A Data Streaming Architecture with Apache Flink

Robert Metzger
@rmetzger_
rmetzger@apache.org

Berlin Buzzwords,
June 7, 2016

Talk overview

- My take on the stream processing space, and how it changes the way we think about data
- Transforming an existing data analysis pattern into the streaming world (“Streaming ETL”)
- Demo

Apache Flink

- Apache Flink is an open source stream processing framework
  - Low latency
  - High throughput
  - Stateful
  - Distributed
- Developed at the Apache Software Foundation, 1.0.0 released in March 2016, used in production

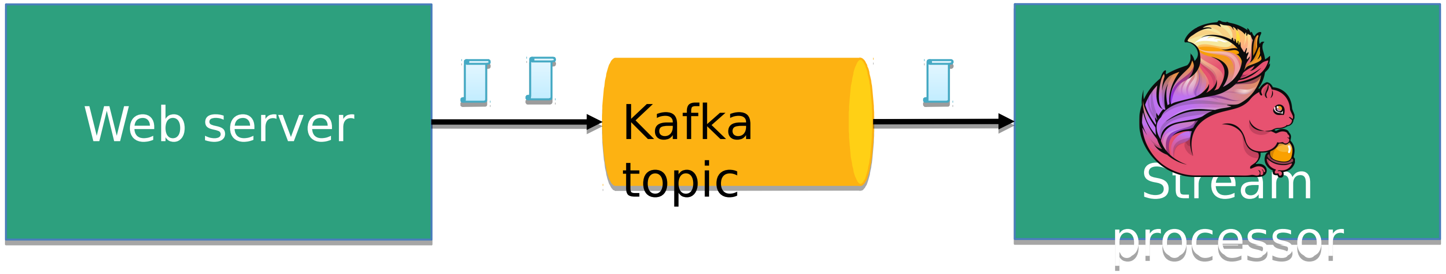
Entering the streaming era

Streaming is the biggest change in data infrastructure since Hadoop

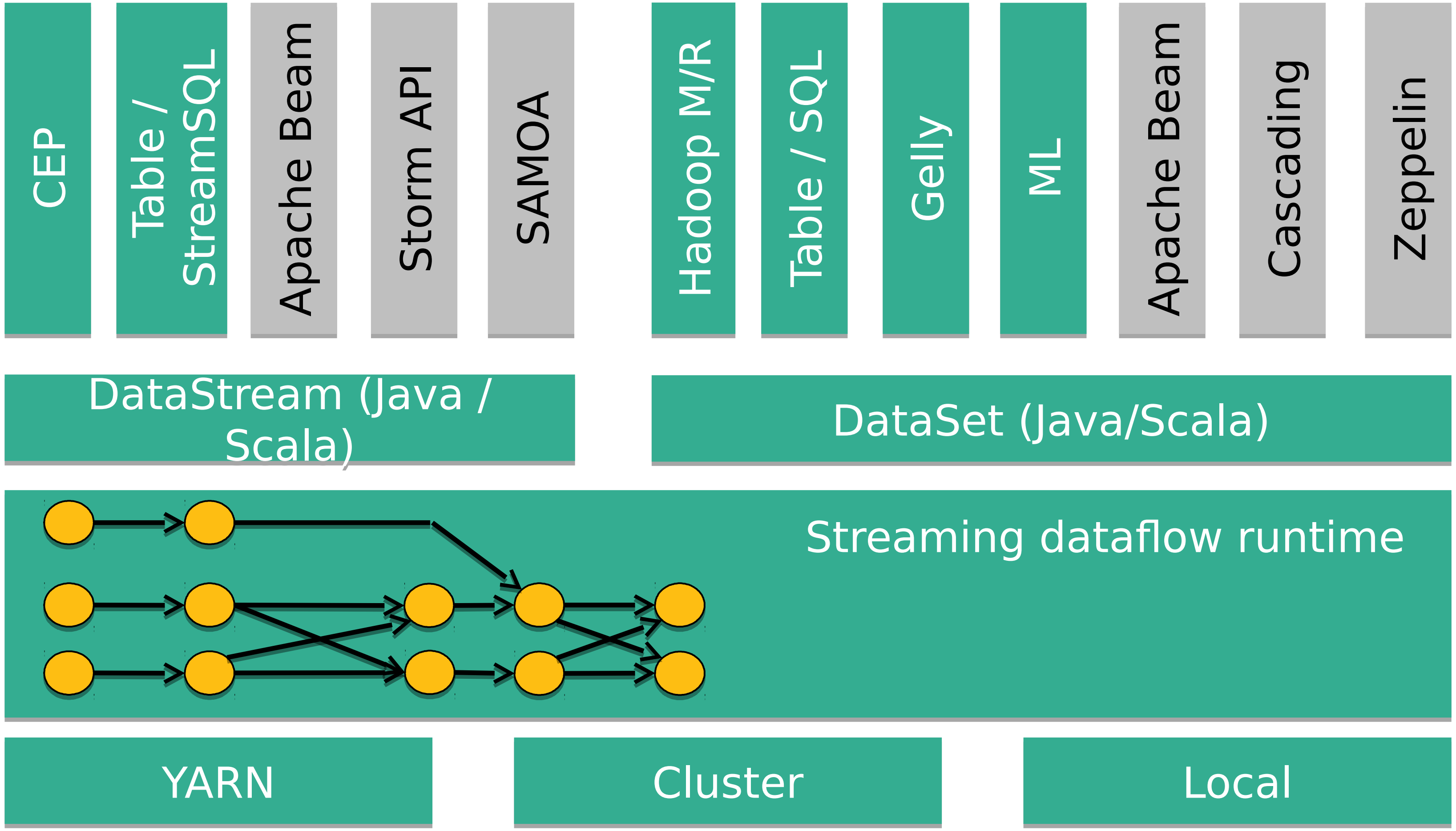
1. Radically simplified infrastructure
2. Do more with your data, faster
3. Can completely subsume batch

Real-world data is produced in a continuous fashion.

New systems like Flink and Kafka embrace streaming nature of data.



