  } yield {
    context.forward(key, Some(queryResponse.geoLocation))
    key
  }).toSet

  // Forward unsuccessful events
  for { (key, value) <- buffer if !forwardedKeys.contains(key) } context.forward(key, None)
  context.commit()
  buffer.clear()
  null
}
```

LOCATION API TRANSFORMER

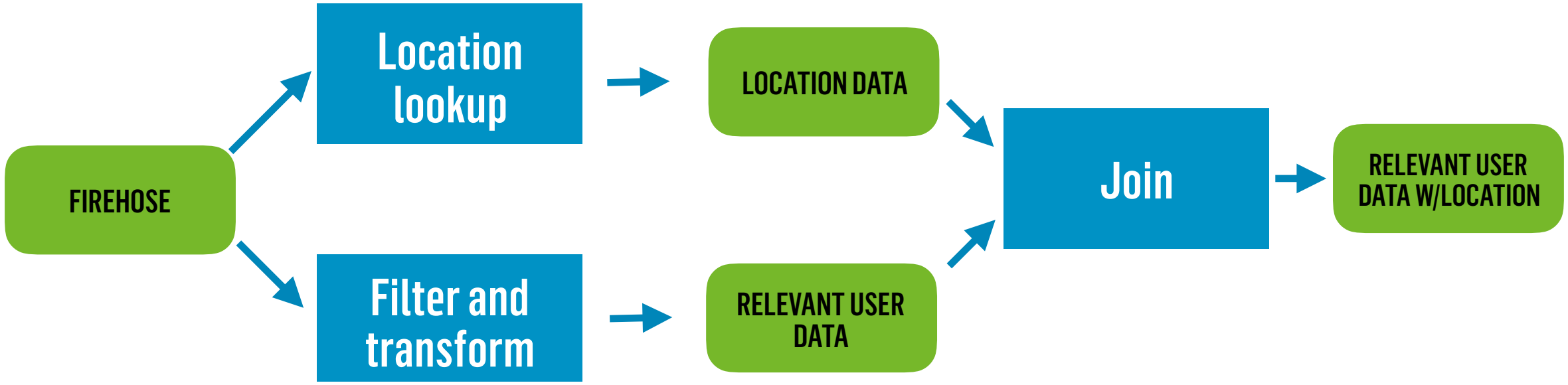
```
override def punctuate(timestamp: Long) = {
  val requests = buffer.map(_._2)
    .grouped(250)
    .map(locationAPIClient.bulkLookup(_))

  val forwardedKeys = (for {
    request <- requests
    queryResponse <- Try(Await.result(request, Duration.Inf)) getOrElse Nil
    (key, _) <- buffer.find(kv => kv._2 == queryResponse.query)
  } yield {
    context.forward(key, Some(queryResponse.geoLocation))
    key
  }).toSet

  // Forward unsuccessful events
  for { (key, value) <- buffer if !forwardedKeys.contains(key) } context.forward(key, None
  context.commit()
  buffer.clear()
  null
}
```

RECAP

```
builder
  .stream(stringSerde, byteSerde, "Firehose")
  .flatMapValues[JsonNode] { (value: Array[Byte]) =>
      Try(jsonSerde.deserializer.deserialize(null, value)).toOption.toIterable.asJava
  }
  .flatMapValues[LocationAPIRequest(jsonToLocationAPIRequest(_).toIterable.asJava)
  .transform(locationAPITransformerSupplier)
  .mapValues[JsonNode](_.map(fromReverseLocationAPIResponseToJson).orNull)
  .to(stringSerde, jsonSerde, "LocationData")
```



JOINING THE STREAMS



JOINING THE STREAMS

```
val userData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "UserData")
  .flatMapValues[JsonNode] {...}

val locationData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "LocationData")
  .flatMapValues[JsonNode] {…}

userData
  .join(
    locationData,
```

JOINING THE STREAMS

```
val userData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "UserData")
  .flatMapValues[JsonNode] {...}

val locationData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "LocationData")
  .flatMapValues[JsonNode] {…}

userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
```

JOINING THE STREAMS

```
val userData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "UserData")
  .flatMapValues[JsonNode] {...}

val locationData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "LocationData")
  .flatMapValues[JsonNode] {…}

userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
    JoinWindows.of(10000),
```

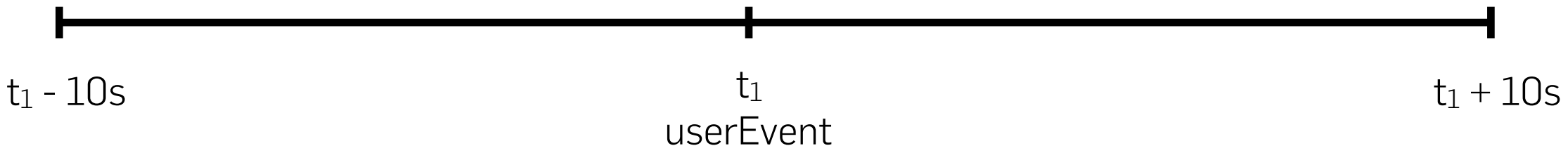
JOINING THE STREAMS

```
val userData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "UserData")
  .flatMapValues[JsonNode] {...}

val locationData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "LocationData")
  .flatMapValues[JsonNode] {…}

userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
    JoinWindows.of(10000),
    stringSerde, // key Serde
    jsonSerde, // relevantUserData value Serde
    jsonSerde // locationData value Serde
  )
```

JOINING THE STREAMS

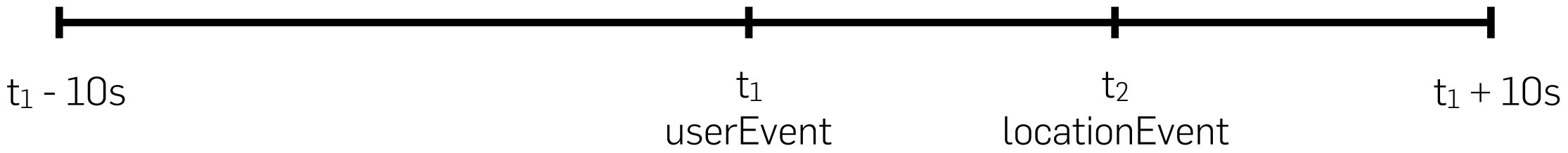
```
val userData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "UserData")
  .flatMapValues[JsonNode] {...}

val locationData: KStream[String, JsonNode] = builder
  .stream(stringSerde, byteSerde, "LocationData")
  .flatMapValues[JsonNode] {…}

userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
    JoinWindows.of(10000),
    stringSerde, // key Serde
    jsonSerde, // relevantUserData value Serde
    jsonSerde // locationData value Serde
  )
  .to(stringSerde, jsonSerde, "UserDataWithLocation")
```

JOIN WINDOWS




```
userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
    JoinWindows.of(10000)
  )
```

JOIN WINDOWS




