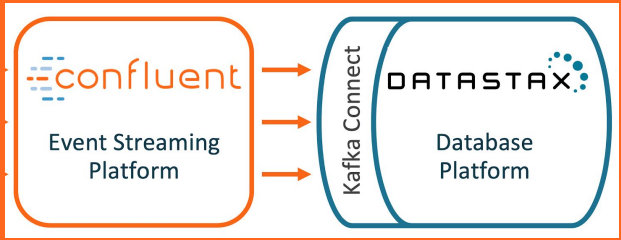
confluent

Kafka meets Cassandra:



Architectural Patterns & Integration Points

Mirko Kämpf, Solutions Architect @ Confluent

Abstract:

Decision support systems are needed in any kind of business. They combine the results of intense research and modern IT technology. Such systems are important aspects of the digital transformation and require the integration of data from multiple domains, using various technologies. This ranges from data capturing over automatic data enrichment to autonomous control systems.

The Big-Data hype and the Cloud movement have shown impressive impact already, but we have to go further: ***Event Driven Architecture*** (EDA) should become a commodity, so that data engineering and data science can be integrated easier and more efficiently.

Besides large scale batch processing and efficient micro-batching we have to go on towards scalable stream processing in which analysis and ML components can be plugged in. This approach reduces that type of latency which isn't caused by slow networks but by badly designed processes and old fashioned software solutions.

In this talk I describe a cloud native architecture for advanced time series analysis and graph analysis using Apache Kafka and Apache Cassandra. I will show typical integration points for both technologies and some generic building blocks for simple but robust streaming solutions, applicable to industry and research in many domains.

Intro:

Understanding of mechanisms behind a data driven economy requires analysis of their core components. Just collecting more and more data, and learning fancy models is not enough! Decisions have to be supported by repeatable data analysis and reliable experiments.

The *lean management approach* is based on the scientific method although it is focused on managing a company or an organisation rather than just producing papers it needs hypothesis and structured procedures.

This means, new IT solutions should support our decision makers by enabling the scientific management methods based on scalable information processing and knowledge management.

Having this in mind, we can see that we need a robust approach for tracking the state of complex systems, especially for those which will emerge in the near future!

Background: Complex Systems Research

Combination of various fields:

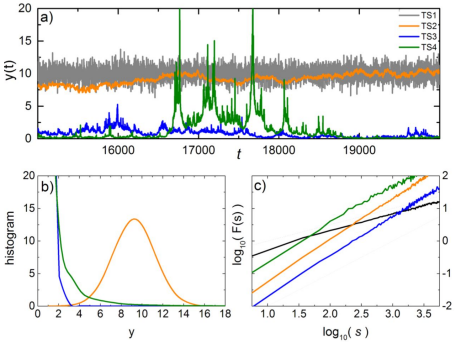
- **Math, Statistics, Physics**
- **Data Visualization**
- **Software- & Data engineering**
- **IT operations**

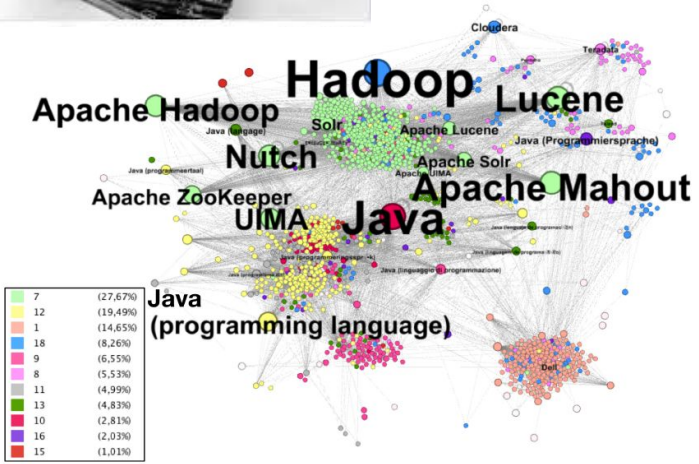
Research Scope:

Analyse usage patterns of Wikipedia pages in different contexts:

- by language
- by topic
- contribution vs. consumption (edit vs. read only)

MARTIN-LUTHER
UNIVERSITÄT
HALLE-WITTENBERG





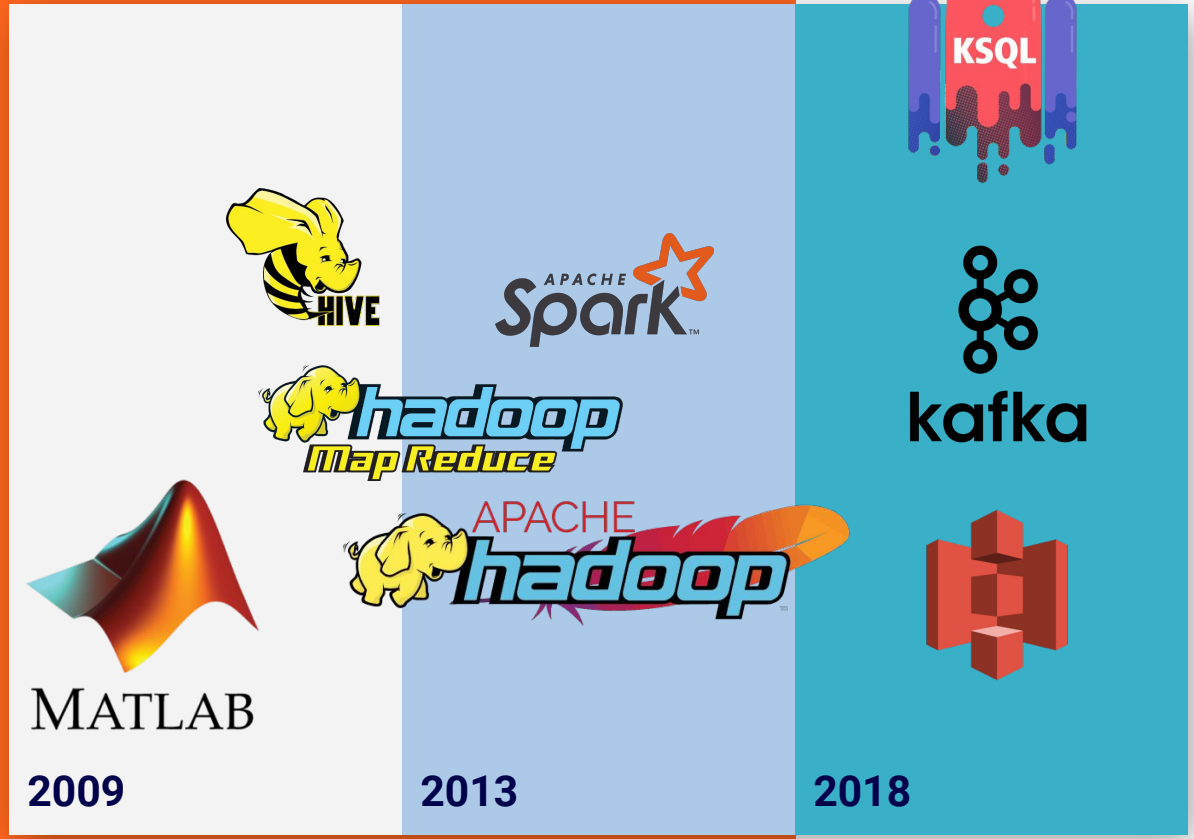
WIKIPEDIA
The Free Encyclopedia

Tech. History

The project is based on software provided by ASF.

System evolved over time in three phases:

Finally it converges in a cloud native streaming platform.



Check Requirements: Why data streaming?

Data sources of multiple types need continuous integration:

- Sources change over time.
- Complex systems have no “end of day”.
- Replay is needed for evolving experimental setups.
- Wikipedia is just an example, but the methods can be generalized to: **industry**, **supply chains**, **traffic control**, and **financial markets**:

The data flies in those contexts !

We’d rather get ready...

confluent

Apache Kafka in 10 min

PART 1:

APACHE kafka®

A distributed streaming platform

PUBLISH & SUBSCRIBE

Read and write streams of data like a messaging system.

PROCESS

Write scalable stream processing applications that react to events in real-time.

STORE

Store streams of data safely in a distributed, replicated, fault-tolerant cluster.