Apache Flink stack



CEP

Table / StreamSQL

Apache Beam

Storm API

SAMOA

Hadoop M/R

Table / SQL

Gelly

ML

Apache Beam

Cascading

Zeppelin

DataStream (Java / Scala)

DataSet (Java/Scala)

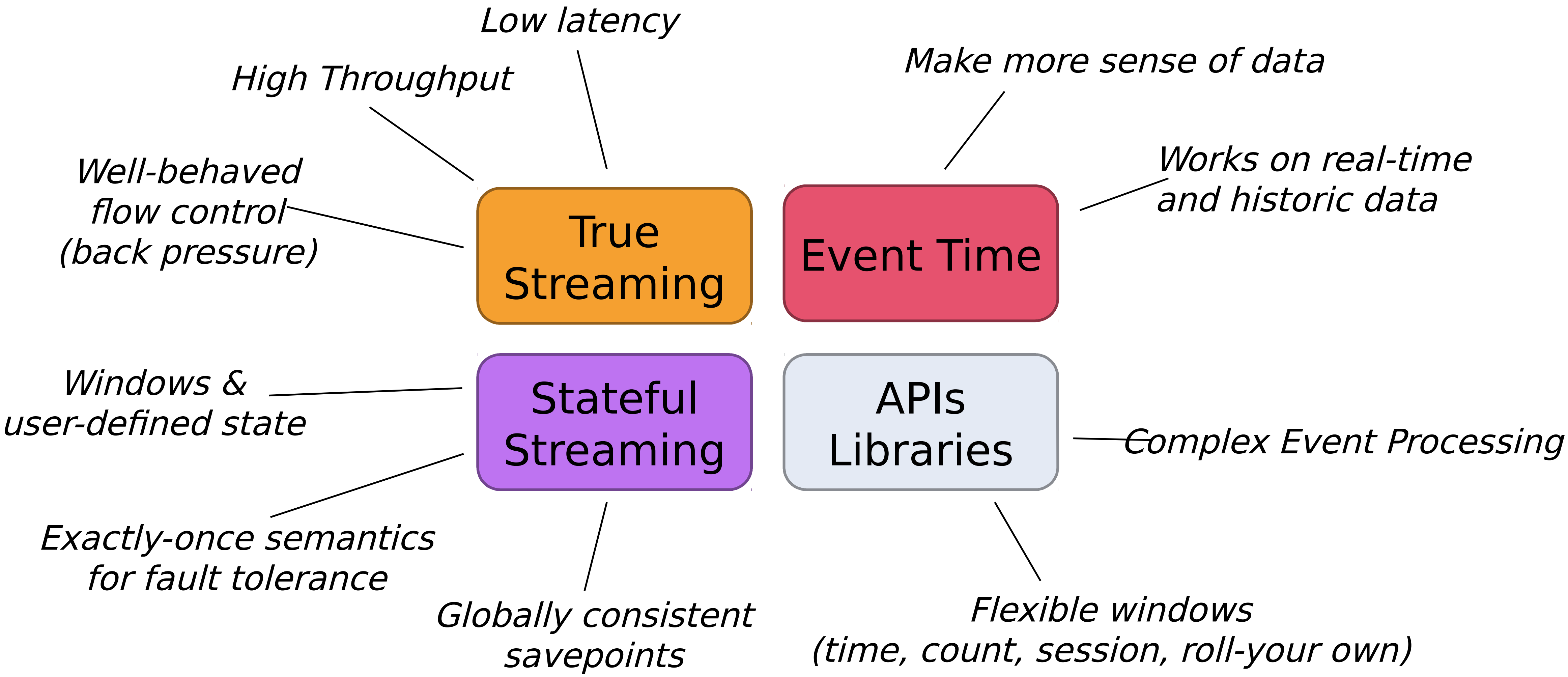
Streaming dataflow runtime

YARN

Cluster

Local

What makes Flink flink?



True Streaming

- *Low latency*
- *High Throughput*
- *Well-behaved flow control (back pressure)*

Event Time

- *Make more sense of data*
- *Works on real-time and historic data*

Stateful Streaming

- *Windows & user-defined state*
- *Exactly-once semantics for fault tolerance*
- *Globally consistent savepoints*

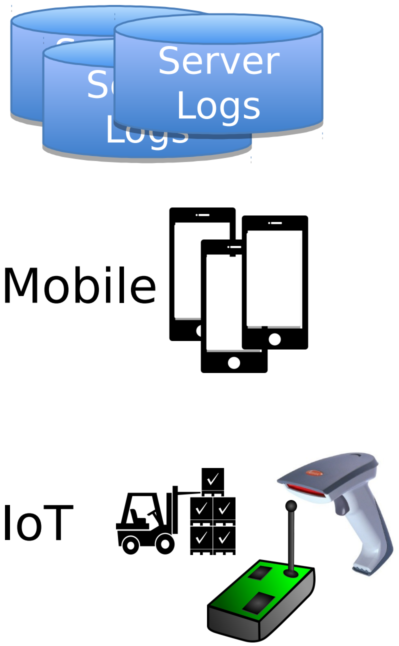
APIs Libraries

- *Complex Event Processing*
- *Flexible windows (time, count, session, roll-your own)*

Moving existing (batch) data analysis into streaming

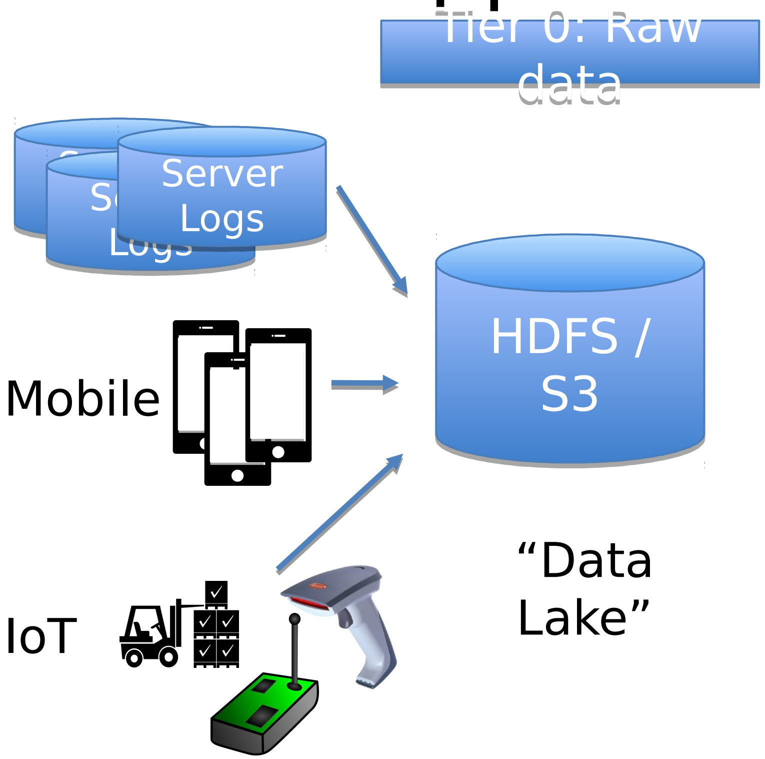
Extract, Transform, Load (ETL)