```
userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
    JoinWindows.of(10000)
  )
```

0. THE BACKGROUND STORY

1. THE CHALLENGE

2. THE TECHNOLOGY

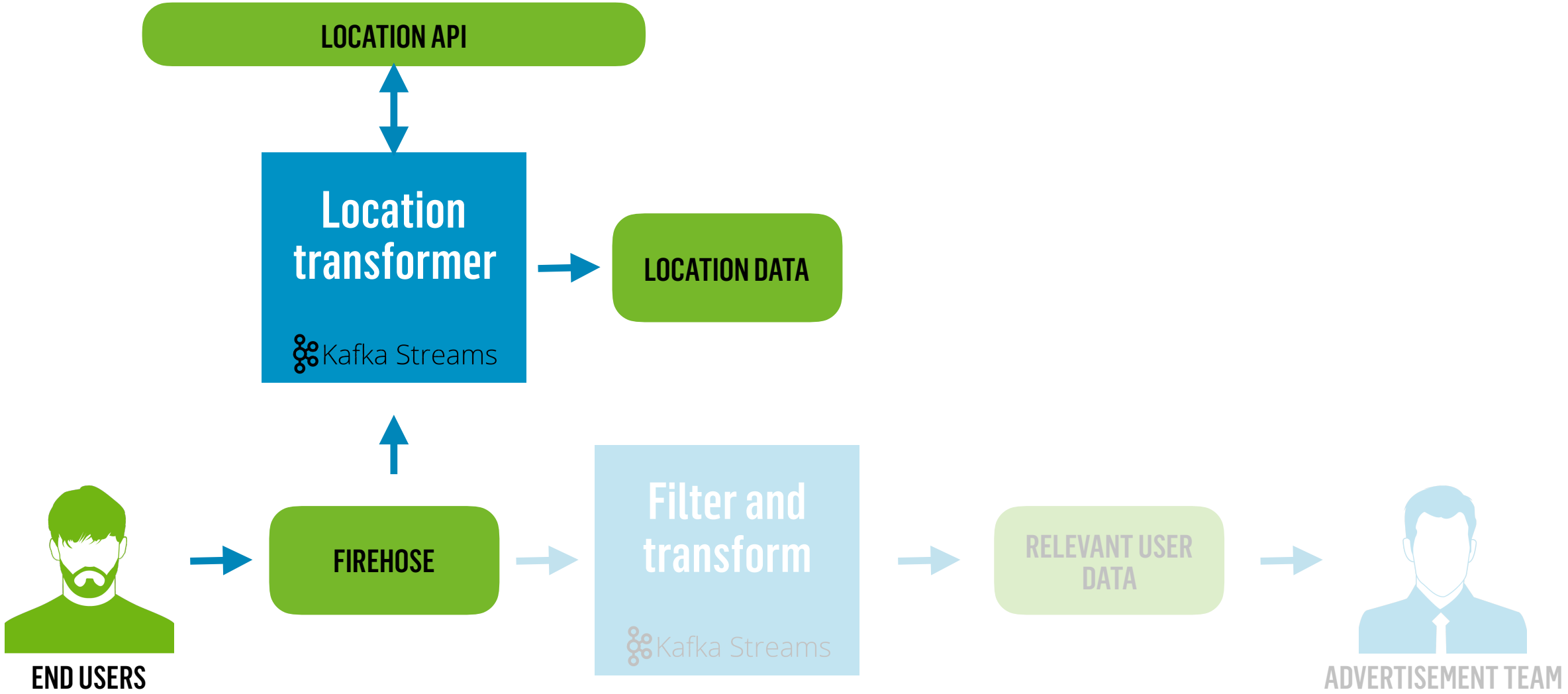
3. THE APPLICATION

4. THE DEPLOYMENT

5. THE CONCLUSION

aka “The WTFs”

THE LOCATION TRANSFORMER



LOCATION TRANSFORMER PERFORMANCE

438.84ms

500MS PUNCTUATION INTERVAL

NO CACHE

LOCATION TRANSFORMER PERFORMANCE

438.84ms **81.13ms**

500MS PUNCTUATION INTERVAL

NO CACHE