https://kafka.apache.org/

All your data as a stream of events

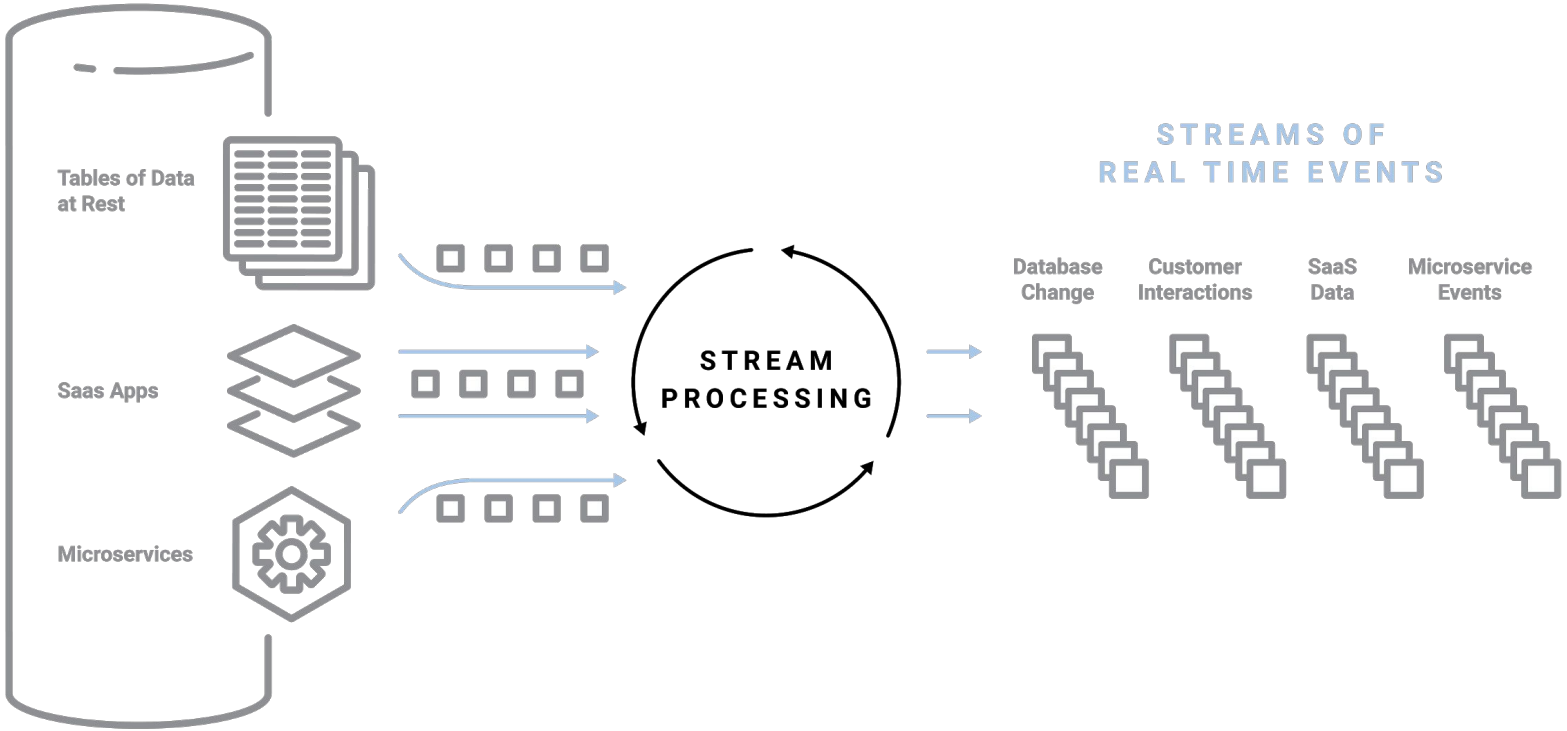
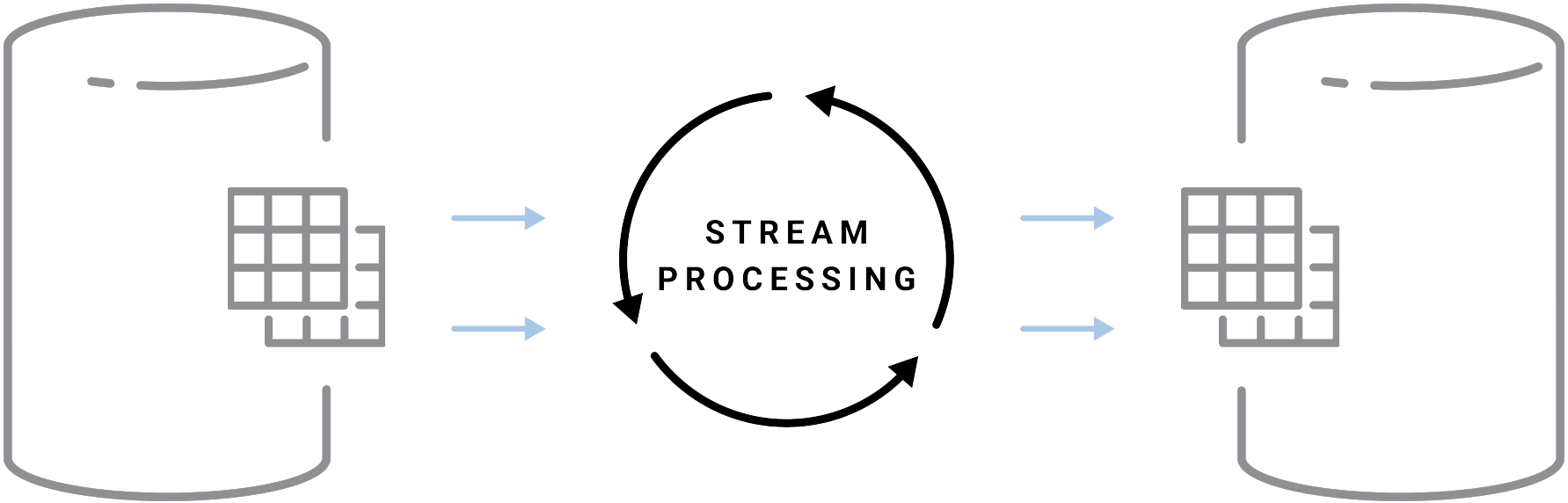
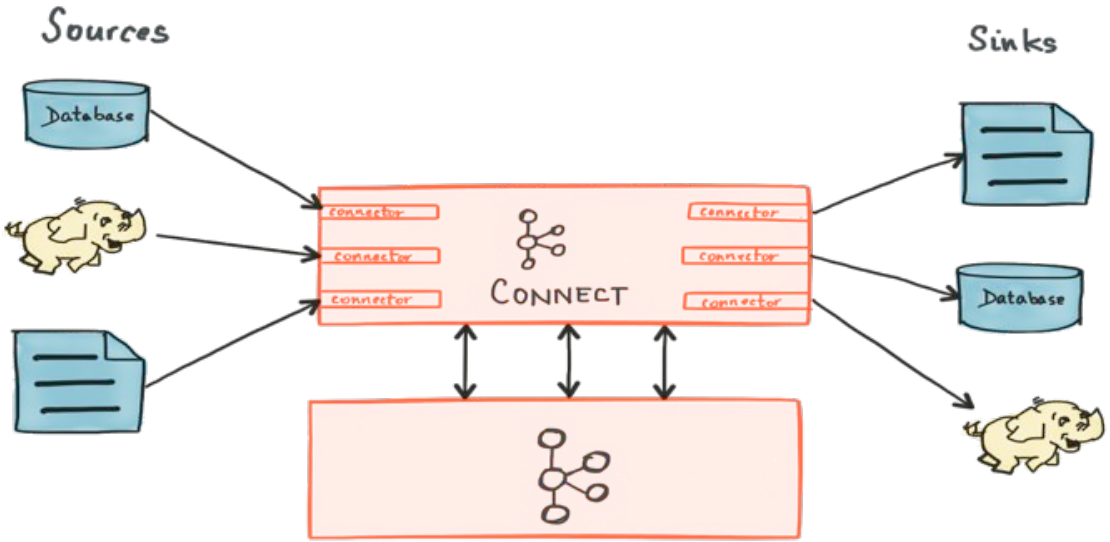


Table Stream Duality



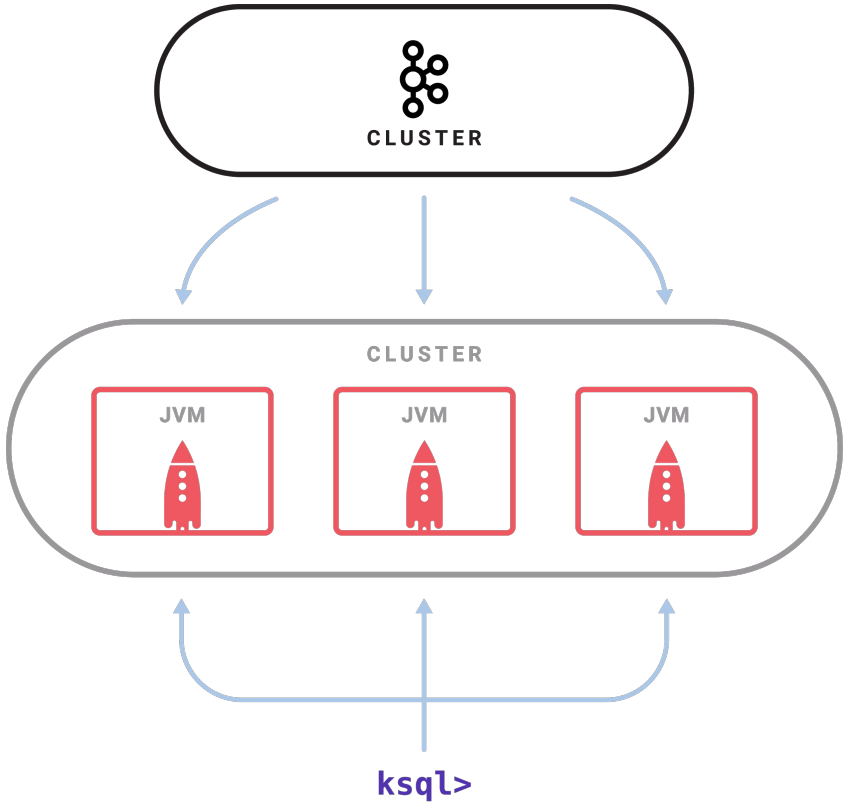
Kafka Connect
Overview:



- Connectors can run on single hosts or in a cluster.
- Reusable components save you a lot of development time.
- Connectors enable automatic of data flows between different technologies.

KSQL Overview:

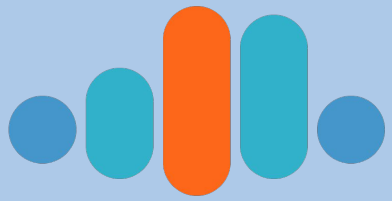
- You can submit queries using an interactive SQL command line client
- Several continuous queries run in parallel on a KSQL cluster
- Adding more server processes scales a KSQL cluster



What is the Confluent Platform?

Apache Kafka:

- scalable **event streaming** platform
- a **scalable** event **processing** platform
- **data flow integration** platform
- a system for **real time data** analytics



PLUS: Additional features & services to help enterprise customers

Kafka for Research: The Challenge

How can we combine unbound data assets & new algorithms?

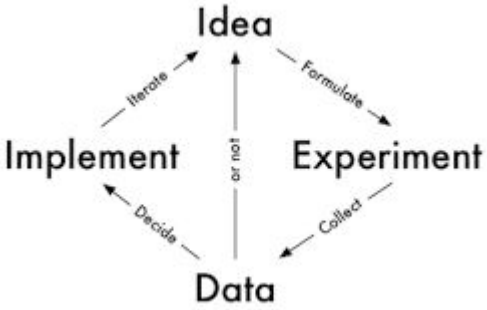
A. you pipe the data to the place where it can be processed easily,
e.g., to the cloud where it is consumed by special purpose systems.

B. you integrate new complex algorithms in your processing pipeline

Miss conceptions regarding Kafka and its Ecosystem …

- are considered to be middleware, managed by IT people:
  - Typically, researchers do not plan nor implement their experiments using such a technology.
- don’t offer ML / AI components:
  - many people think, that a predictive models have to be executed on edge-device or in the cloud.

Kafka supports agile experiments …



- it gives access to data (flows) in real time,
  - in a way, which allows a replay of experiments at a later point in time
  - Confluent cloud is Apache Kafka completely managed by Confluent ready to be used by “non IT people”.

- allows **variation of analysis without redoing the same experiment** by simply reusing the persisted event-stream again.

- Kafka Streams and KSQL allow data processing in place
  - this allows faster iterations because plausibility checks can be done in place
  - the streaming API gives freedom for extension
  - DSL and KSQL save you a lot of time

confluent

Kafka meets Cassandra

Architectural Patterns & Integration Points

PART 1:

Architecture for

Advanced Time Series Analysis & Graph Processing

using Kafka & Cassandra:

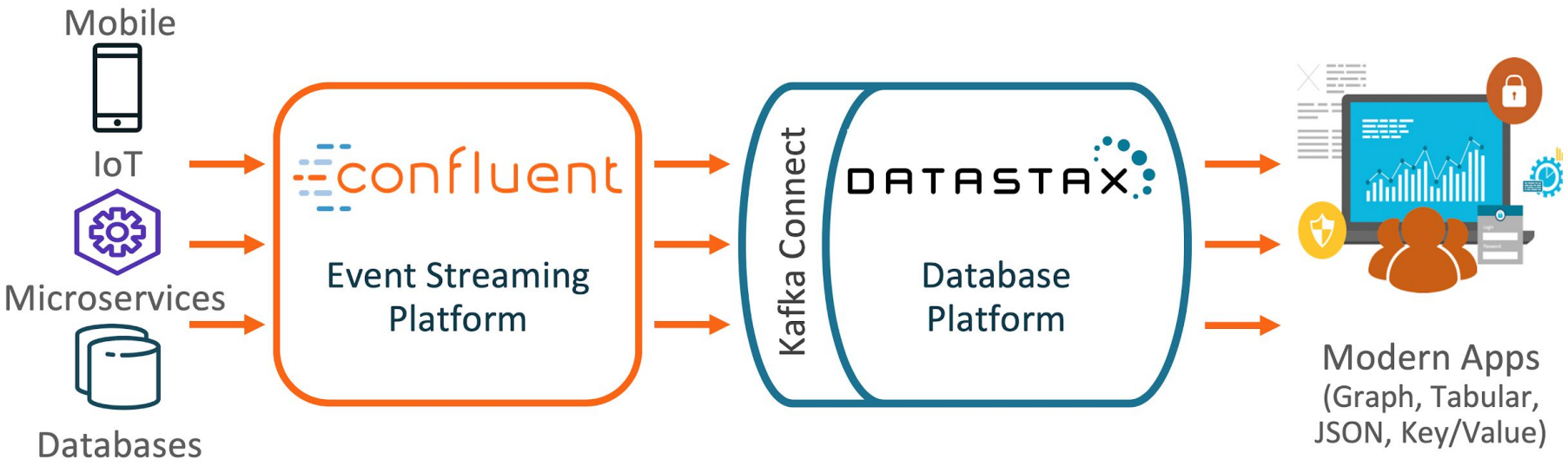
Why Architecture?