- ETL: Move data from A to B and transform it on the way
- Old approach:



Extract, Transform, Load (ETL)

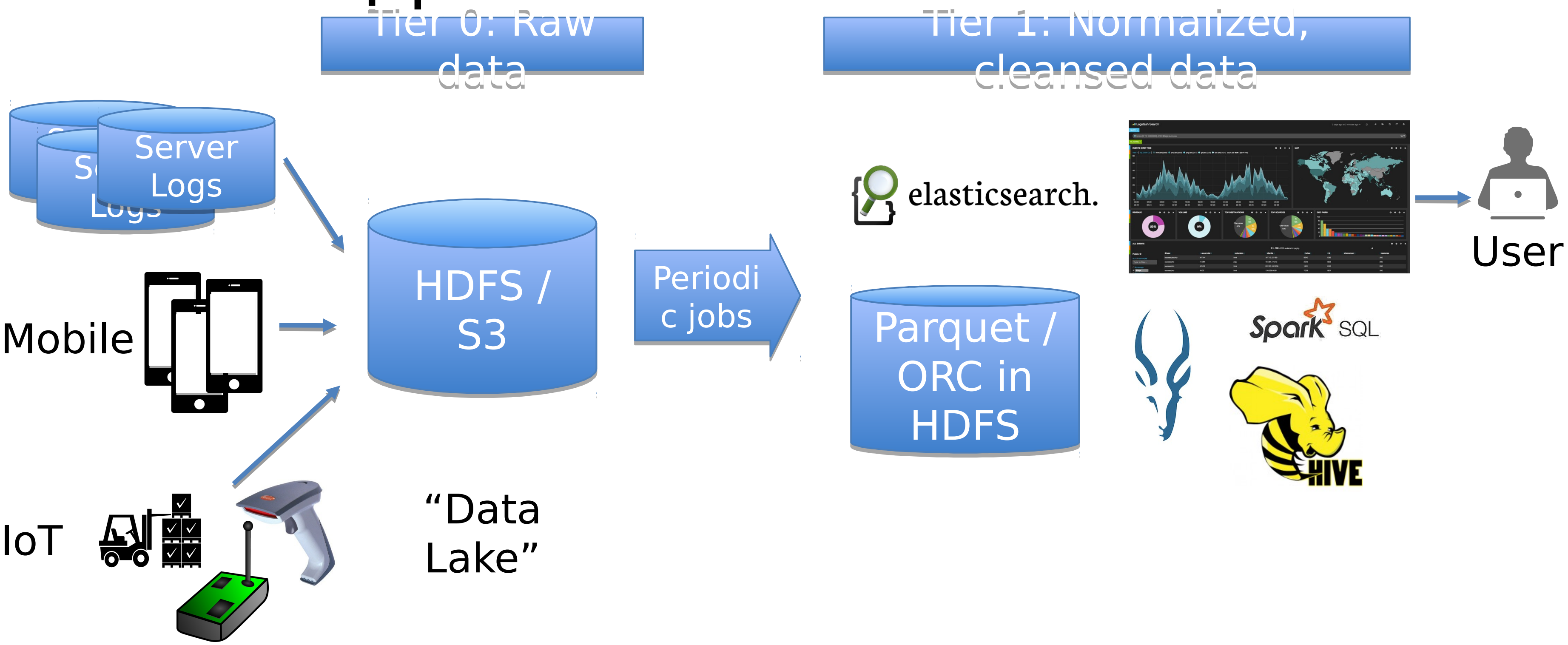
- ETL: Move data from A to B and transform it on the way
- Old approach:



“Data Lake”

Extract, Transform, Load (ETL)

- ETL: Move data from A to B and transform it on the way
- Old approach:



Tier 0: Raw data

Tier 1: Normalized, cleansed data

Server Logs

Mobile

IoT

HDFS / S3

Periodic jobs

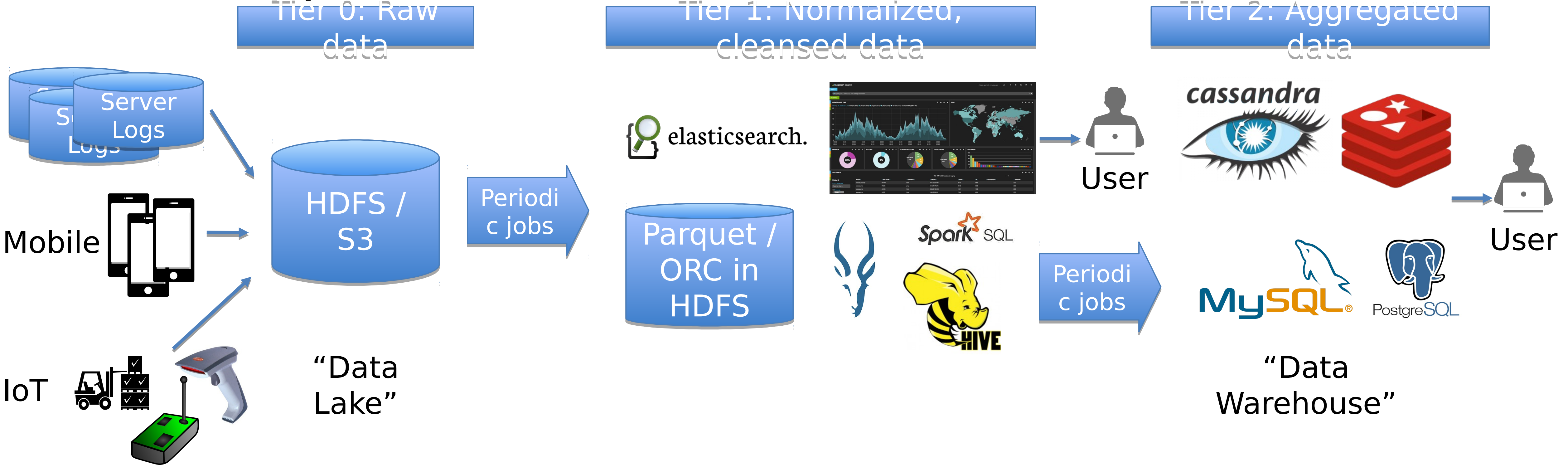
Parquet / ORC in HDFS

“Data Lake”

User

Extract, Transform, Load (ETL)

- ETL: Move data from A to B and transform it on the way
- Old approach:



Tier 0: Raw data

Tier 1: Normalized, cleansed data

Tier 2: Aggregated data

Server Logs

Mobile

IoT

HDFS / S3

"Data Lake"