LOCATION TRANSFORMER PERFORMANCE

438.84ms

500MS PUNCTUATION INTERVAL
NO CACHE

81.13ms

500MS PUNCTUATION INTERVAL
IN-MEMORY CACHE

LOCATION TRANSFORMER PERFORMANCE

438.84ms

500MS PUNCTUATION INTERVAL
NO CACHE

81.13ms

500MS PUNCTUATION INTERVAL
IN-MEMORY CACHE

36.04ms

LOCATION TRANSFORMER PERFORMANCE

438.84ms

500MS PUNCTUATION INTERVAL
NO CACHE

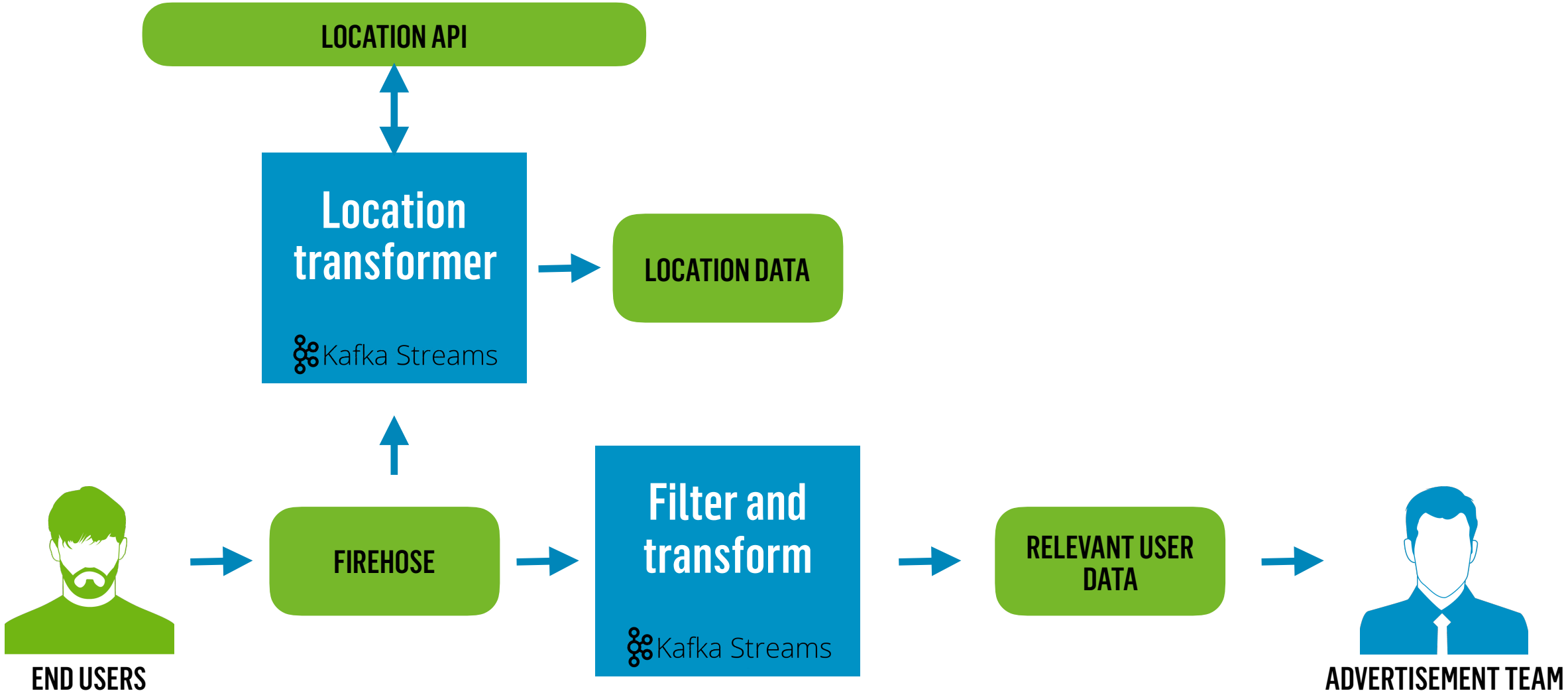
81.13ms

500MS PUNCTUATION INTERVAL
IN-MEMORY CACHE

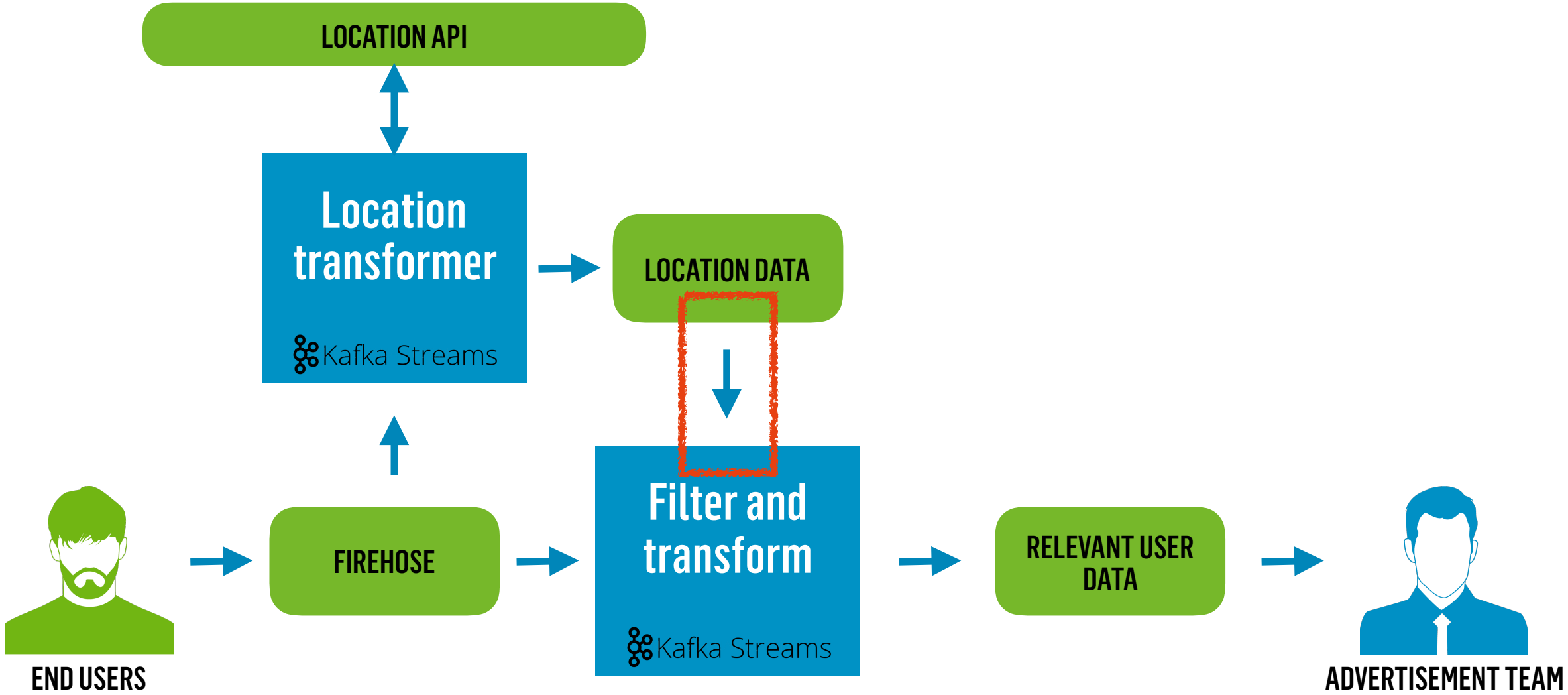