- Identify Patterns &
  Define Building blocks

- Provide Guard Rails

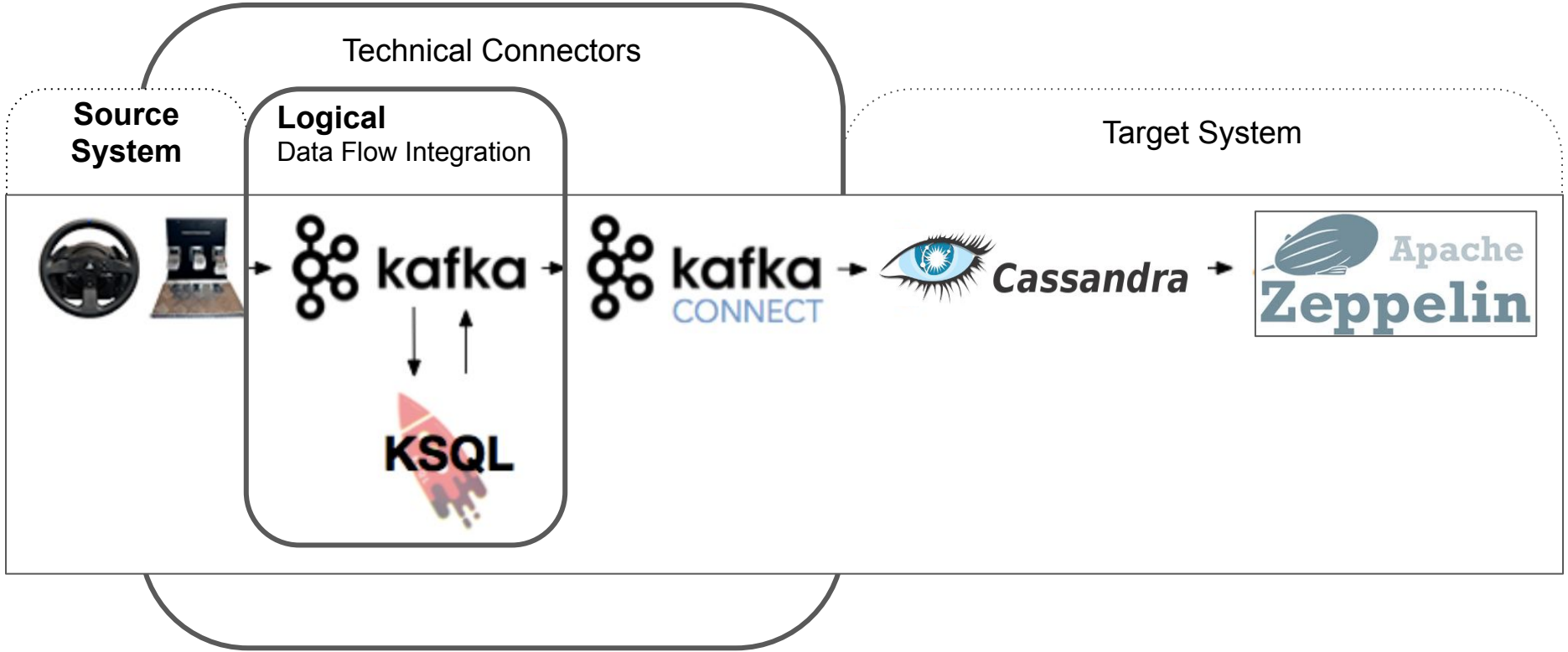
- Simplify

Data Flow Pattern: Linear Flow

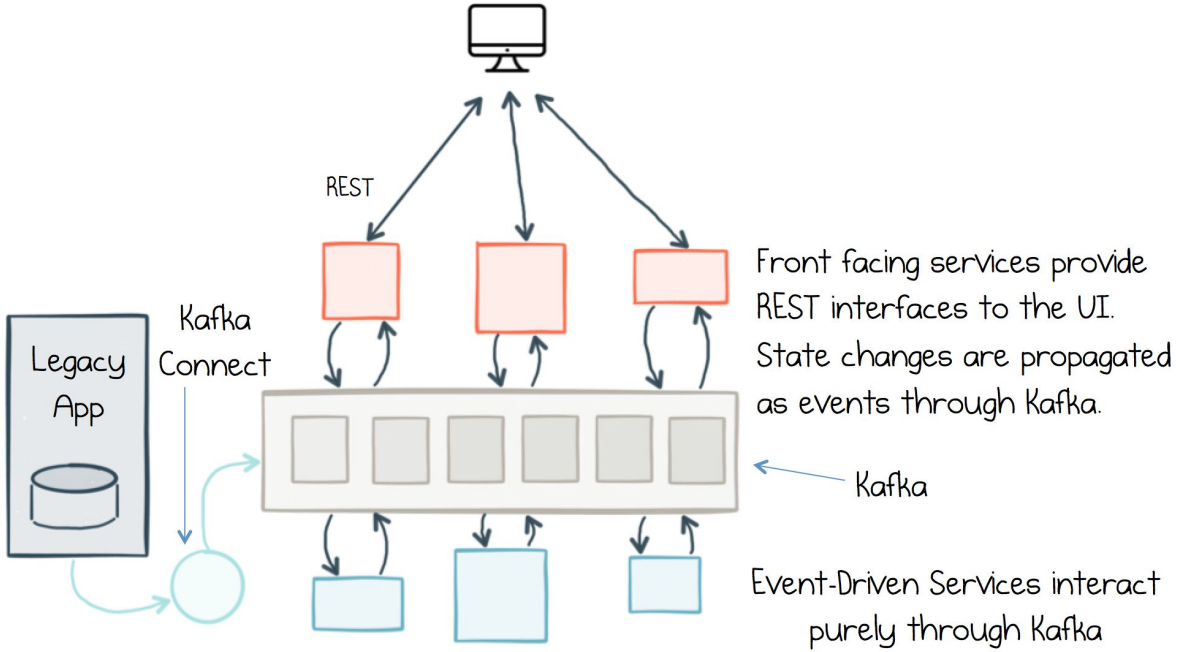


Data Flow Pattern:

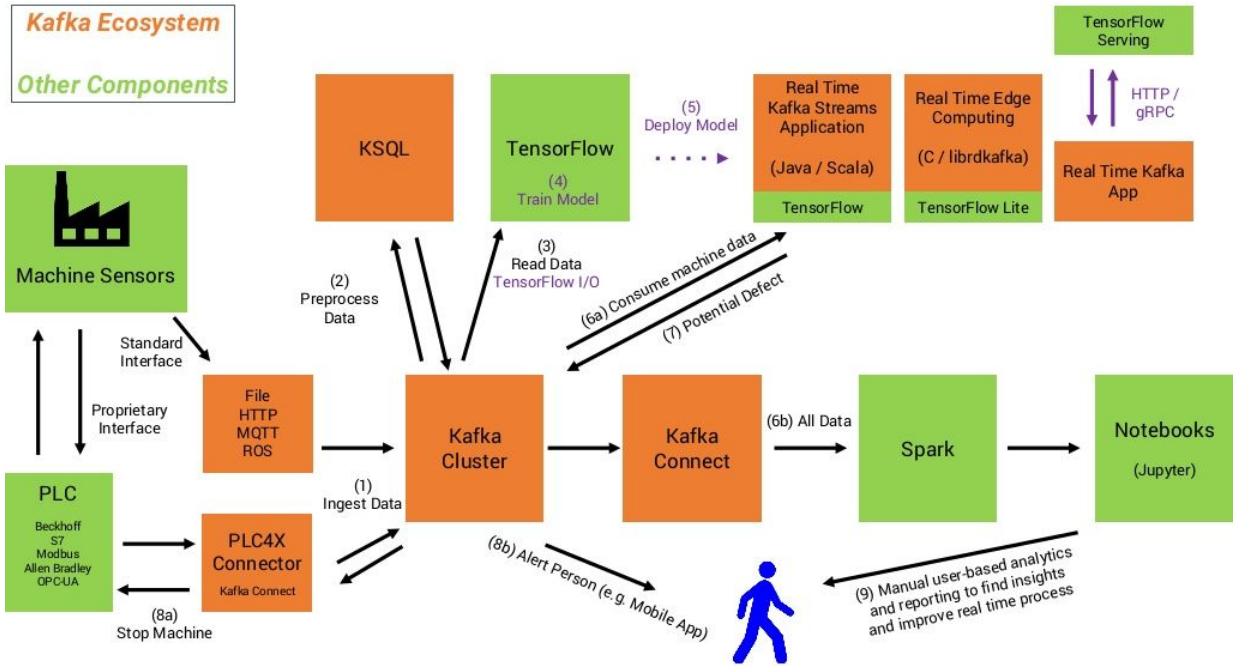
Linear Flow + In Flight Enrichment + Interactive ML



Data Flow Pattern:
Bi-directional Information Stream in EDA



Scenario: Supply Chain Optimization



https://www.slideshare.net/KaiWaehner/iiot-industry-40-with-apache-kafka-connect-ksql-apache-plc4x

Those examples illustrate the variety
of interconnected components.

The resulting complexity is a risk for projects!

Architects have to simplify …

… and the Kafka ecosystem helps us on this journey!

Building Blocks: for Stream based Apps

Domain specific logic is implemented in *reusable components*:

UDFs:

Domain Driven Design

SourceConnectors
Integrate input side …

Kafka Connect

Legacy and Future Systems

Data flows are no longer transient.
The *event log acts as single source of truth*.