Periodic jobs

Parquet / ORC in HDFS

User

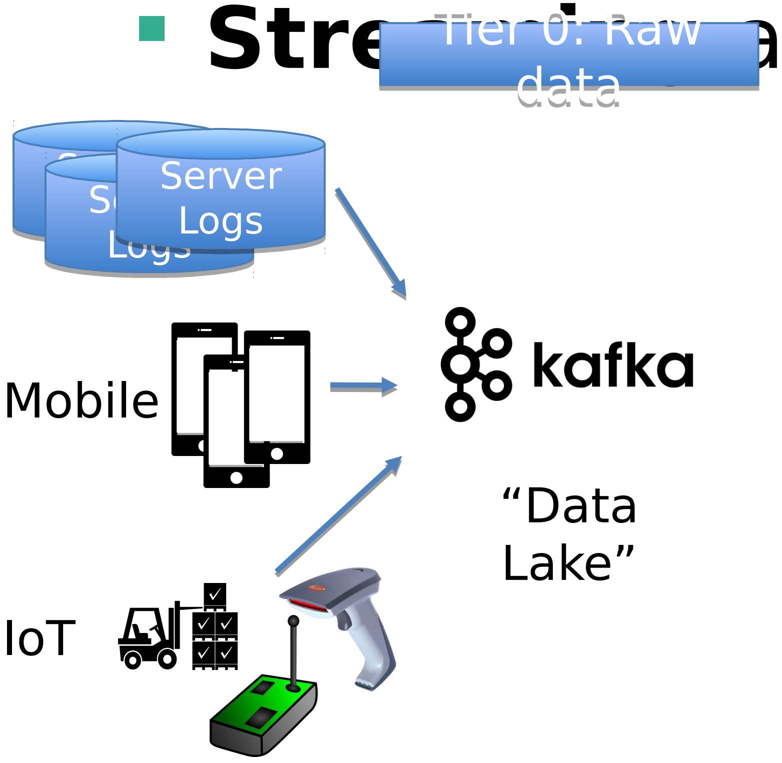
Periodic jobs

User

"Data Warehouse"

Extract, Transform, Load (**Streaming** ETL)

- ETL: Move data from A to B and transform it on the way
- **Streaming** approach:



Tier 0: Raw data

Server Logs

Server Logs

Server Logs

Mobile

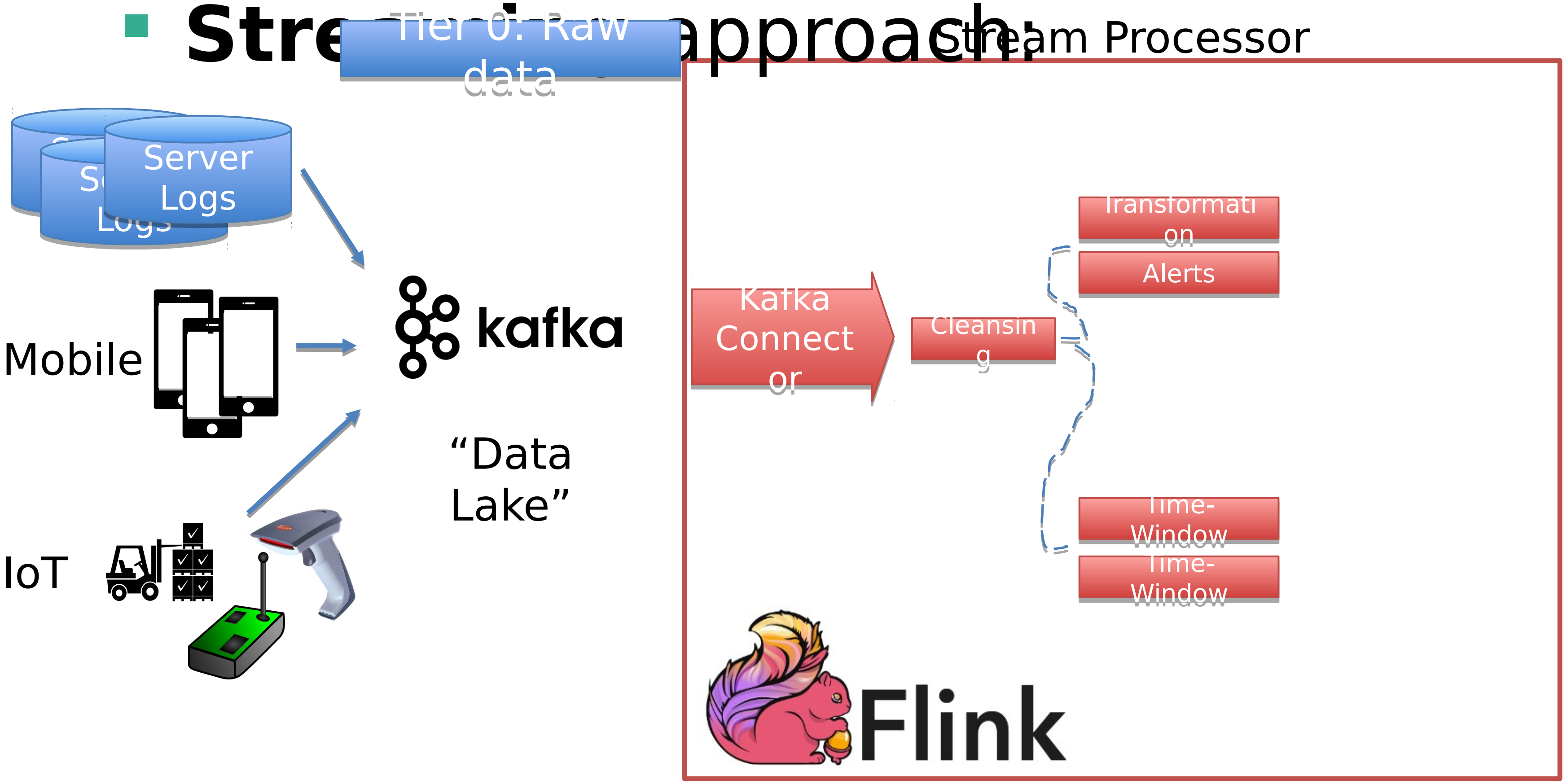
IoT

kafka

“Data Lake”