36.04ms

100MS PUNCTUATION INTERVAL
IN-MEMORY CACHE

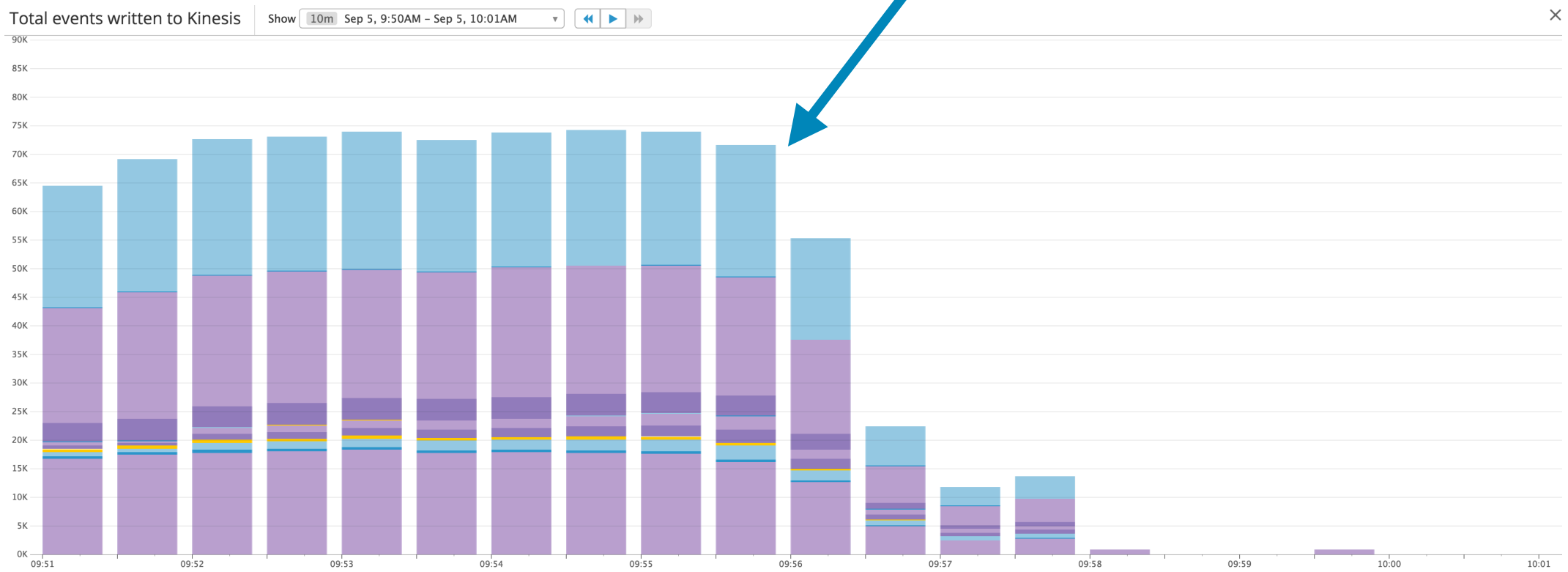
THE JOIN COMPONENT

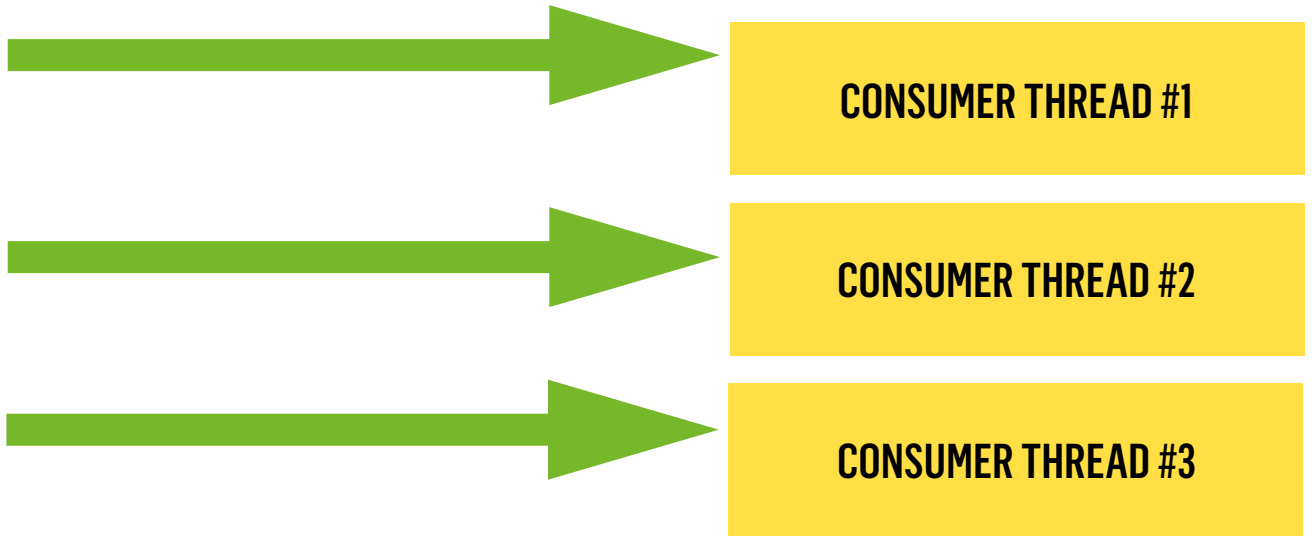


THE JOIN COMPONENT

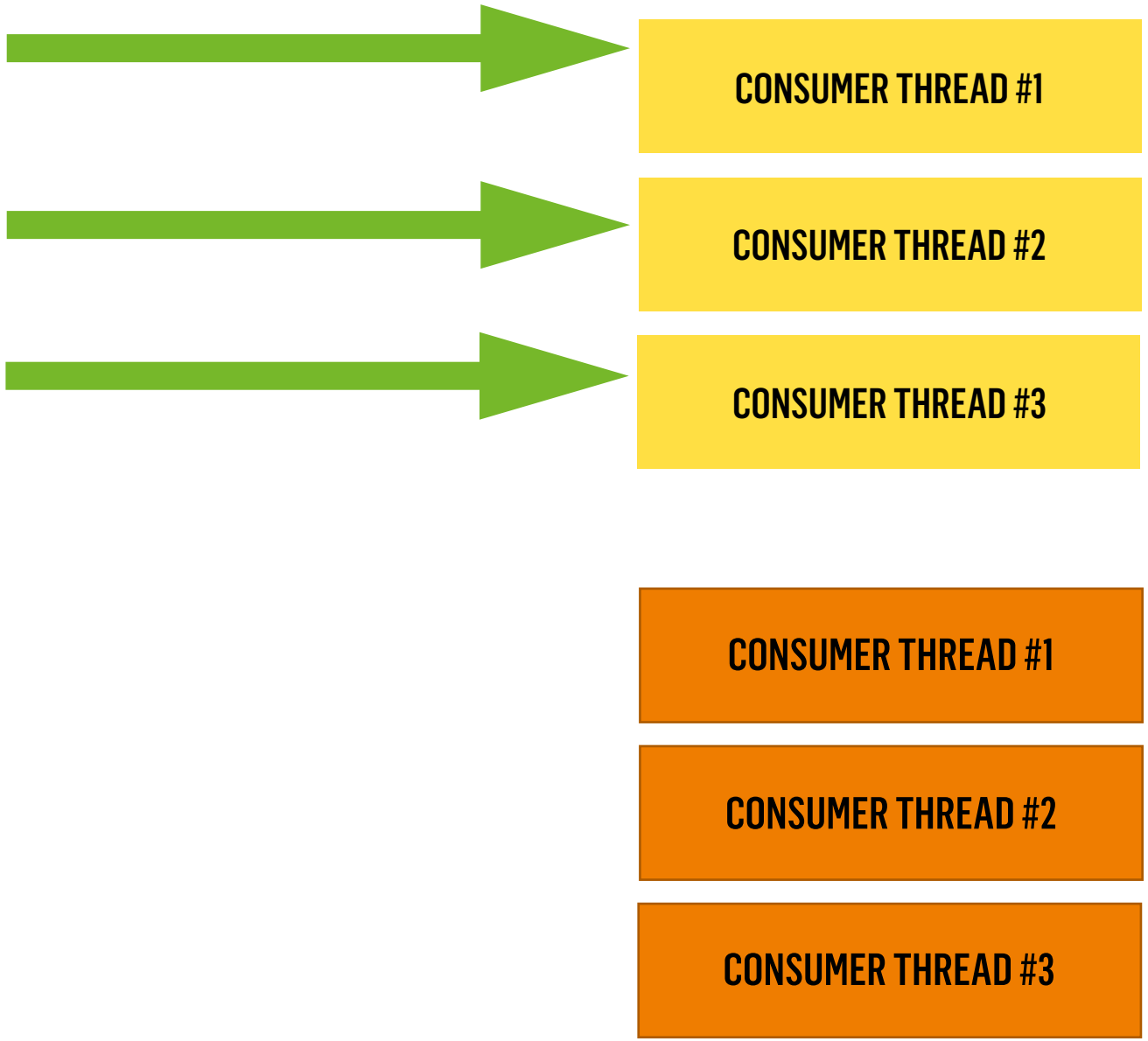


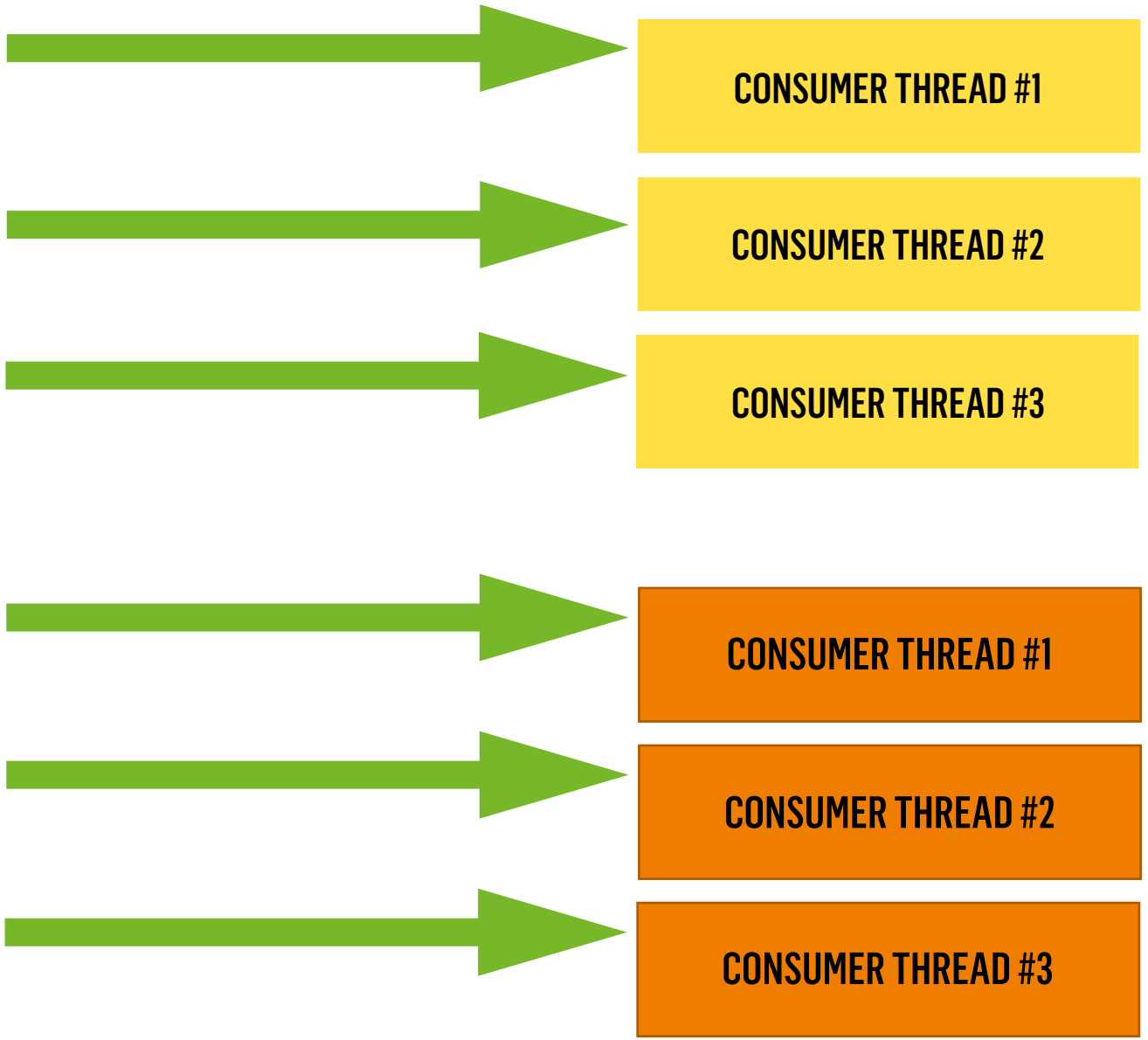
Deployment of new application with **`join`**





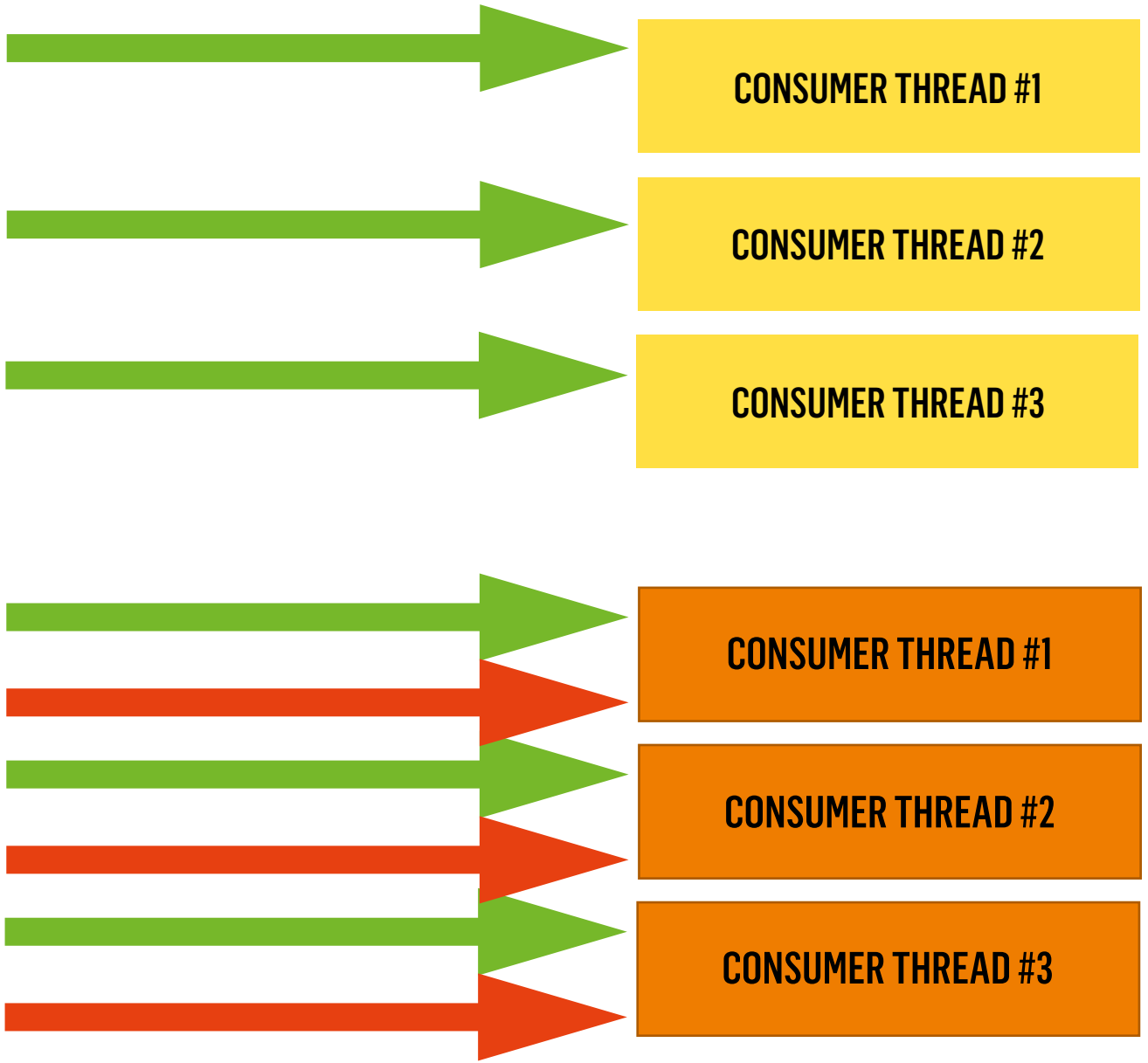
OLD CLUSTER


CONSUMER THREAD #1
CONSUMER THREAD #2
CONSUMER THREAD #3
OLD CLUSTER
CONSUMER THREAD #1
CONSUMER THREAD #2