Data Assets

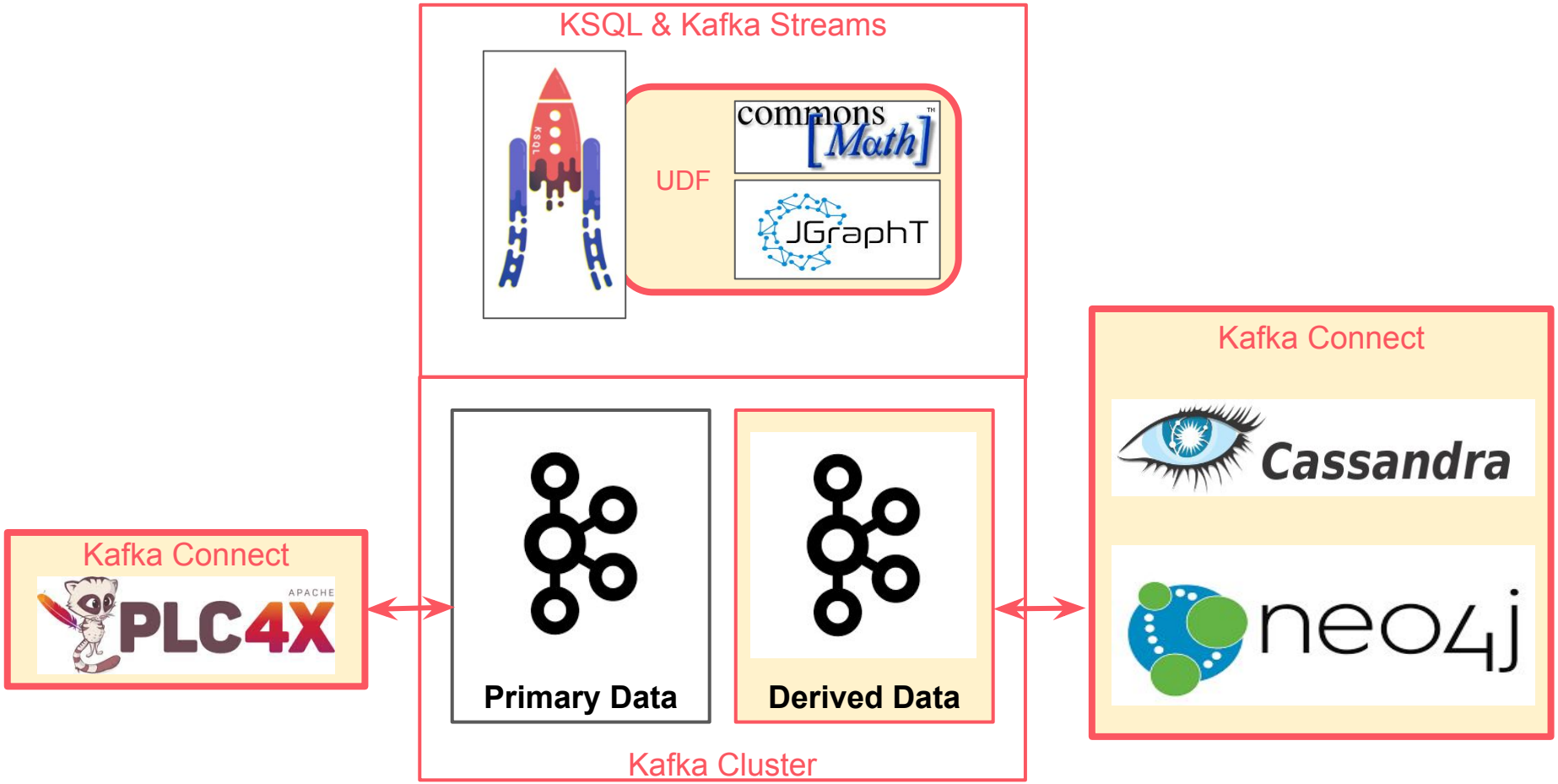
Paradigm Shift in Data Management

SinkConnectors
Integrate output side …

Kafka Connect

Special Purpose Systems

Kafka: Platform for Event/Stream Processing



confluent

Kafka meets Cassandra

Architectural Patterns & Integration Points

PART 2:

Integration Points: Kafka and Cassandra

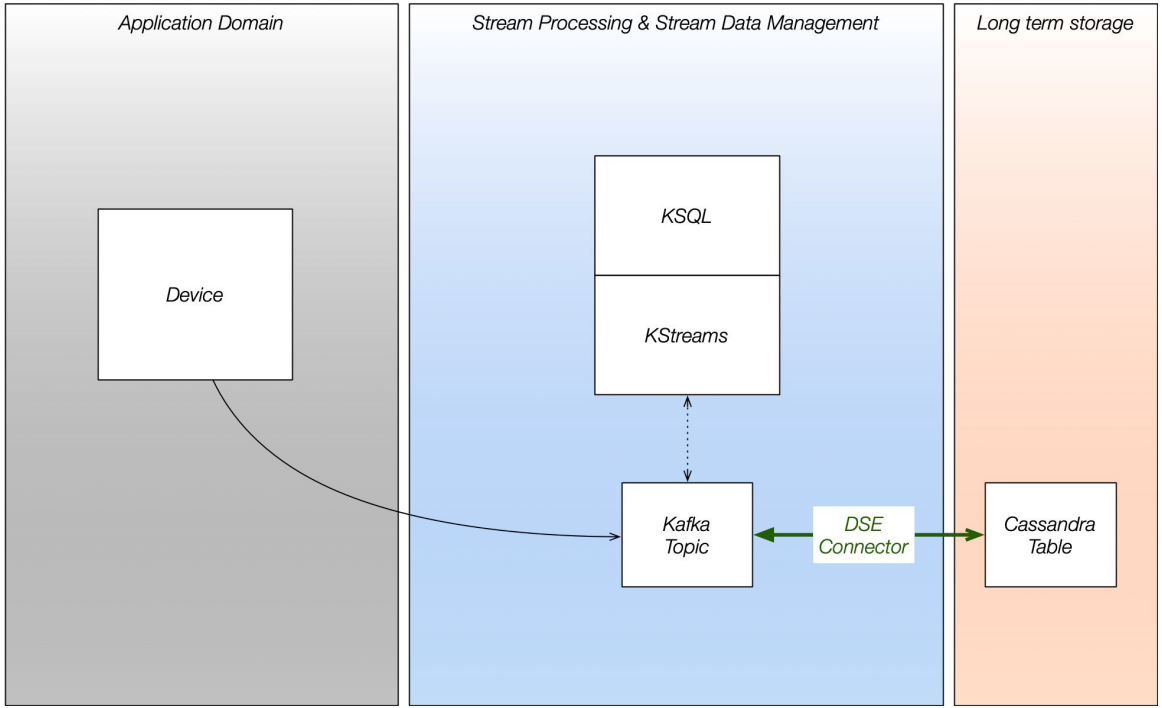
How to use Cassandra alongside your KStreams application?

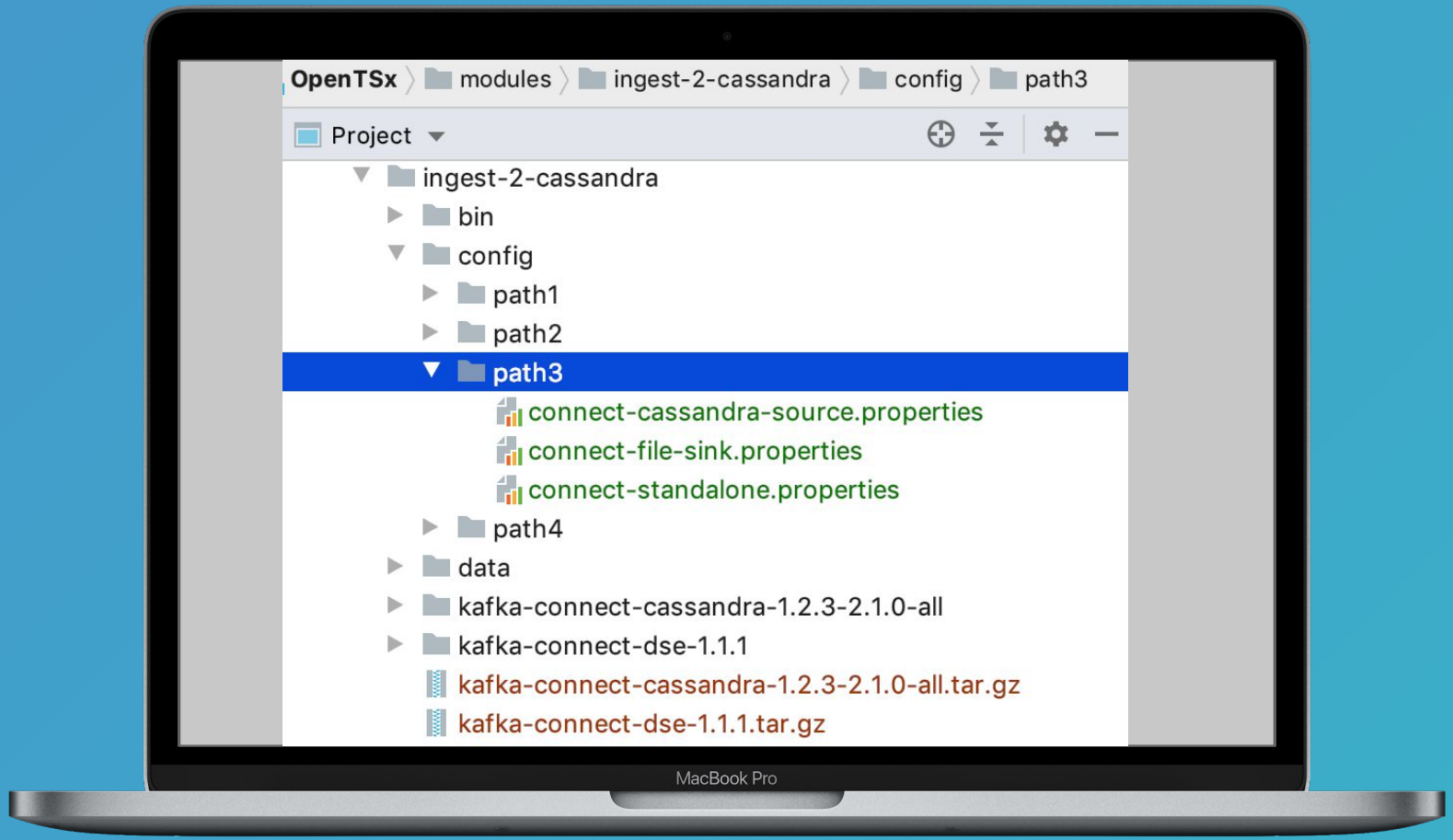
Kafka Connect Sink for Cassandra	**Kafka Connect Source** for Cassandra
KStreams StateStore on Cassandra **KStreams with Lookup** in Cassandra	Cassandra as long term store for **Time Series & Graphs**