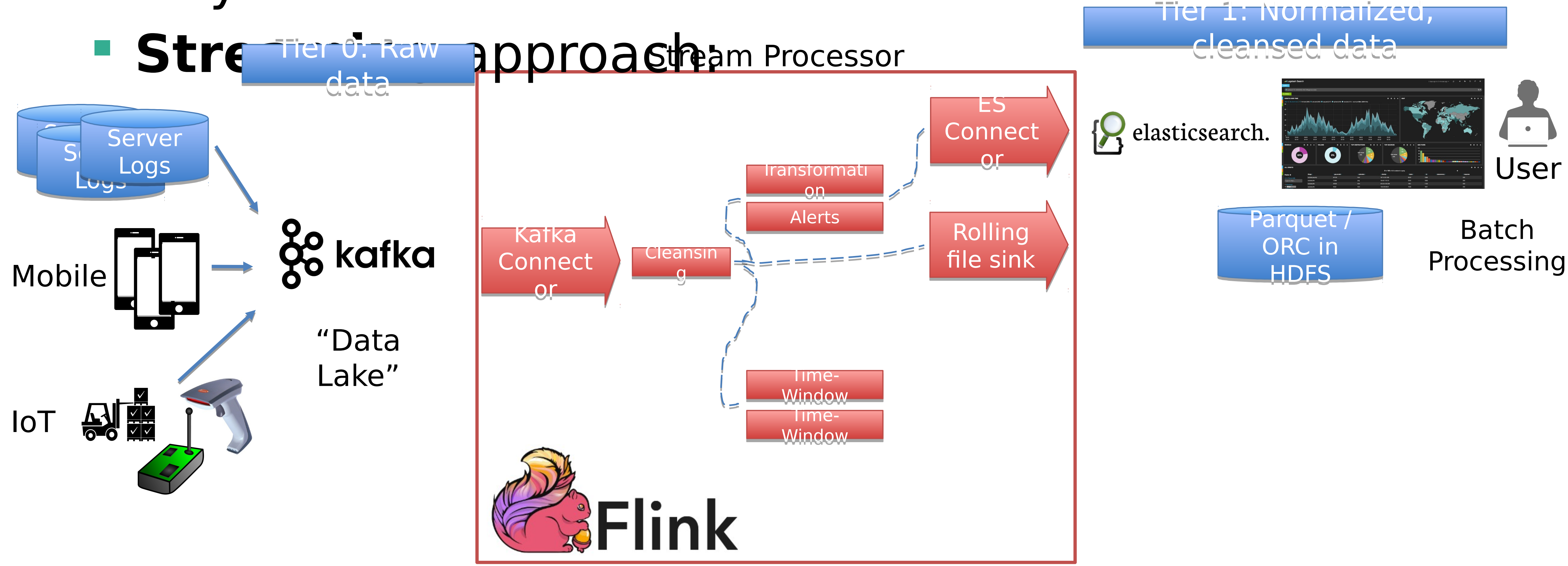
Extract, Transform, Load (**Streaming** ETL)

- ETL: Move data from A to B and transform it on the way
- **Streaming** approach:



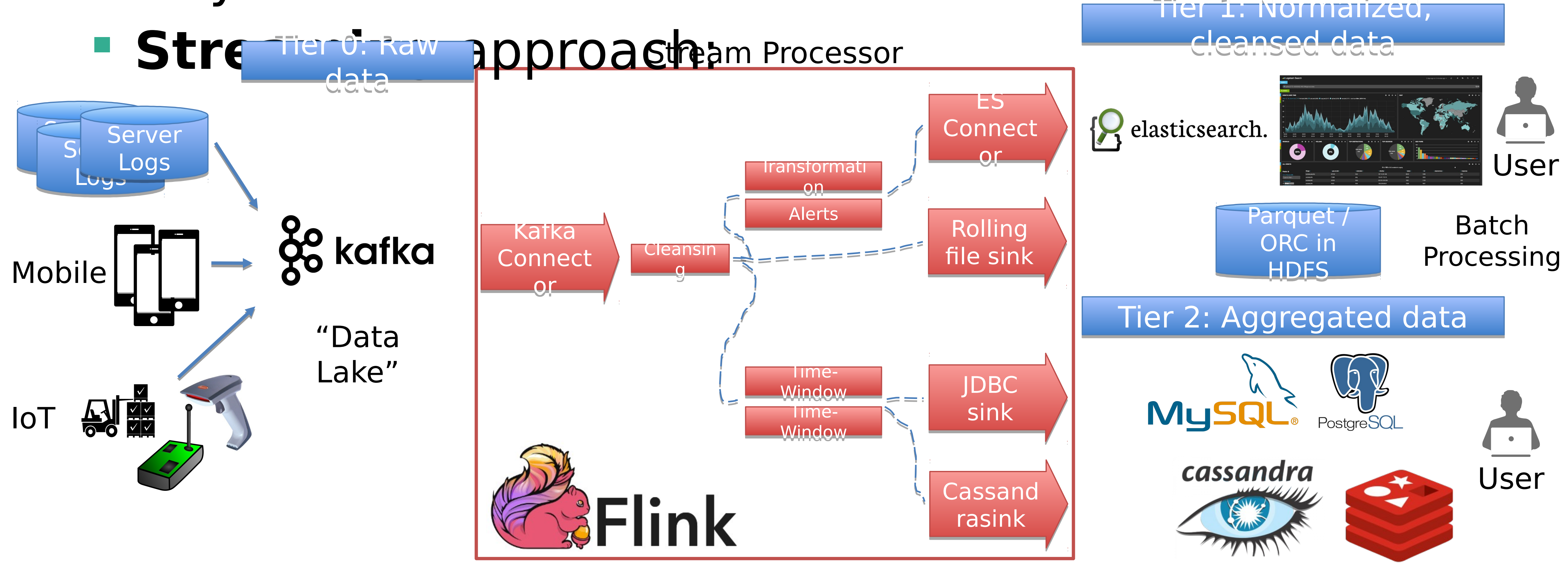
Extract, Transform, Load (**Streaming** ETL)

- ETL: Move data from A to B and transform it on the way
- **Streaming approach:**



Extract, Transform, Load (**Streaming** ETL)

- ETL: Move data from A to B and transform it on the way
- **Streaming approach:**