CONSUMER THREAD #3
NEW CLUSTER



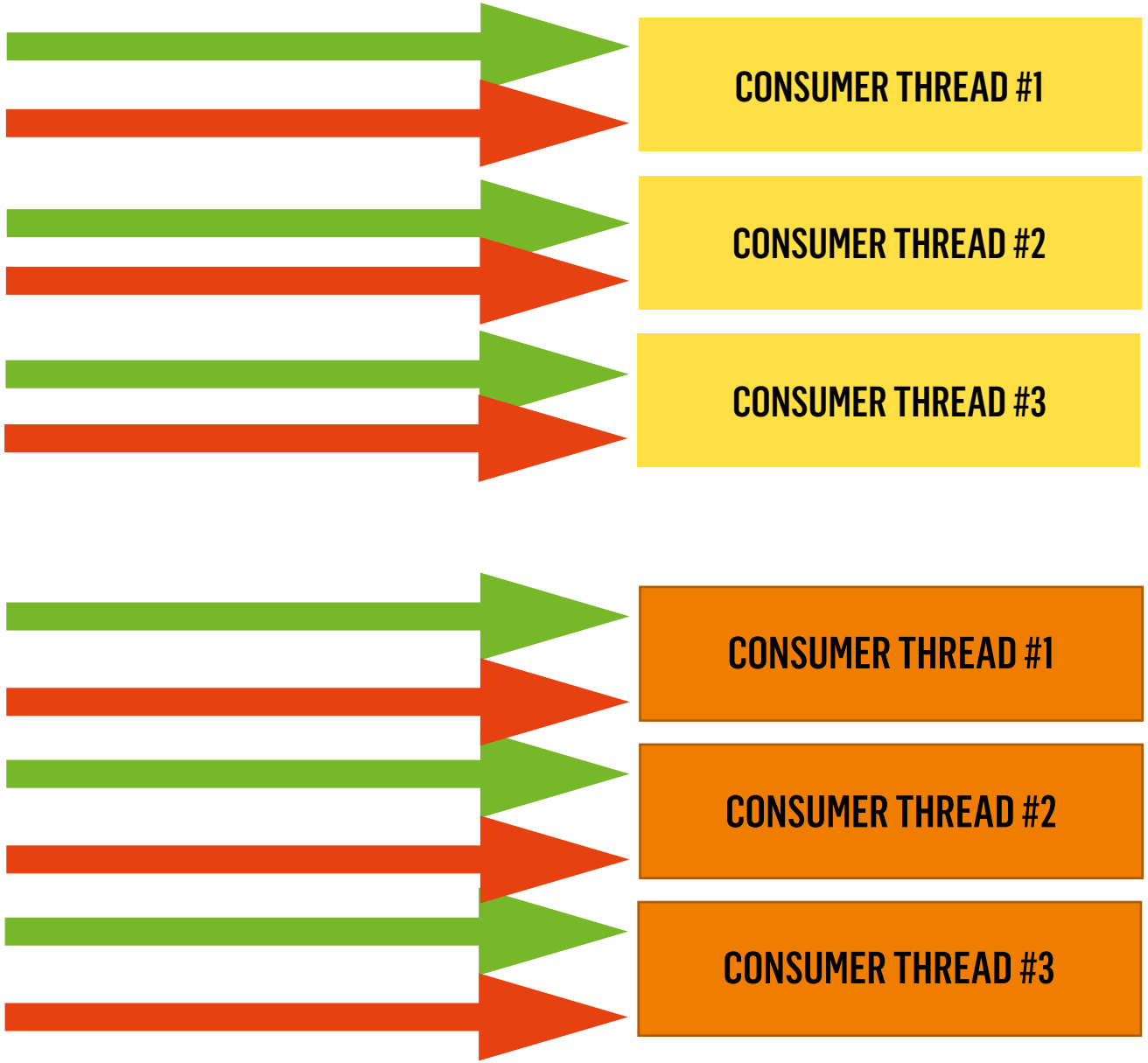
OLD CLUSTER

NEW CLUSTER



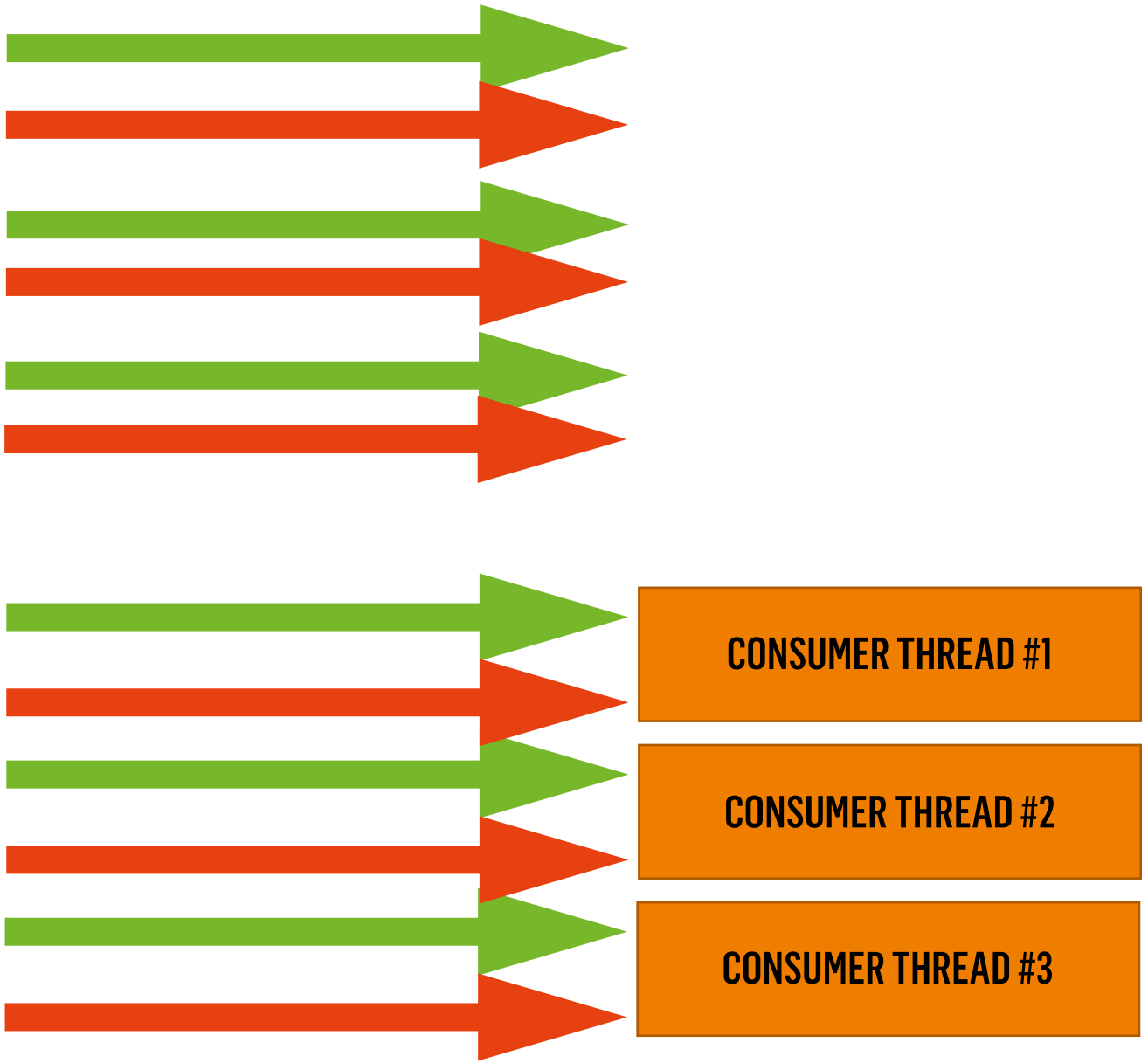
OLD CLUSTER

NEW CLUSTER

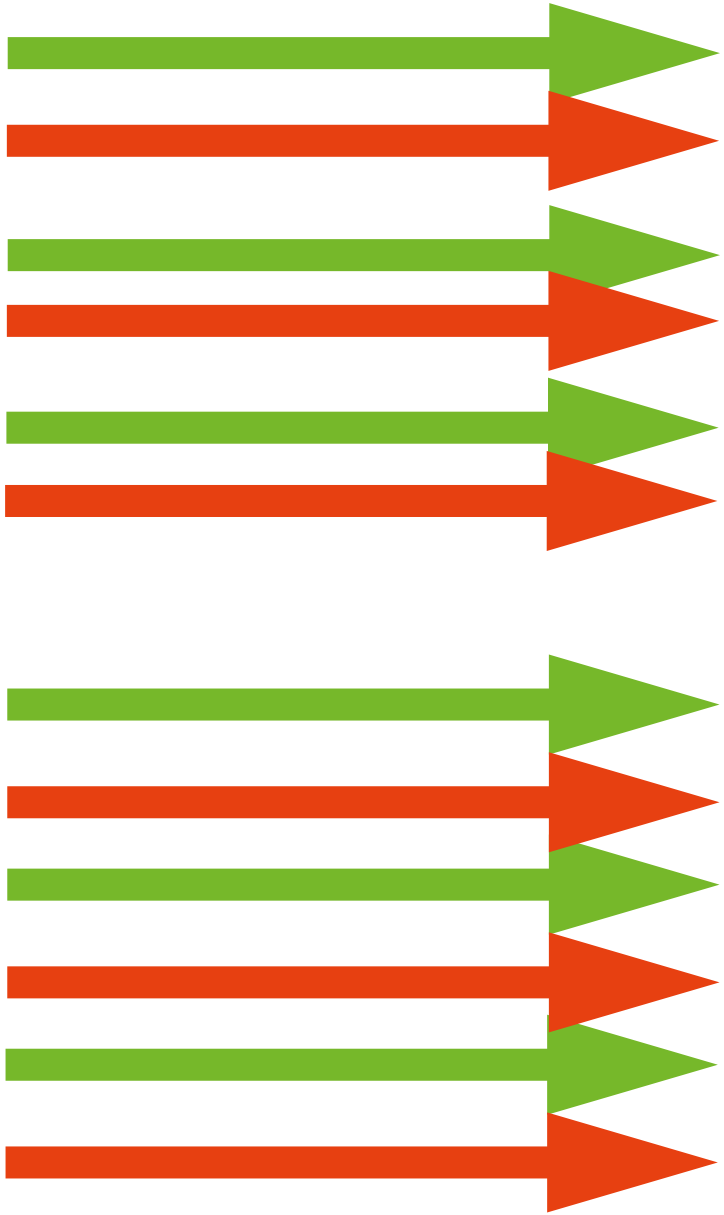


OLD CLUSTER

NEW CLUSTER

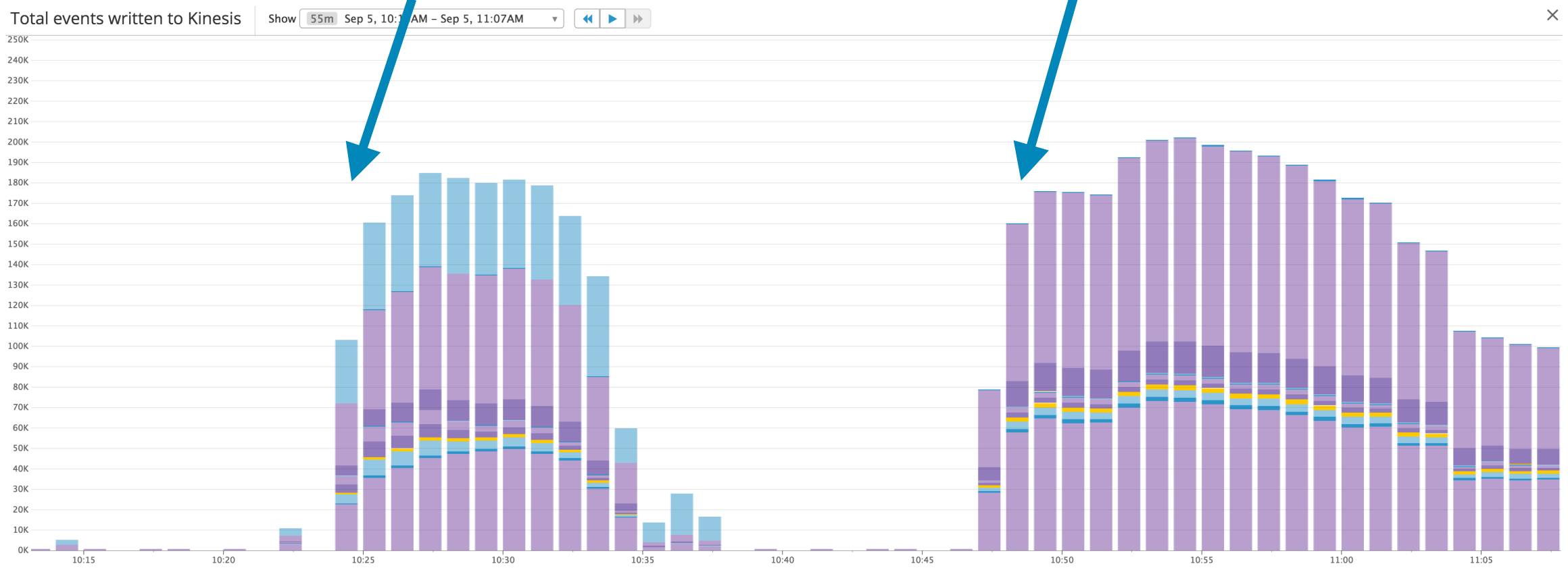


NEW CLUSTER



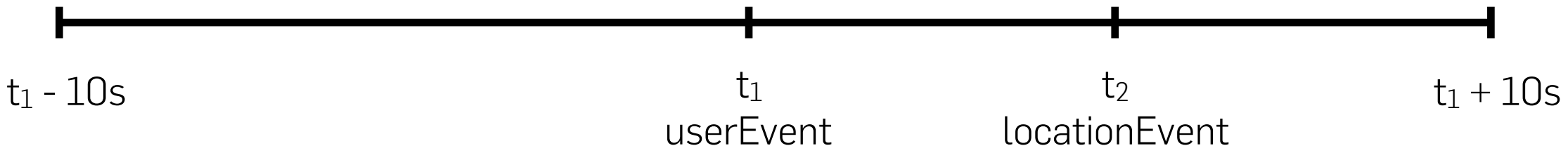
Scale up new cluster only

Capitulation



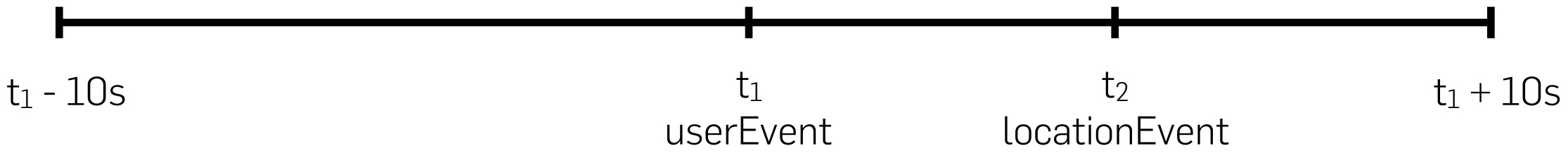
SIGSEGV

JOIN WINDOWS



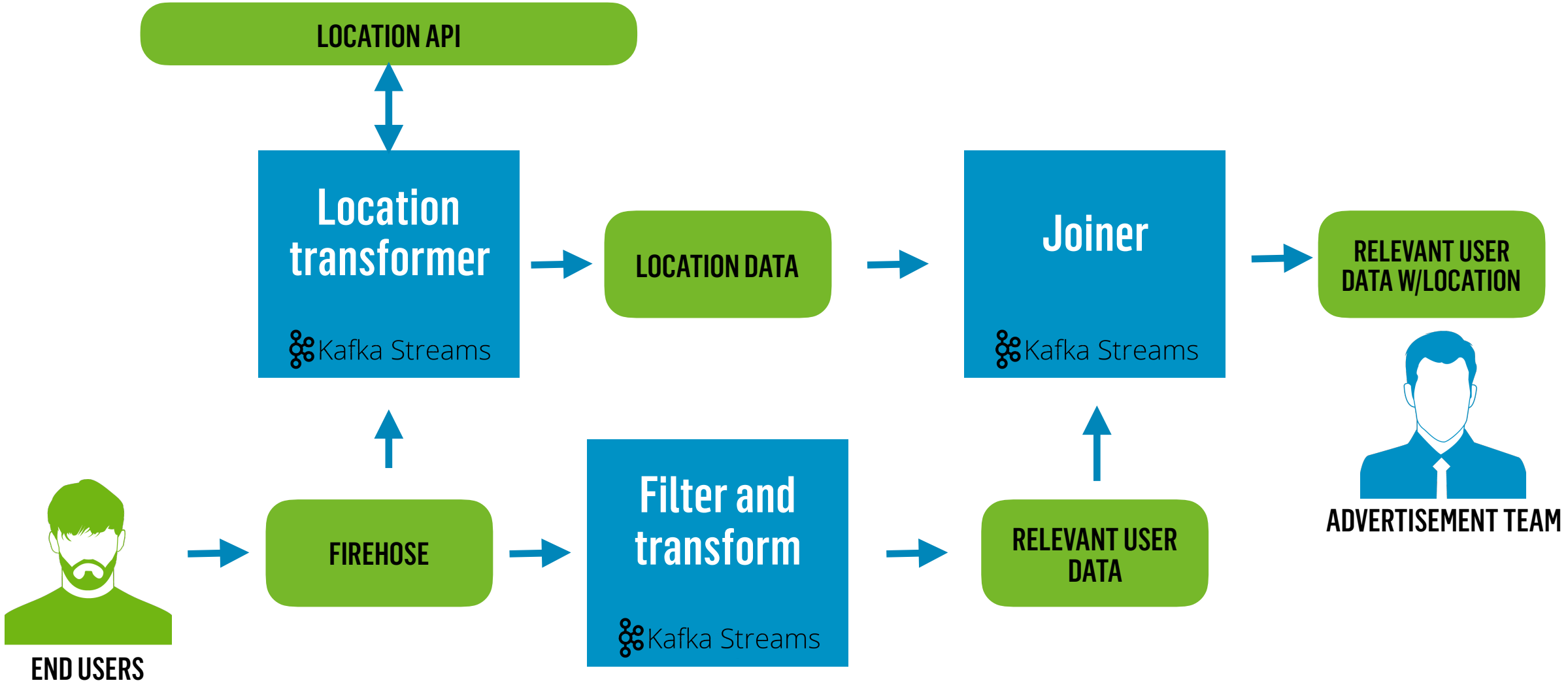