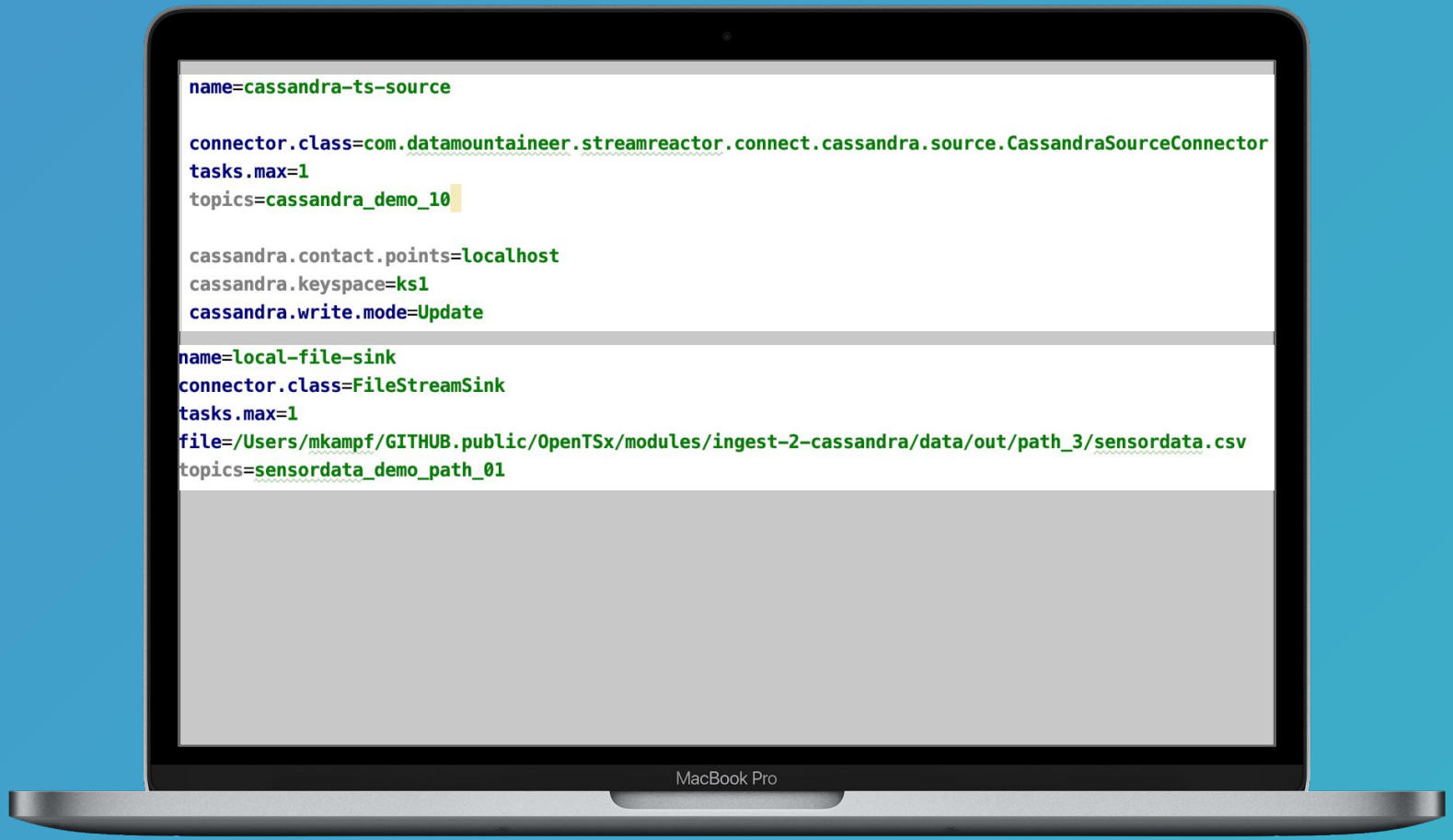
Kafka Connect allows automation of data replication between different technologies.

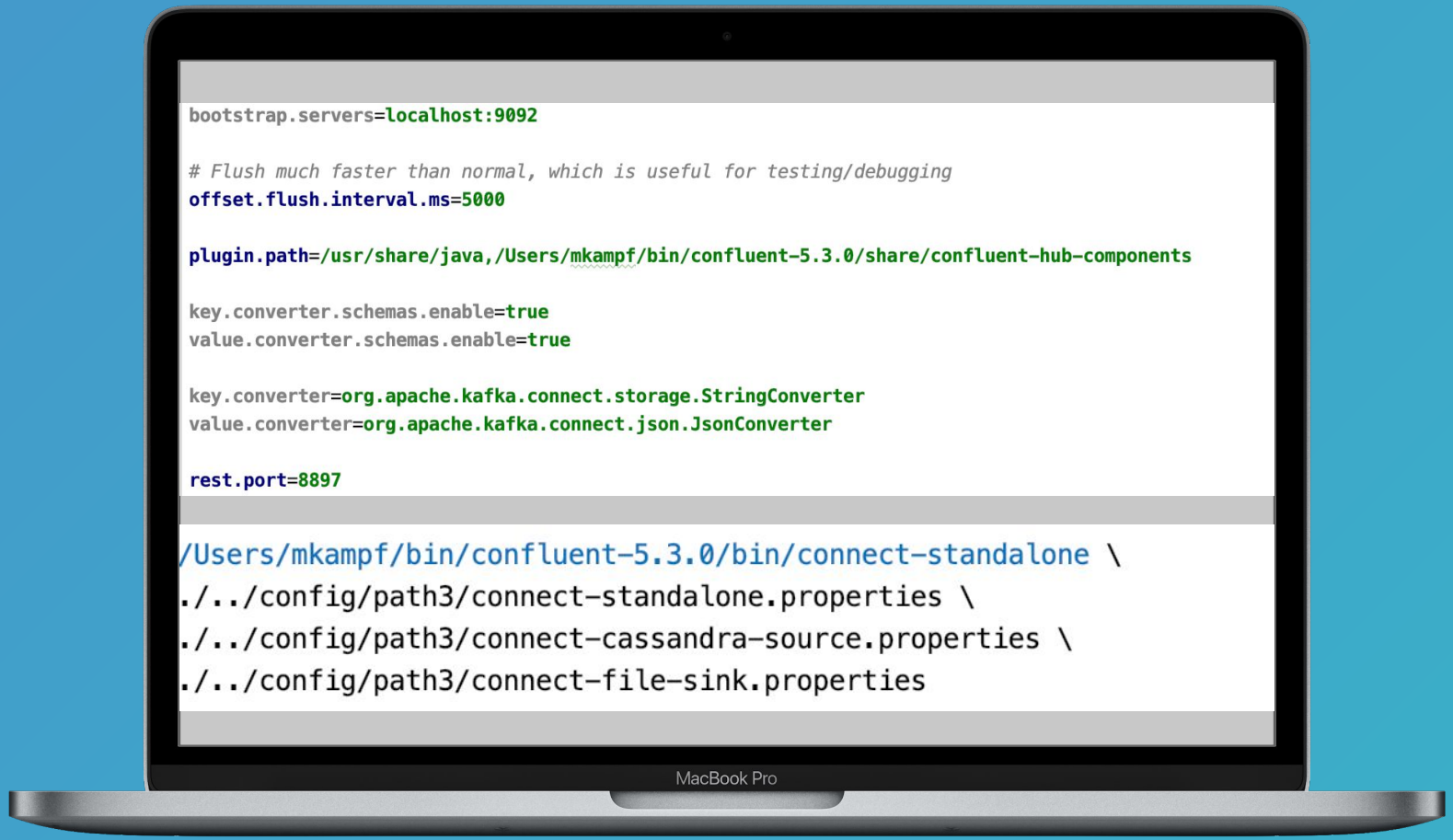
Kafka Connect Sink

pipe **extracted** data and **aggregated** data into a long term storage layer which provides various access paths, beyond those of a file system.




OpenTSx
modules
ingest-2-cassandra
config
path3
Project
ingest-2-cassandra
bin
config
path1
path2
path3
connect-cassandra-source.properties
connect-file-sink.properties
connect-standalone.properties
path4
data
kafka-connect-cassandra-1.2.3-2.1.0-all
kafka-connect-dse-1.1.1
kafka-connect-cassandra-1.2.3-2.1.0-all.tar.gz
kafka-connect-dse-1.1.1.tar.gz
MacBook Pro

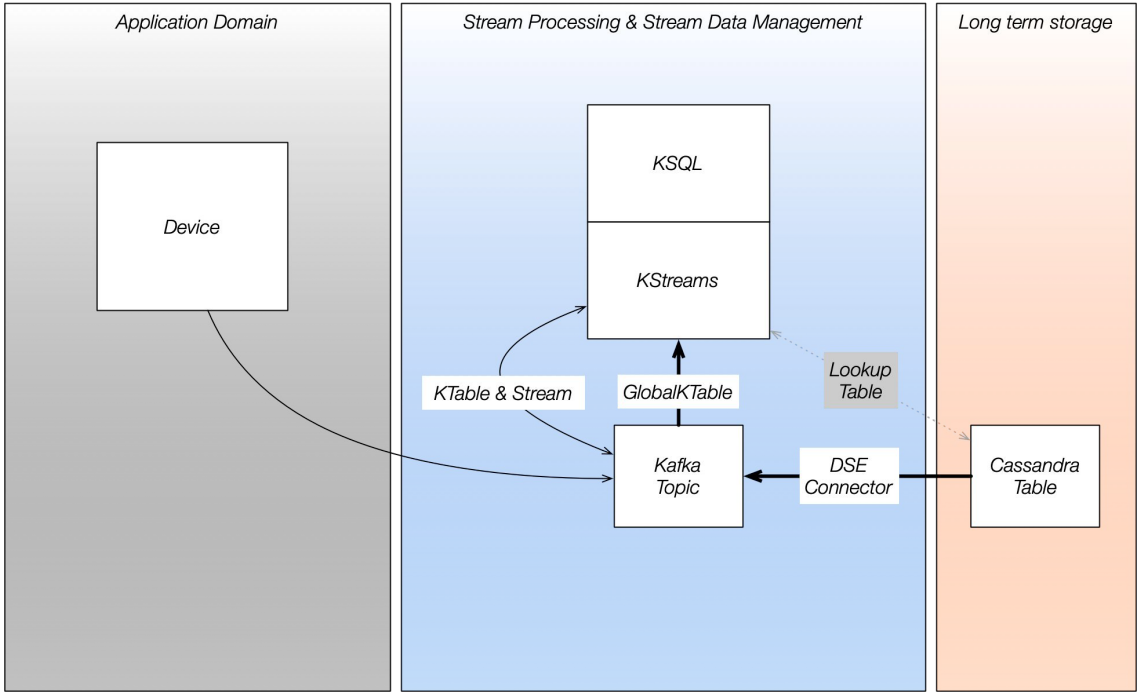

name=cassandra-ts-source
connector.class=com.datamountaineer.streamreactor.connect.cassandra.source.CassandraSourceConnector
tasks.max=1
topics=cassandra_demo_10
cassandra.contact.points=localhost
cassandra.keyspace=ks1
cassandra.write.mode=Update
name=local-file-sink
connector.class=FileStreamSink
tasks.max=1
file=/Users/mkampf/GITHUB.public/OpenTSx/modules/ingest-2-cassandra/data/out/path_3/sensordata.csv
topics=sensordata_demo_path_01
MacBook Pro