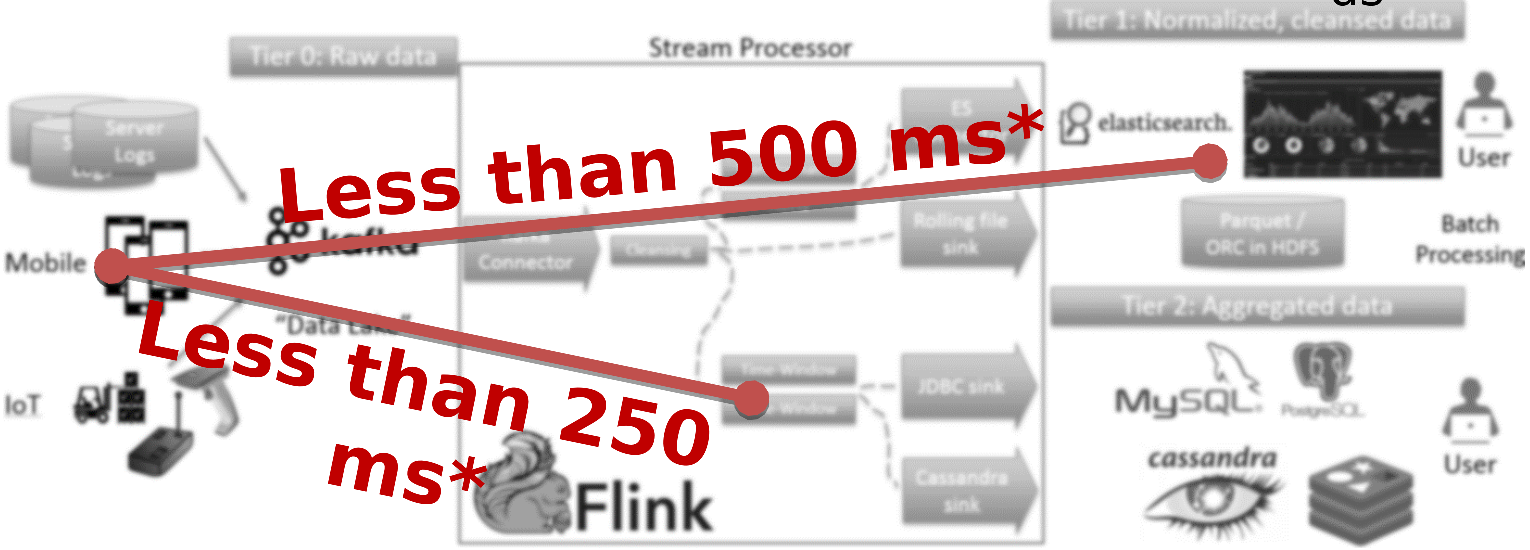


Streaming ETL: Low Latency

- Events are processed immediately
  - No need to wait until the next “load” batch job is running

Approach Periodic batch job | Batch processor with micro-batches | Stream processor

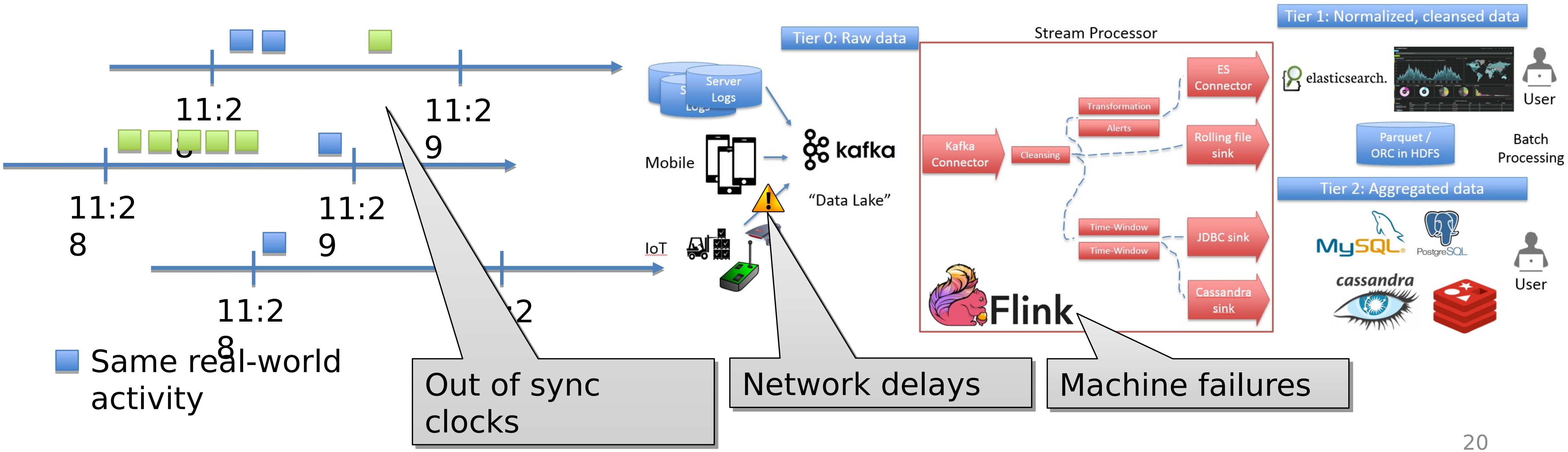
Latency hours | minutes | seconds | milliseconds



* Your mileage may vary. These are rule of thumb estimates.

Streaming ETL: Event-time aware

- Events derived from the same real-world activity might arrive out of order in the system
- Flink is event-time aware