```
userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
    JoinWindows.of(10000)
  )
```

JOIN WINDOWS



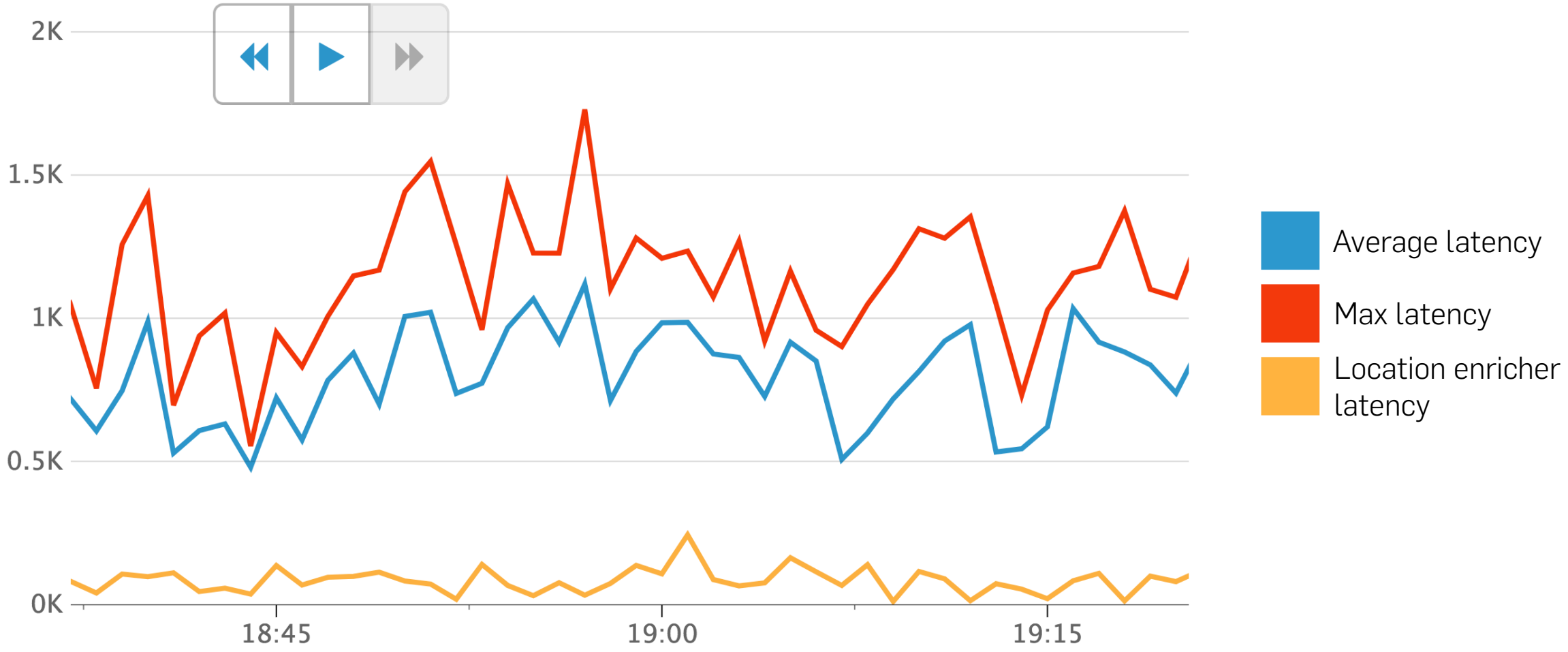