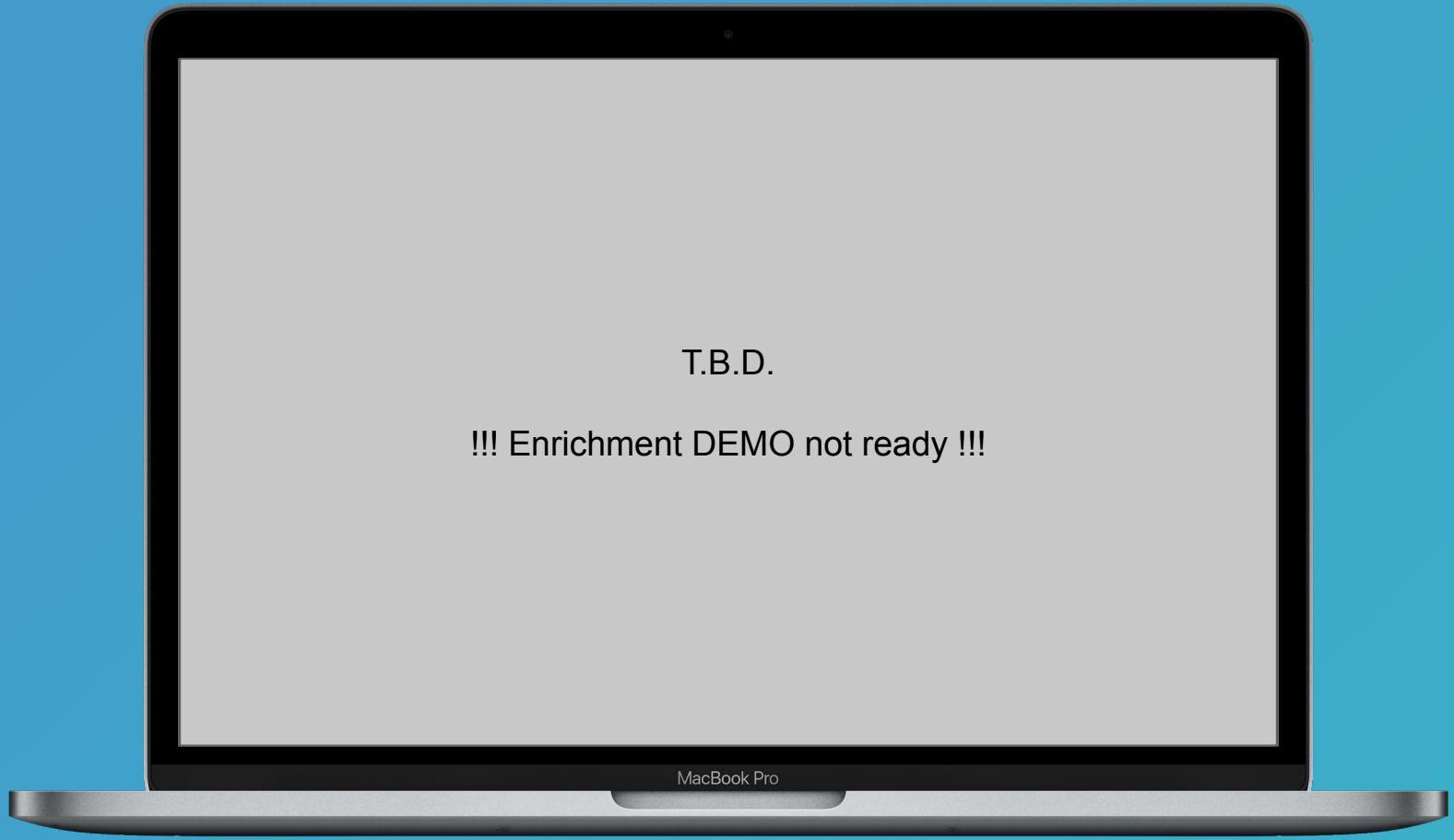

bootstrap.servers=localhost:9092
# Flush much faster than normal, which is useful for testing/debugging
offset.flush.interval.ms=5000
plugin.path=/usr/share/java,/Users/mkampf/bin/confluent-5.3.0/share/confluent-hub-components
key.converter.schemas.enable=true
value.converter.schemas.enable=true
key.converter=org.apache.kafka.connect.storage.StringConverter
value.converter=org.apache.kafka.connect.json.JsonConverter
rest.port=8897
/Users/mkampf/bin/confluent-5.3.0/bin/connect-standalone \
./../config/path3/connect-standalone.properties \
./../config/path3/connect-cassandra-source.properties \
./../config/path3/connect-file-sink.properties
MacBook Pro

Kafka Streams use local copies of lookup tables, based on GlobalKTables.

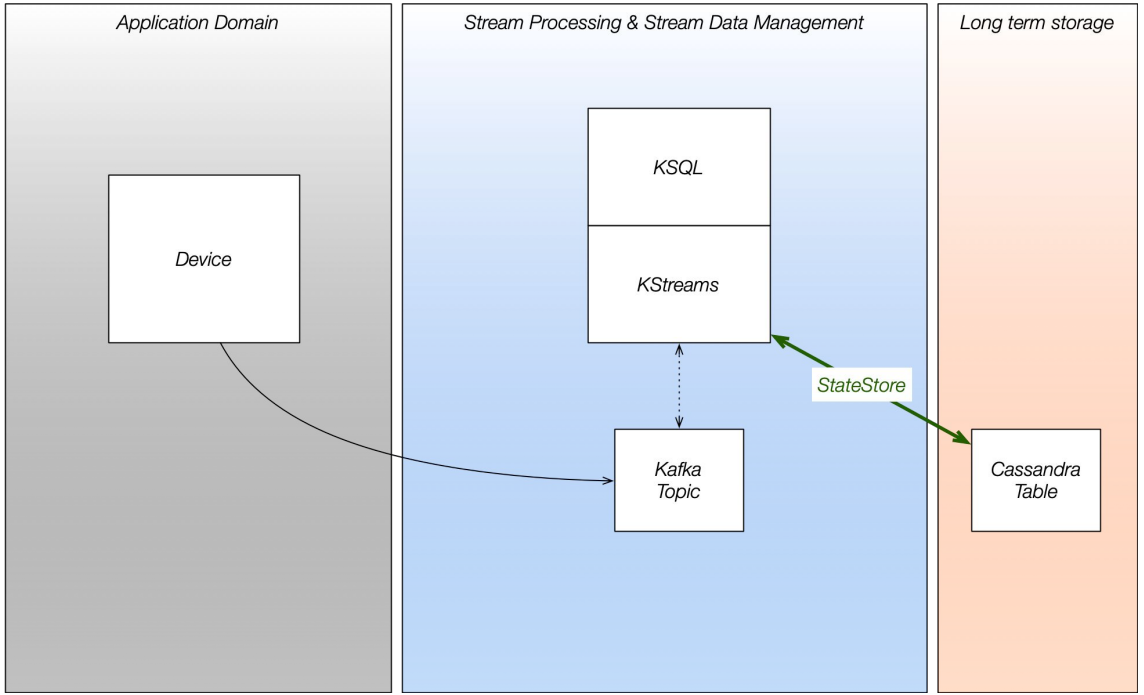
Kafka Connect Source
provide data which is needed for local lookups in KStream applications in Kafka topics.

The ***GlobalKTable*** gives a kind of local access to global data.




T.B.D.
!!! Enrichment DEMO not ready !!!
MacBook Pro

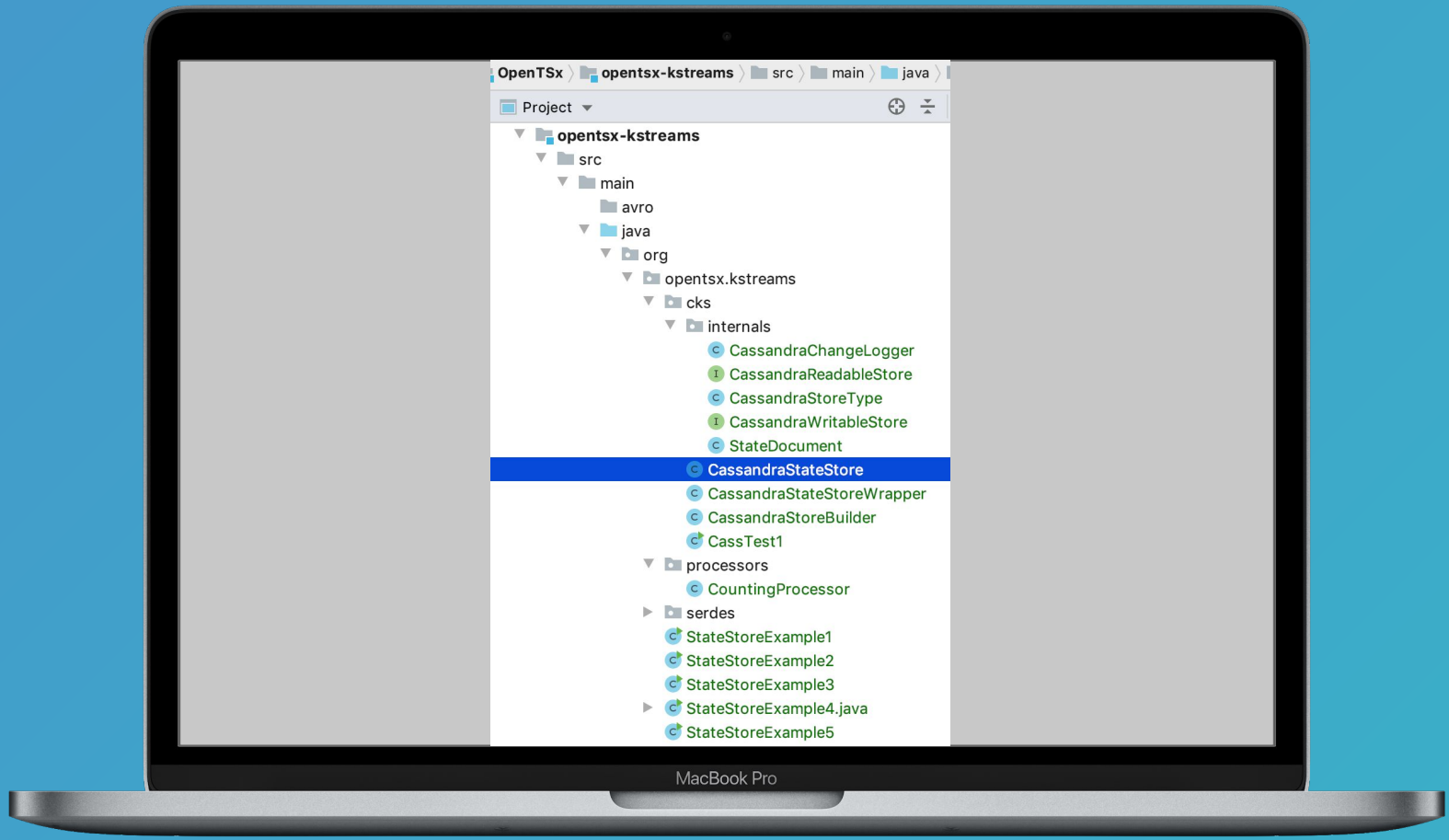
The StateStore exposes internal information & supports fault tolerance