🟦 Same real-world activity

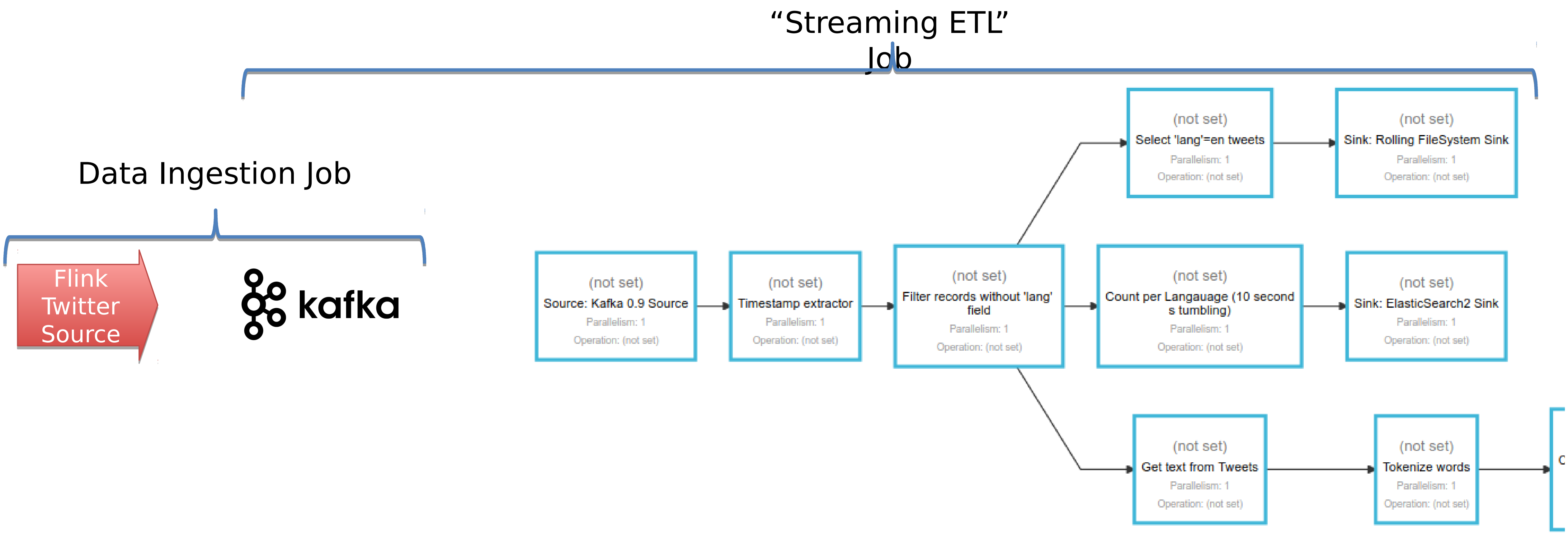
Out of sync clocks

Network delays

Machine failures

Demo

Job Overview



"Streaming ETL" Job

Data Ingestion Job

Flink Twitter Source

kafka

(not set)
Source: Kafka 0.9 Source
Parallelism: 1
Operation: (not set)

(not set)
Timestamp extractor
Parallelism: 1
Operation: (not set)

(not set)
Filter records without 'lang' field
Parallelism: 1
Operation: (not set)

(not set)
Select 'lang'=en tweets
Parallelism: 1
Operation: (not set)

(not set)
Sink: Rolling FileSystem Sink
Parallelism: 1
Operation: (not set)

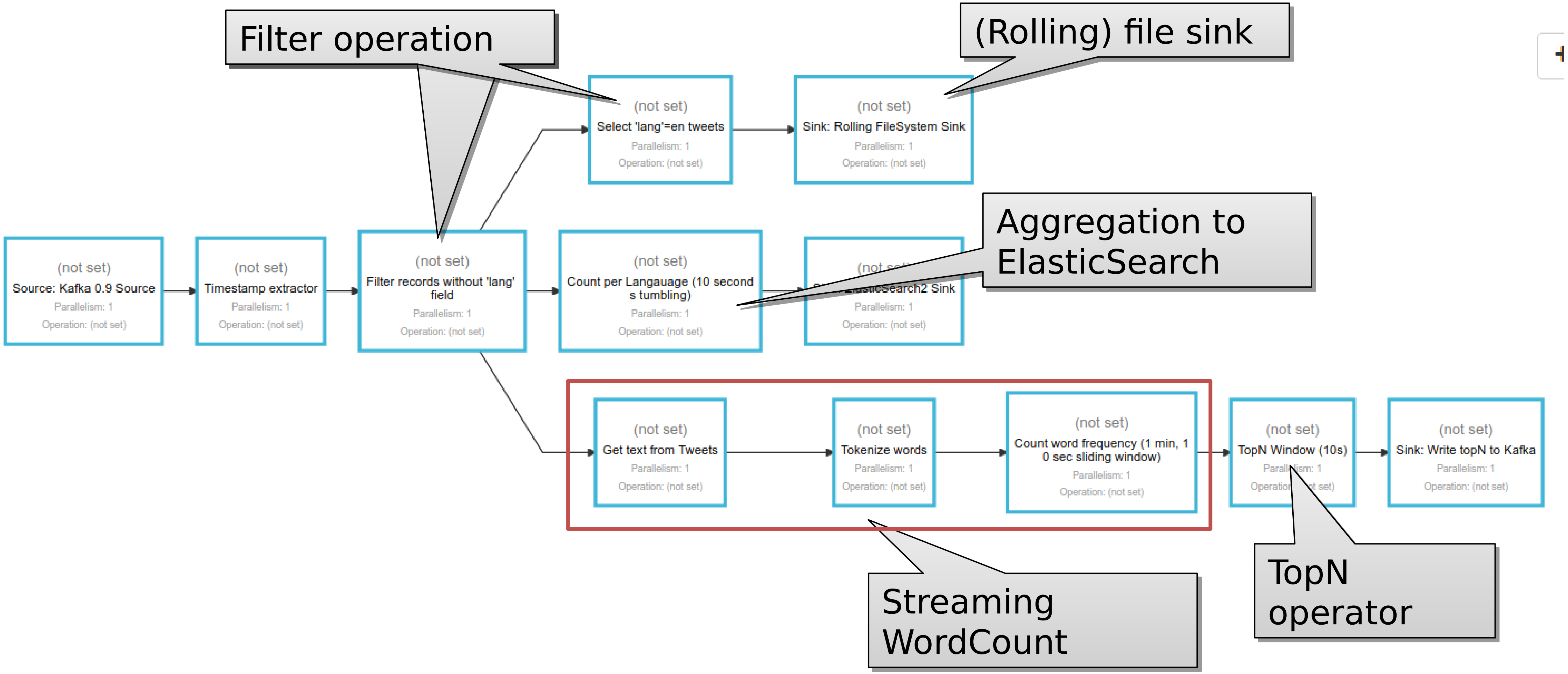
(not set)
Count per Langauage (10 seconds tumbling)
Parallelism: 1
Operation: (not set)

(not set)
Sink: ElasticSearch2 Sink
Parallelism: 1
Operation: (not set)

(not set)
Get text from Tweets
Parallelism: 1
Operation: (not set)

(not set)
Tokenize words
Parallelism: 1
Operation: (not set)