```
userData
  .join(
    locationData,
    (userEvent: JsonNode, locationEvent: JsonNode) => {
      userEvent.asInstanceOf[ObjectNode].set("location", locationEvent)
    },
    JoinWindows.of(10000).until(30000)
  )
```

THE JOIN COMPONENT

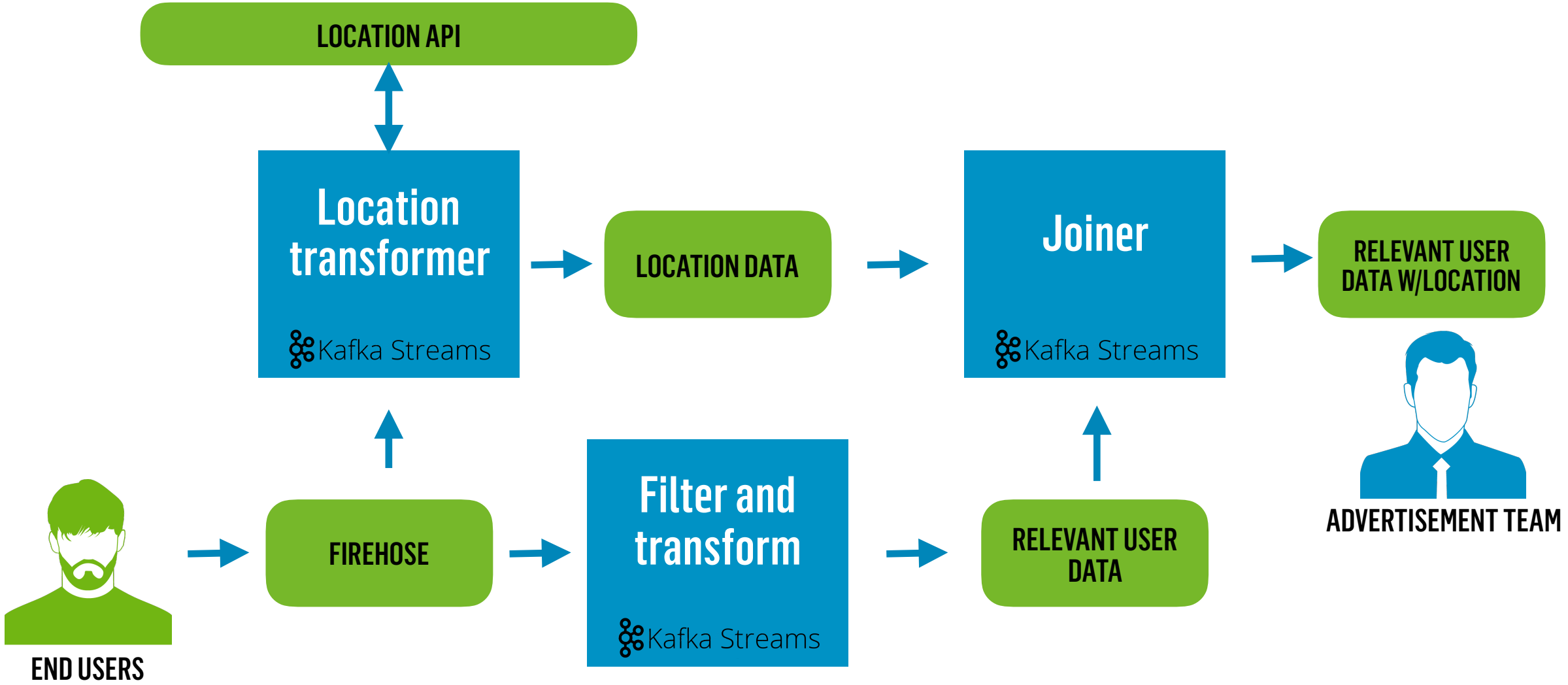


http://knowyourmeme.com/memes/success-kid-i-hate-sandcastles

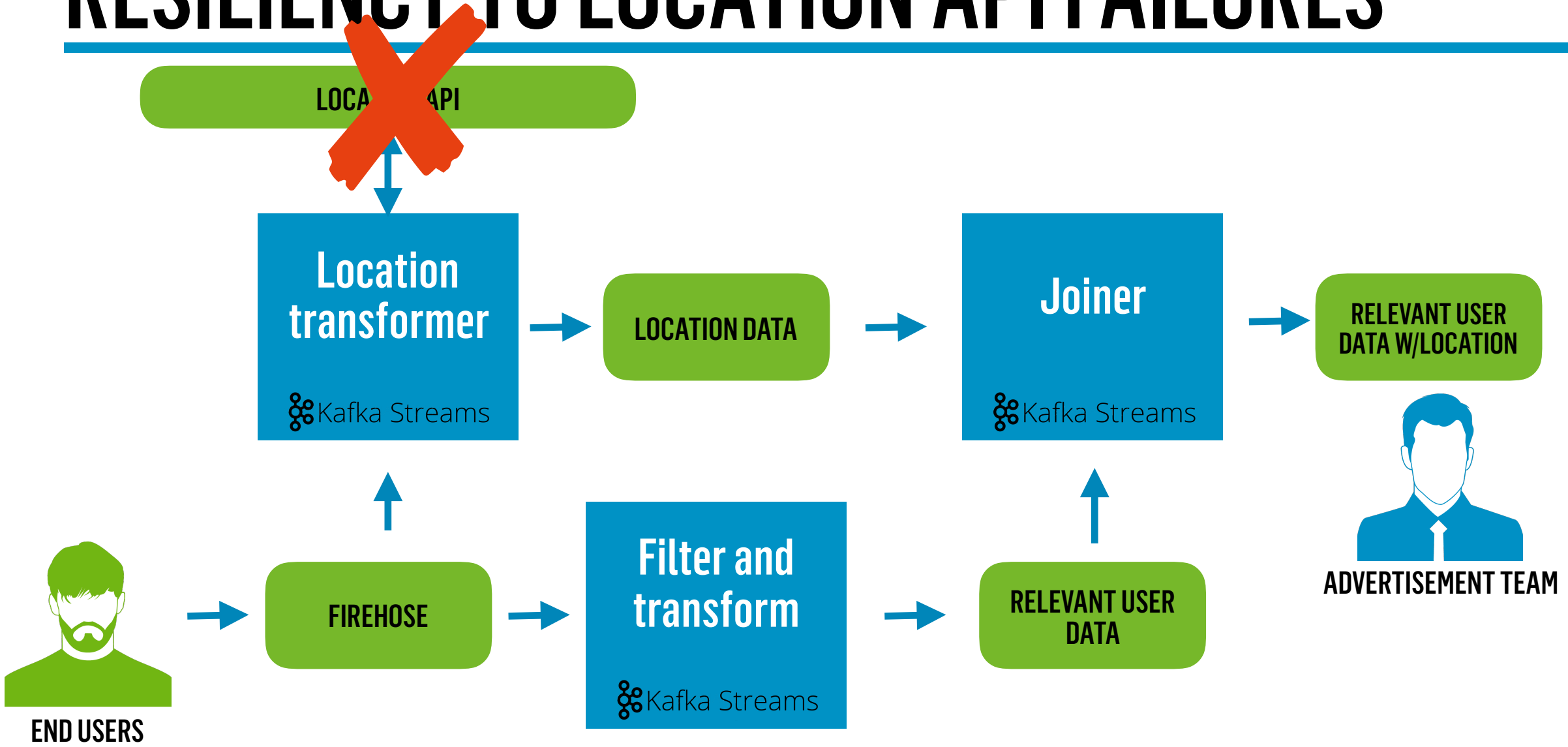
JOIN PERFORMANCE



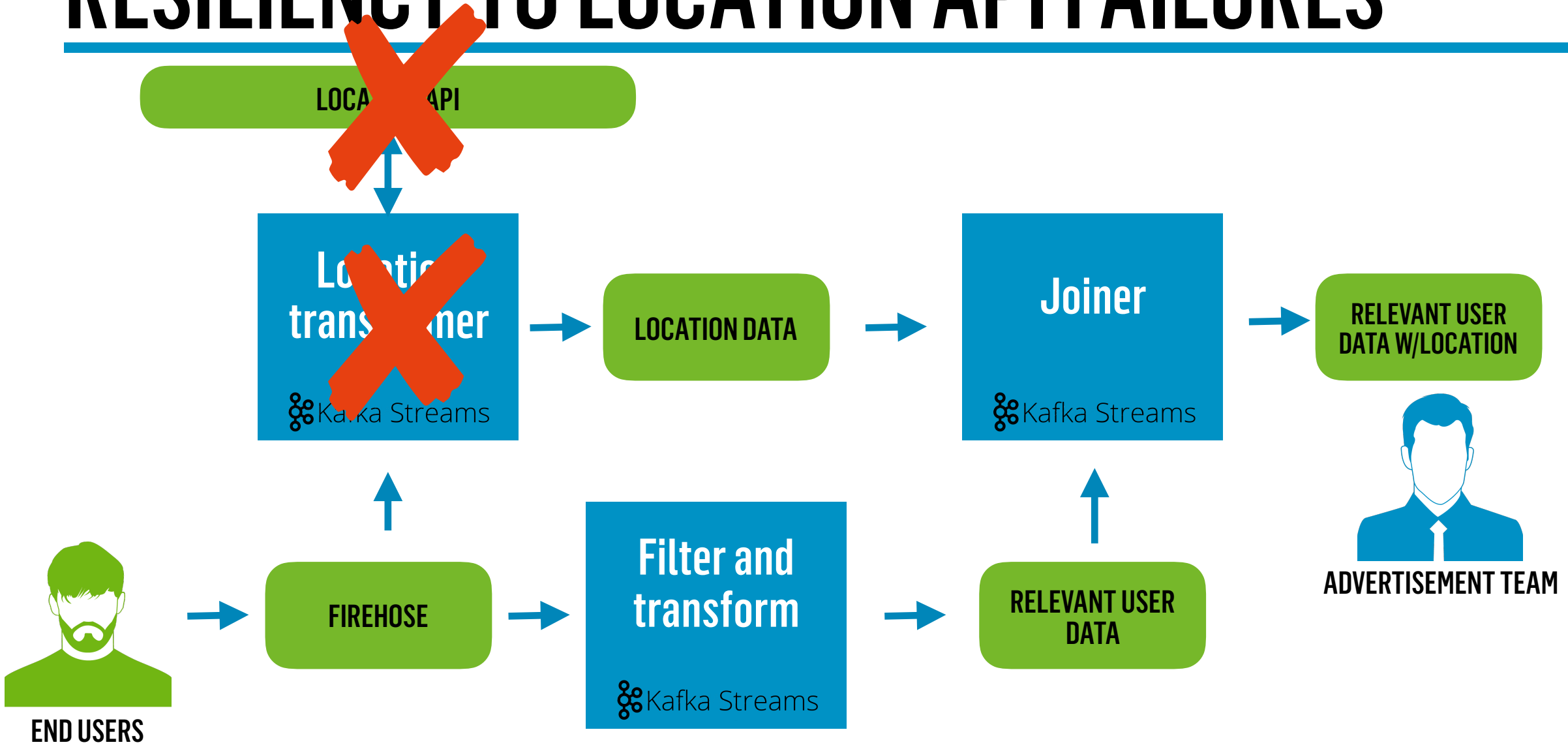
RESILIENCY TO LOCATION API FAILURES



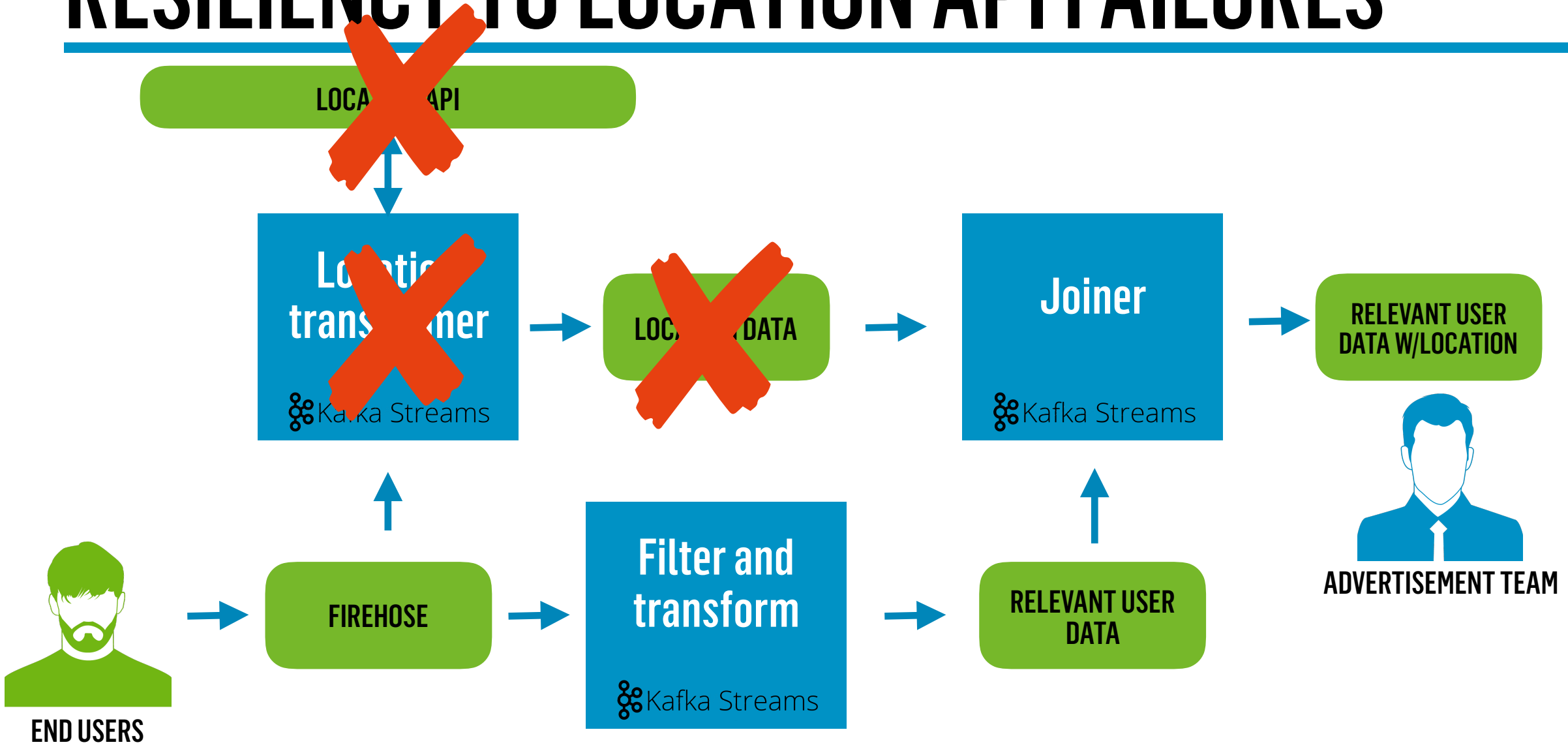
RESILIENCY TO LOCATION API FAILURES



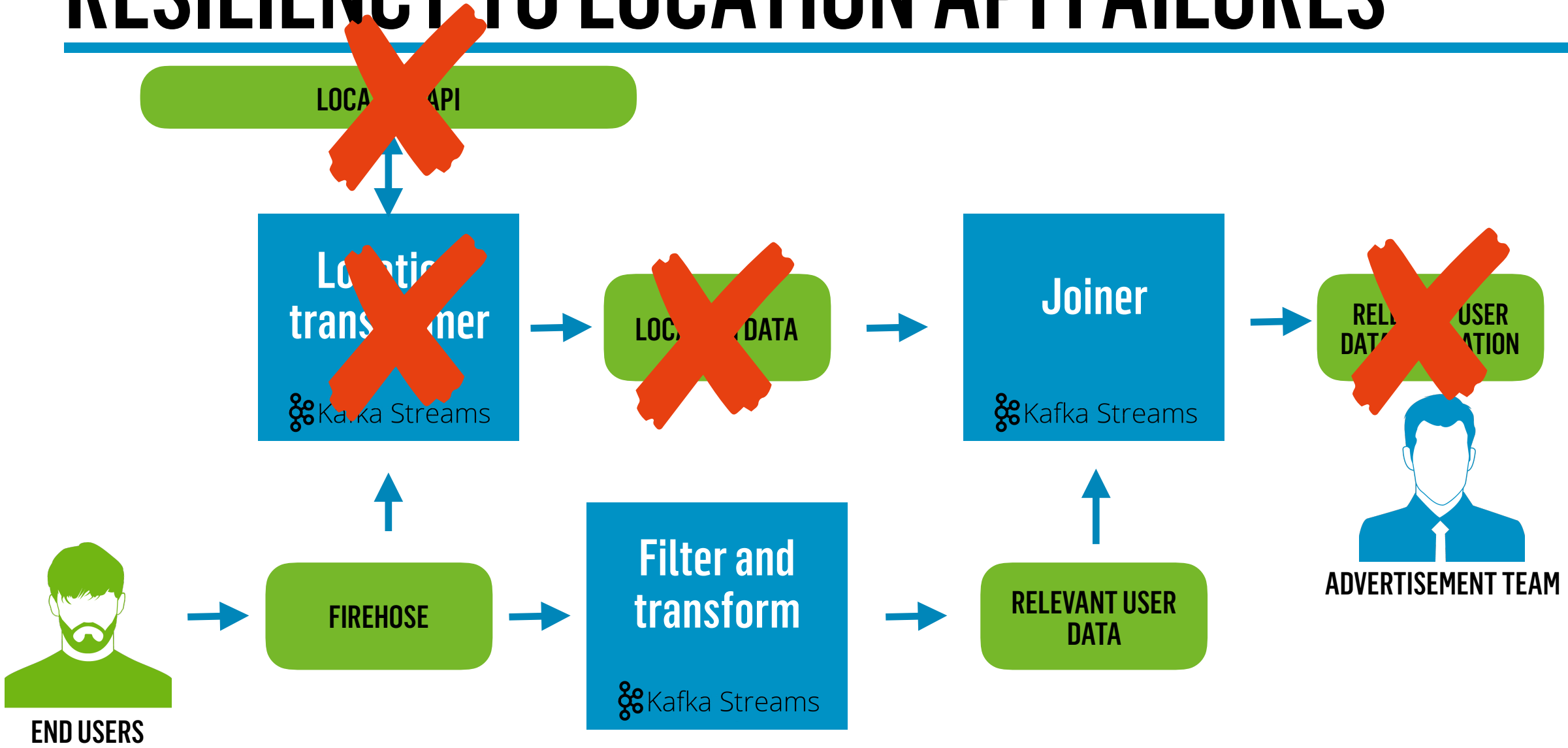
RESILIENCY TO LOCATION API FAILURES