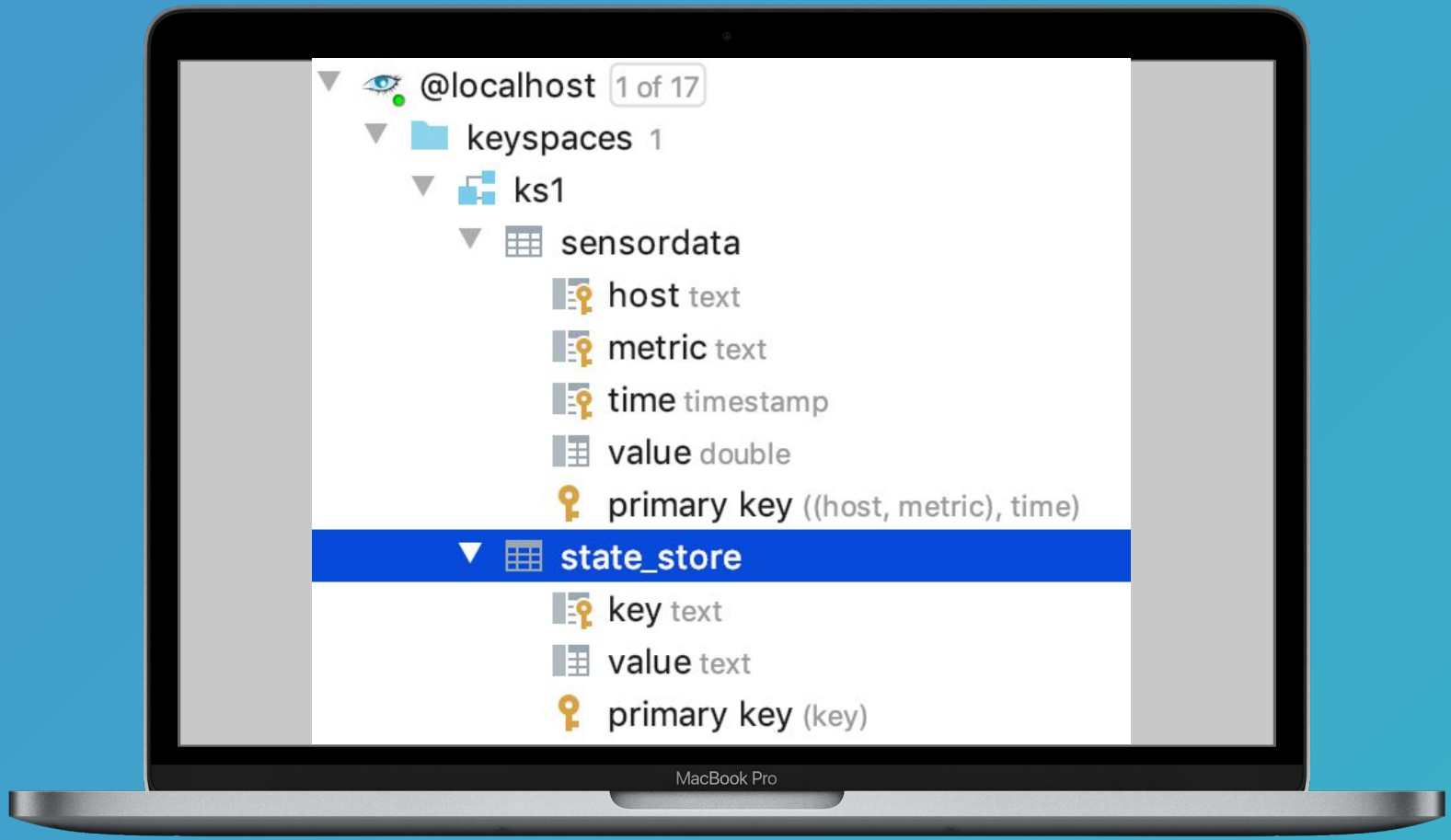


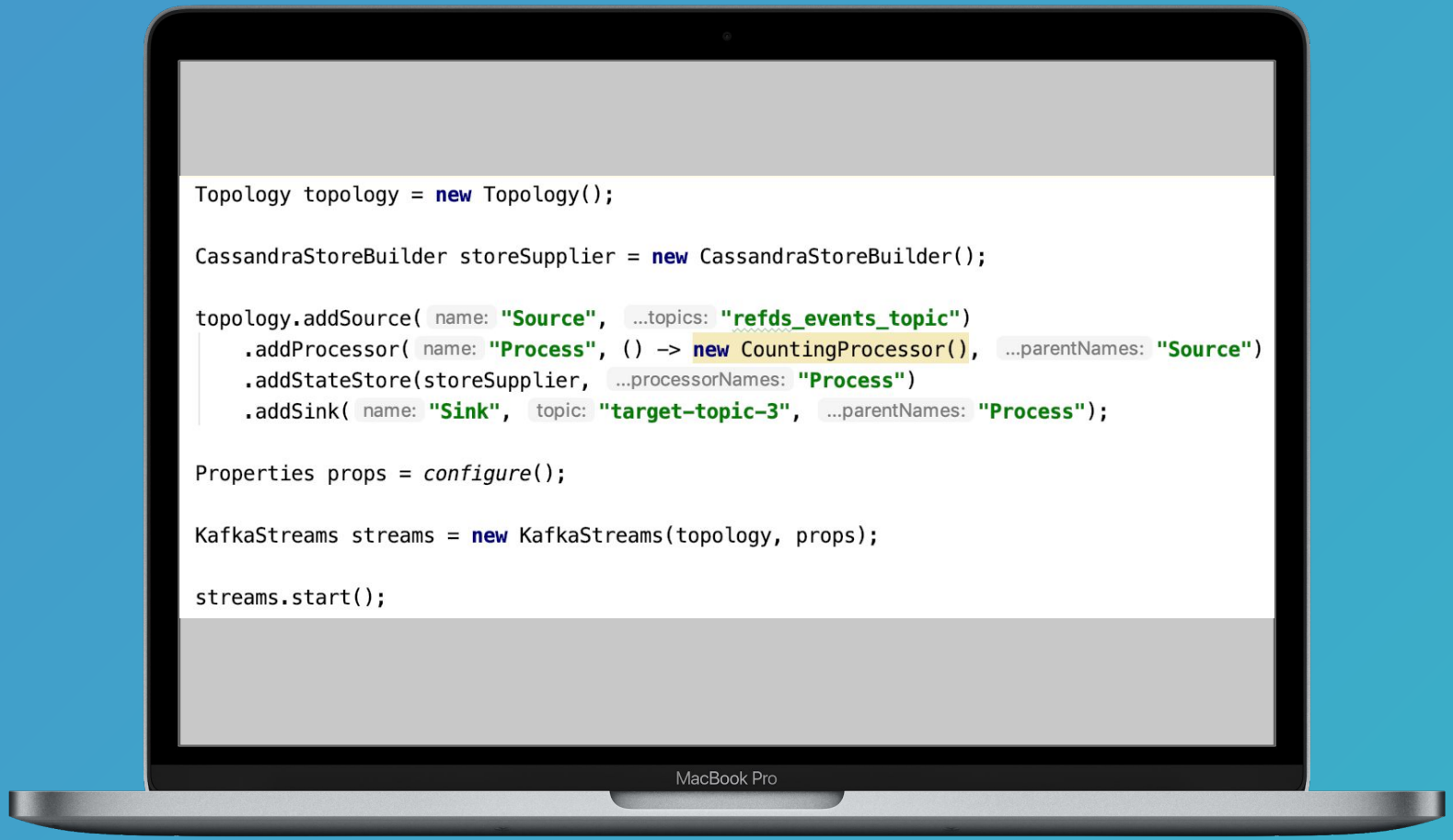
StateStore

Internal state information can be exposed via API and to Cassandra.

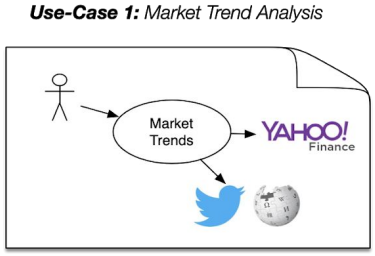
This allows usage in other streams (as derived information).


OpenTSx
opentsx-kstreams
src
main
java
Project
opentsx-kstreams
src
main
avro
java
org
opentsx.kstreams
cks
internals
CassandraChangeLogger
CassandraReadableStore
CassandraStoreType
CassandraWritableStore
StateDocument
CassandraStateStore
CassandraStateStoreWrapper
CassandraStoreBuilder
CassTest1
processors
CountingProcessor
serdes
StateStoreExample1
StateStoreExample2
StateStoreExample3
StateStoreExample4.java
StateStoreExample5
MacBook Pro


@localhost 1 of 17
keyspaces 1
ks1
sensordata
host text
metric text
time timestamp
value double
primary key ((host, metric), time)
state_store
key text
value text
primary key (key)
MacBook Pro

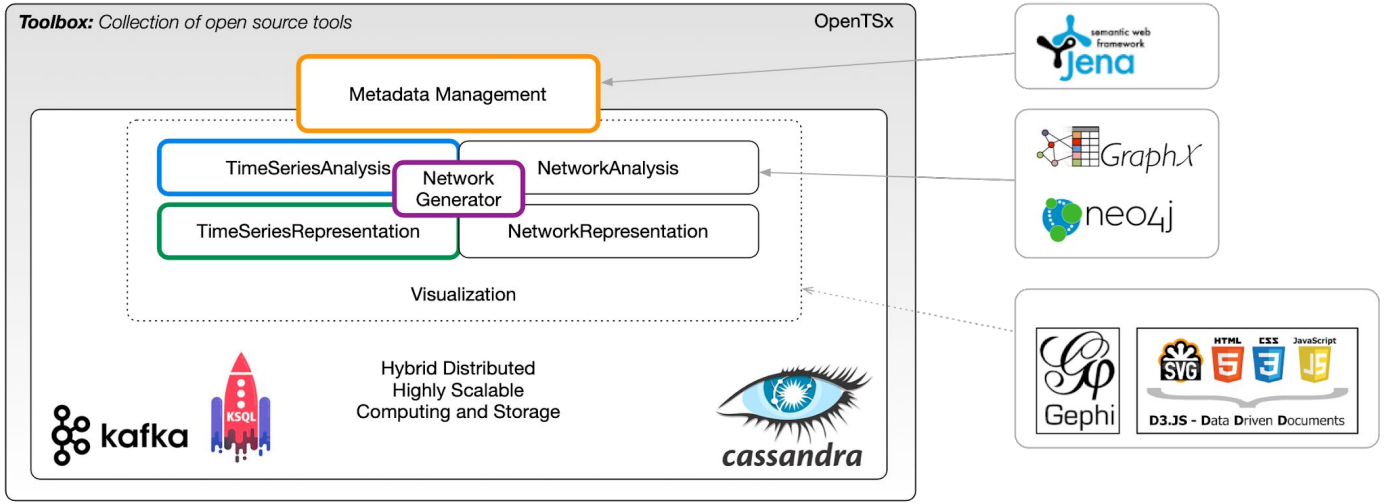

Topology topology = new Topology();
CassandraStoreBuilder storeSupplier = new CassandraStoreBuilder();
topology.addSource(name: "Source", ...topics: "refds_events_topic")
.addProcessor(name: "Process", () -> new CountingProcessor(), ...parentNames: "Source")
.addStateStore(storeSupplier, ...processorNames: "Process")
.addSink(name: "Sink", topic: "target-topic-3", ...parentNames: "Process");
Properties props = configure();
KafkaStreams streams = new KafkaStreams(topology, props);
streams.start();
MacBook Pro