Job Overview



Filter operation

(Rolling) file sink

Aggregation to ElasticSearch

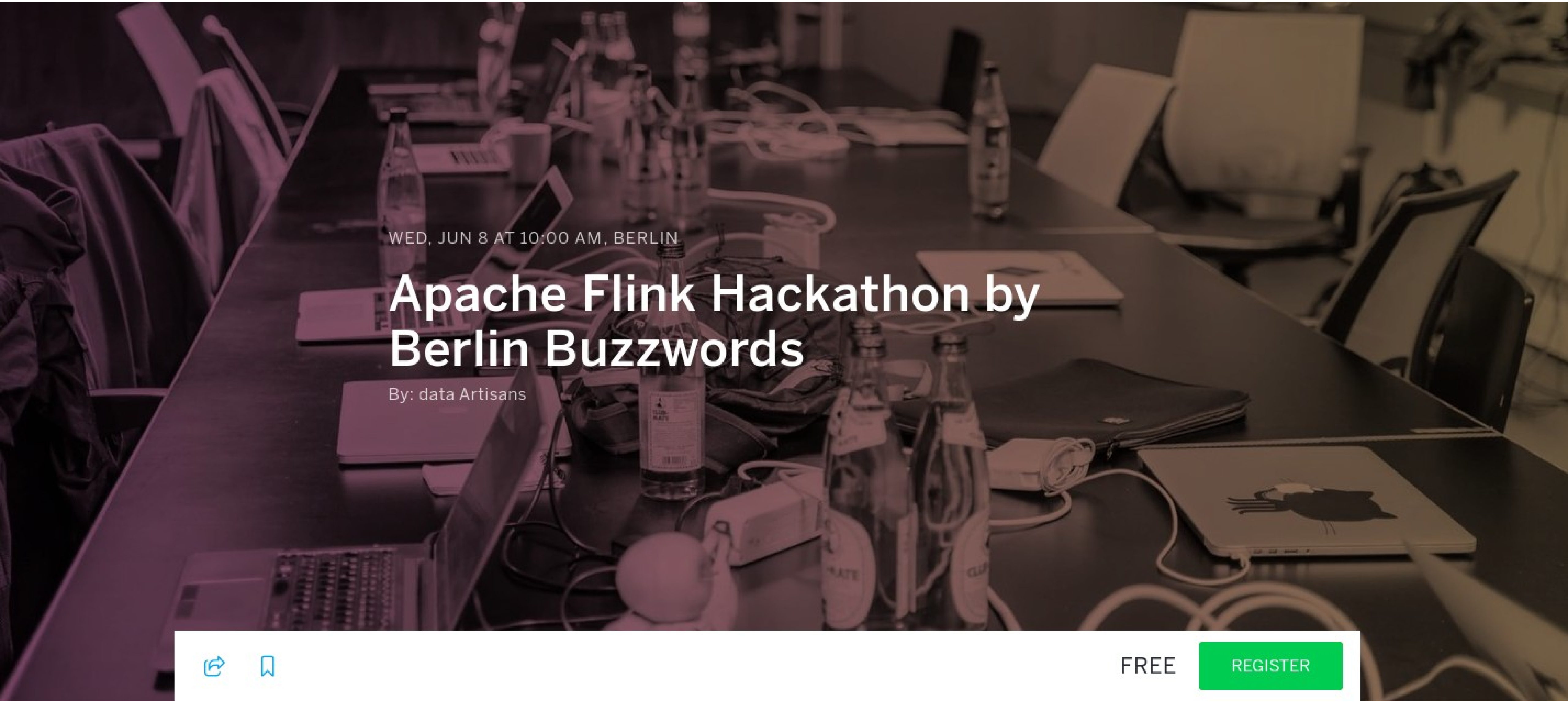
Streaming WordCount

TopN operator

Demo code @ GitHub

https://github.com/rmetzger/flink-streaming-etl

Closing

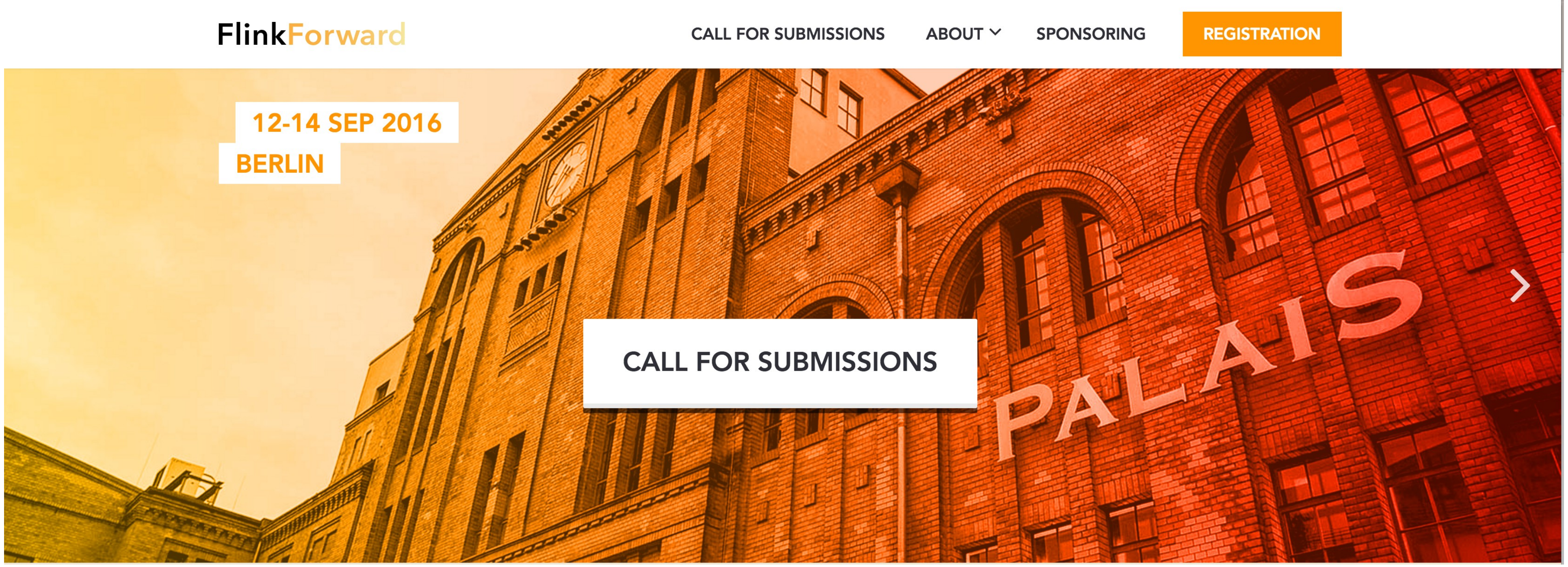


WED, JUN 8 AT 10:00 AM, BERLIN

Apache Flink Hackathon by Berlin Buzzwords

By: data Artisans

https://www.eventbrite.com/e/apache-flink-hackathon-by-berlin-buzzwords-tickets-25580481910



Flink Forward 2016, Berlin

Submission deadline: June 30, 2016
Early bird deadline: July 15, 2016

www.flink-forward.org

dataArtisans

We are hiring!

data-artisans.com/careers

Questions?

- Ask now!
- eMail: rmetzger@apache.org
- Twitter: @rmetzger_

- Follow: @ApacheFlink
- Read: flink.apache.org/blog, data-artisans.com/blog/
- Mailinglists: (news | user | dev)@flink.apache.org

Appendix

Sources

- “Large scale ETL with Hadoop” http://www.slideshare.net/OReillyStrata/large-scale-etl-with-hadoop