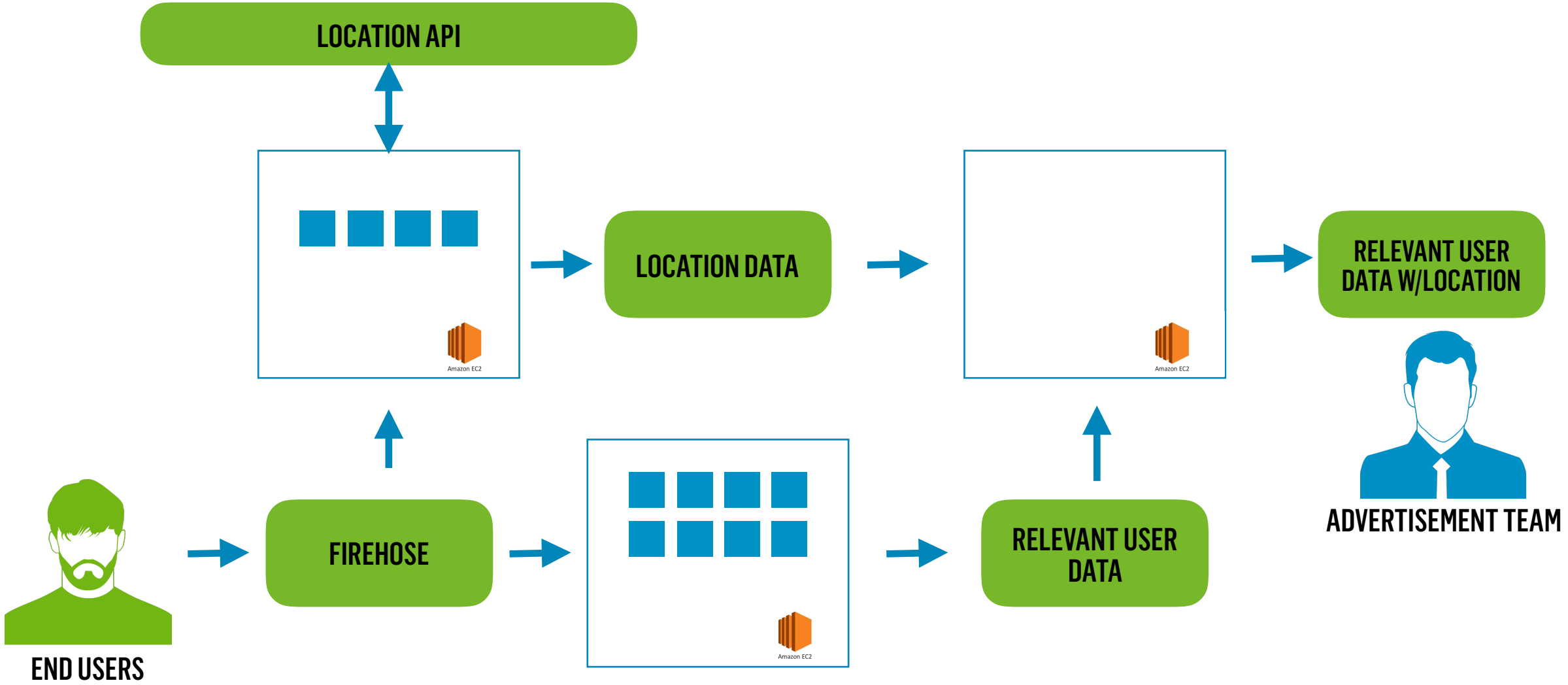
RESILIENCY TO LOCATION API FAILURES



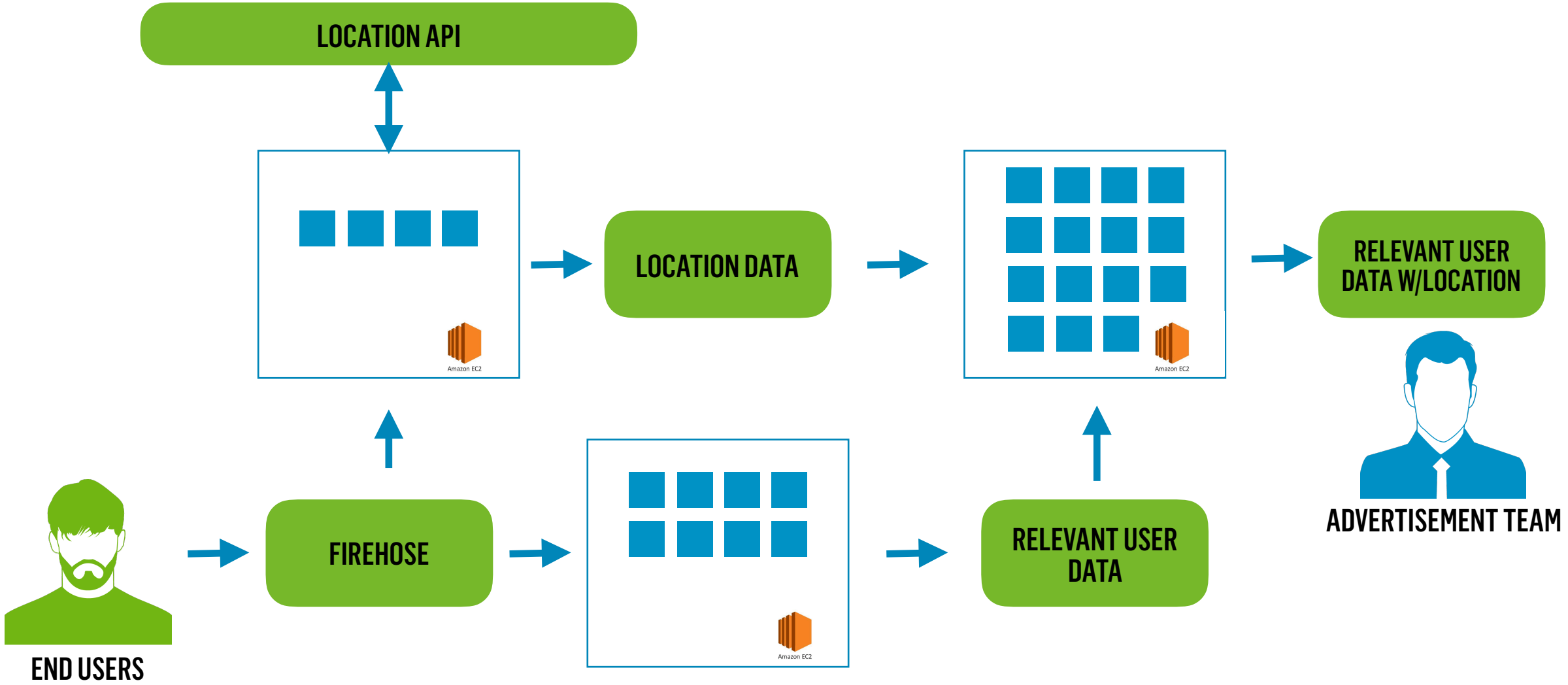
RESILIENCY TO LOCATION API FAILURES



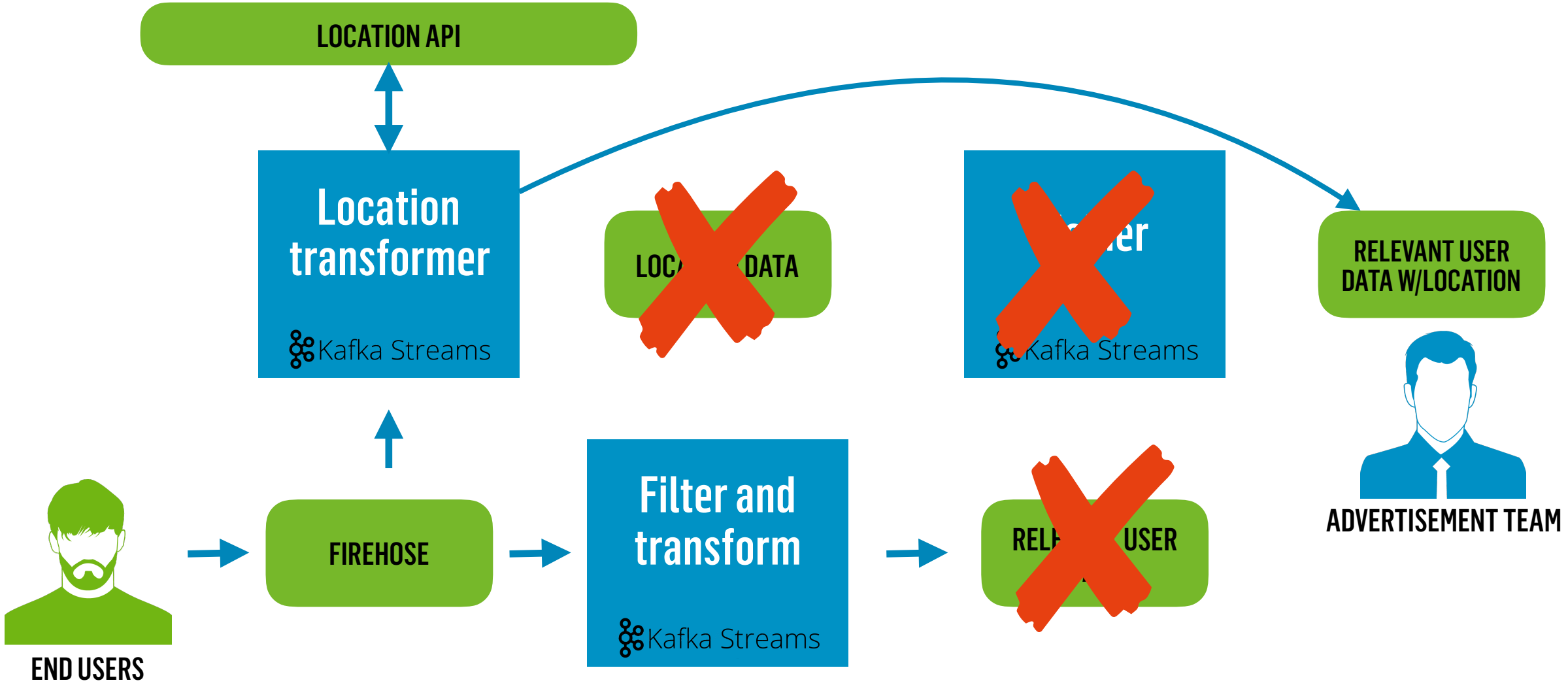
AVERAGE CLUSTER SIZES



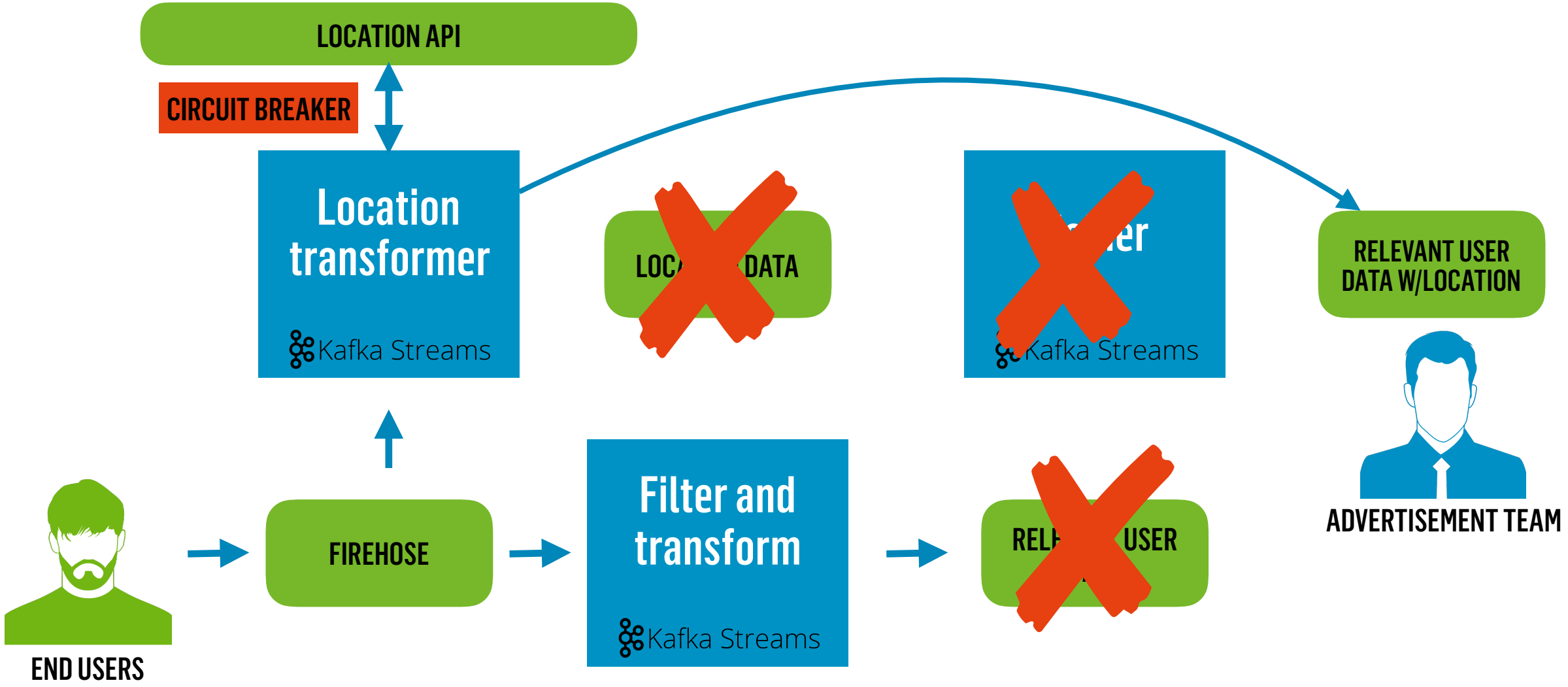
AVERAGE CLUSTER SIZES



CURRENT STATUS



CURRENT STATUS



0. THE BACKGROUND STORY

1. THE CHALLENGE

2. THE TECHNOLOGY

3. THE APPLICATION

4. THE DEPLOYMENT

5. THE CONCLUSION

CONCLUSION

- STRAIGHT FORWARD APPLICATION DEVELOPMENT
- SCALES WELL
- ONE SHOULD OPTIMIZE FOR MANY SMALLER APPLICATIONS
- STATEFUL OPERATIONS CAN BE EXPENSIVE AND INFLEXIBLE

QUESTIONS?