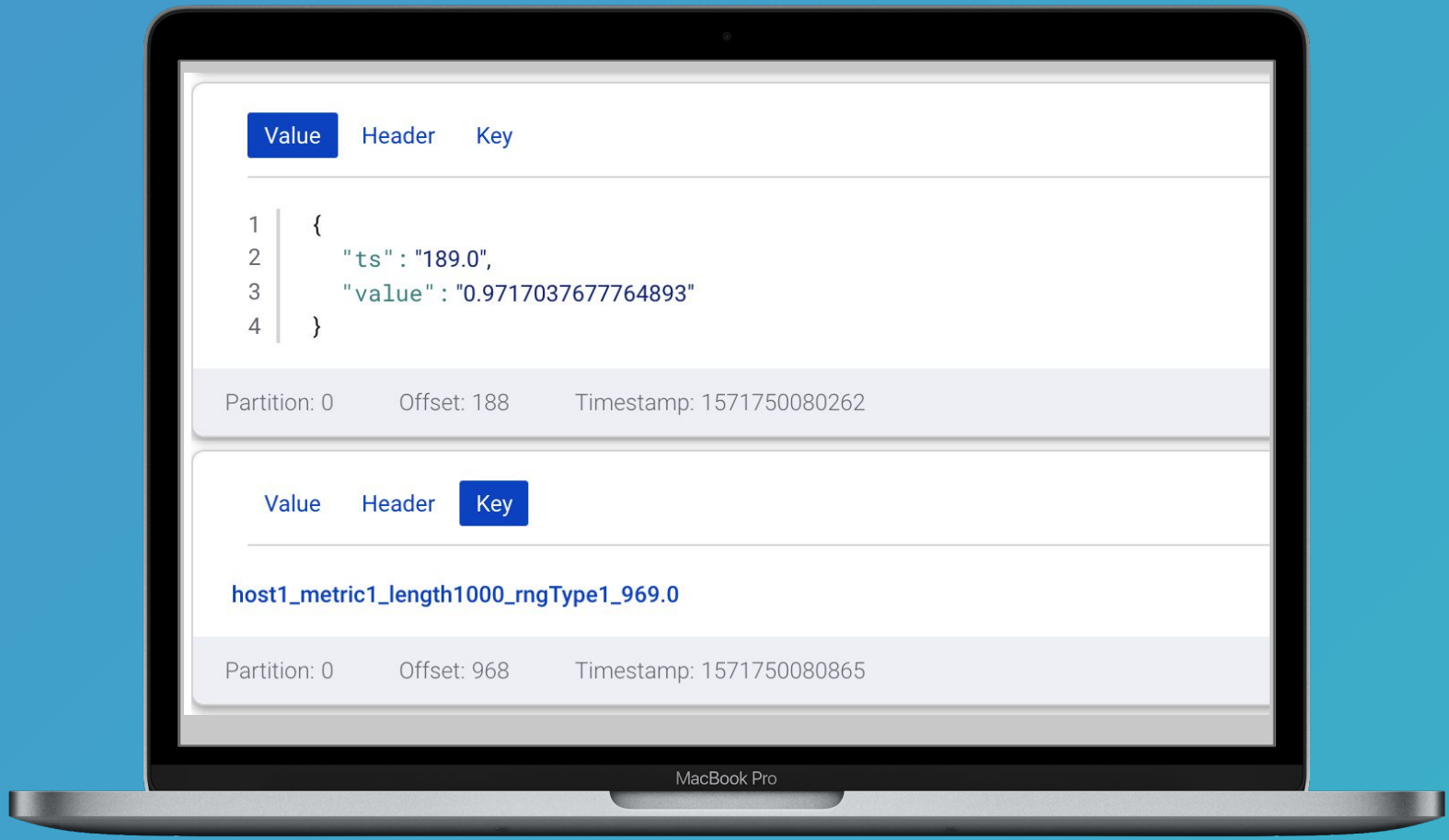
Topics can keep more than just events:

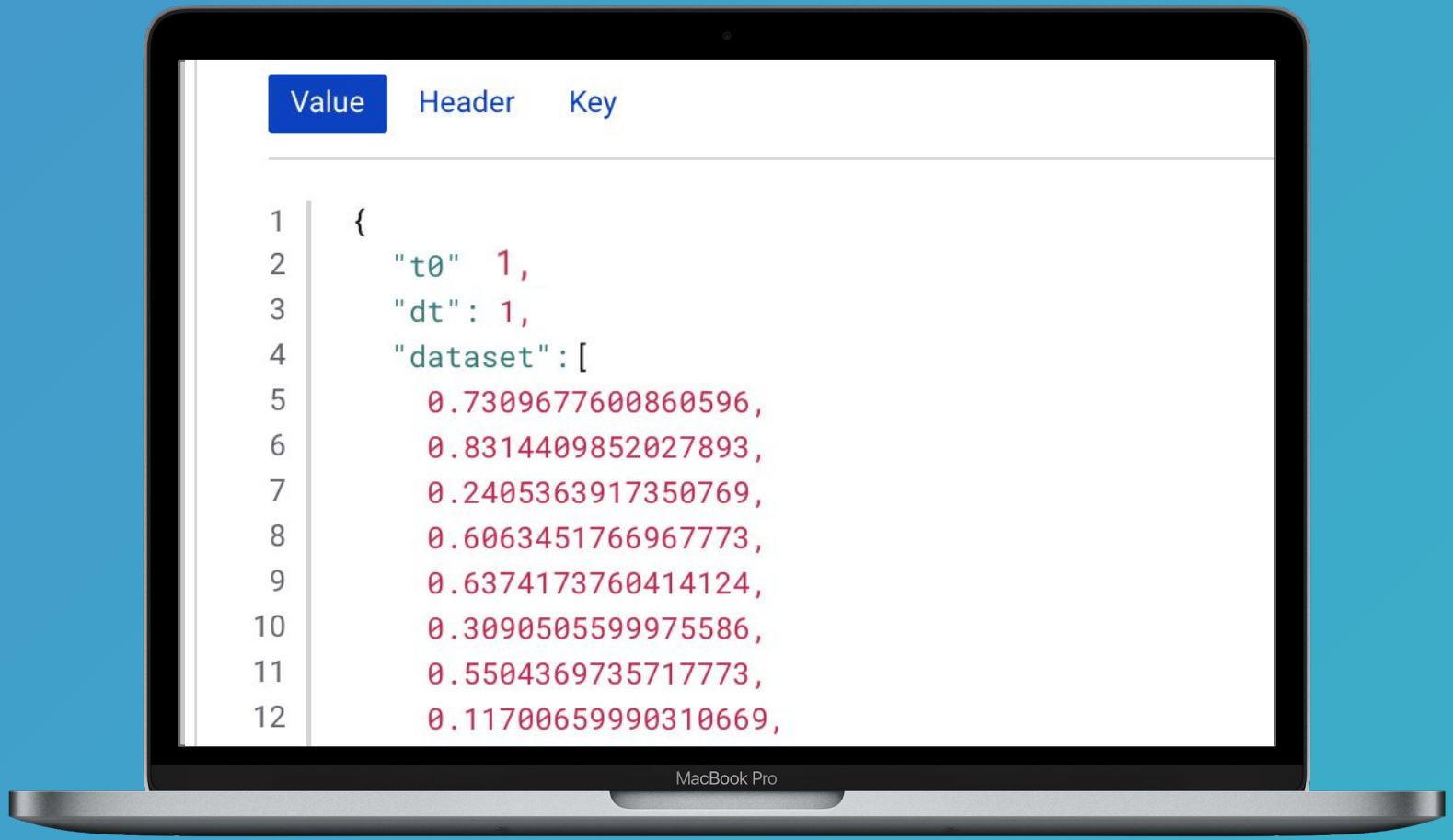


Time Series & Graph Store

Special data models are needed for some special algorithms. e.g.:
- **node lists**
- **edge lists**
- aggregated **episodes**
- normalized episodes



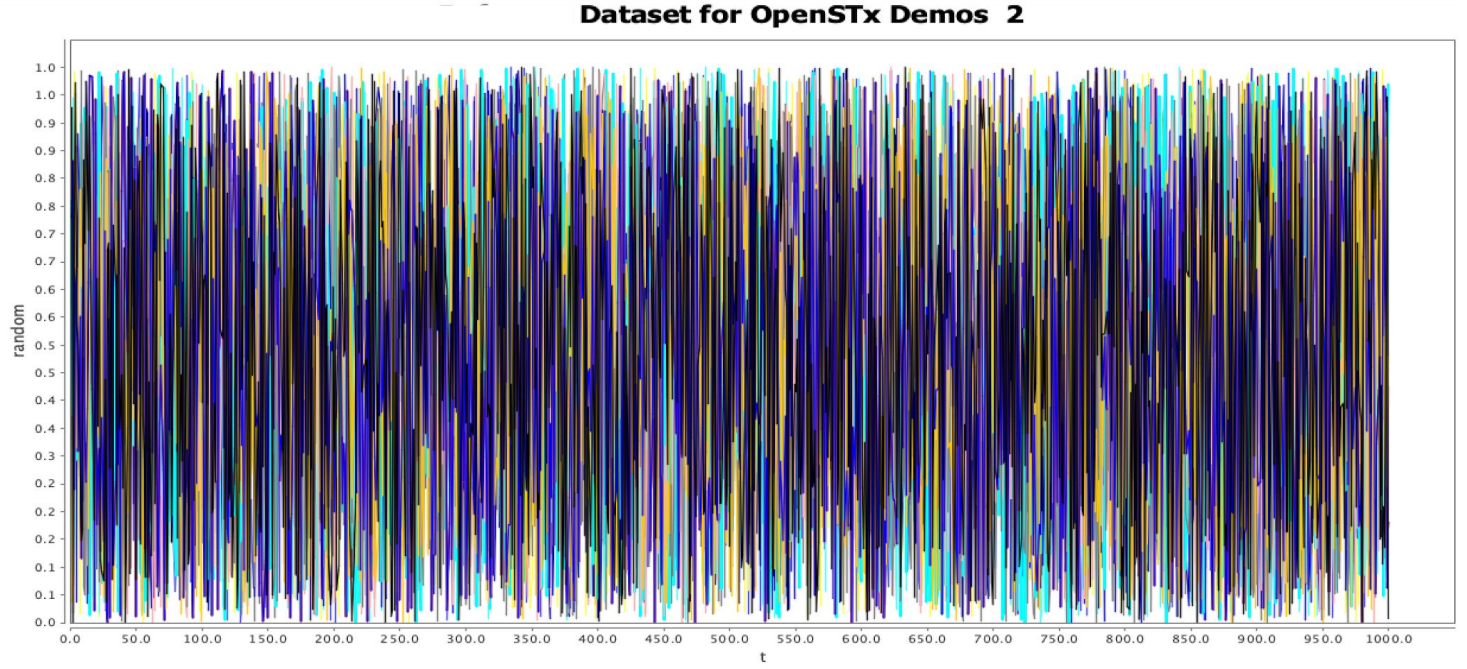

Value
Header
Key
1 {
2 "ts" : "189.0",
3 "value" : "0.9717037677764893"
4 }
Partition: 0
Offset: 188
Timestamp: 1571750080262
Value
Header
Key
host1_metric1_length1000_rngType1_969.0
Partition: 0
Offset: 968
Timestamp: 1571750080865
MacBook Pro


Value
Header
Key
{
"t0" 1,
"dt": 1,
"dataset":[
0.7309677600860596,
0.8314409852027893,
0.2405363917350769,
0.6063451766967773,
0.6374173760414124,
0.3090505599975586,
0.5504369735717773,
0.11700659990310669,
MacBook Pro

TSB | Pix-Node | Transformations | Uni-variate. Analysis | Multi-variate Analysis | Group Operations | TS-Model

Univariate Analysis

-10 | 10 | 50 | Histogram per series | Histogram cross-cut | TEST Normal Distribution | CDF | ENTROPY | Auto-Correlation | DFA | MF-DFA



(---){refds_small_bucket_topic_A}
[host1_metric1_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric1_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric2_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric2_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric3_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric3_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric4_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric4_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric5_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric5_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric6_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric6_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric7_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric7_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric8_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric8_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric9_length1000_rngType1
(---){refds_small_bucket_topic_A}
[host1_metric9_length1000_rngType2
(---){refds_small_bucket_topic_A}
[host1_metric10_length1000_rngType
(---){refds_small_bucket_topic_A}
[host1_metric10_length1000_rngType

Close

Integration of Kafka and Cassandra:

We use the Kafka-source- & -sink-connectors to manage data flows between Kafka and Cassandra.

We use Cassandra as a state store to expose process internal data and derived data without the need of additional processing.

Cassandra provides special access patterns to the data, access to historical data, and long term storage.

Summary:

Because Kafka is a scalable & extensible platform it is good for complex event processing in any industry, on premise, and in the cloud.

The Kafka ecosystem provides extension points for any kind of domain specific functionality - from advanced analytics to real time data enrichment.

Complex solutions can be composed from a few fundamental building blocks.

confluent

MANY THANKS TO MY TEAM
@Confluent:

Professional Services
Community Services

cnfl.io/blog

THANK YOU !

mirko@confluent.io

cnfl.io/meetups

cnfl.io/blog

cnfl.io/slack