



Unleashing Apache Kafka and TensorFlow in Hybrid Architectures

Kai Waehner

Technology Evangelist
kontakt@kai-waehner.de
LinkedIn
[@KaiWaehner](https://www.linkedin.com/in/kaiwaehner)
www.confluent.io
www.kai-waehner.de



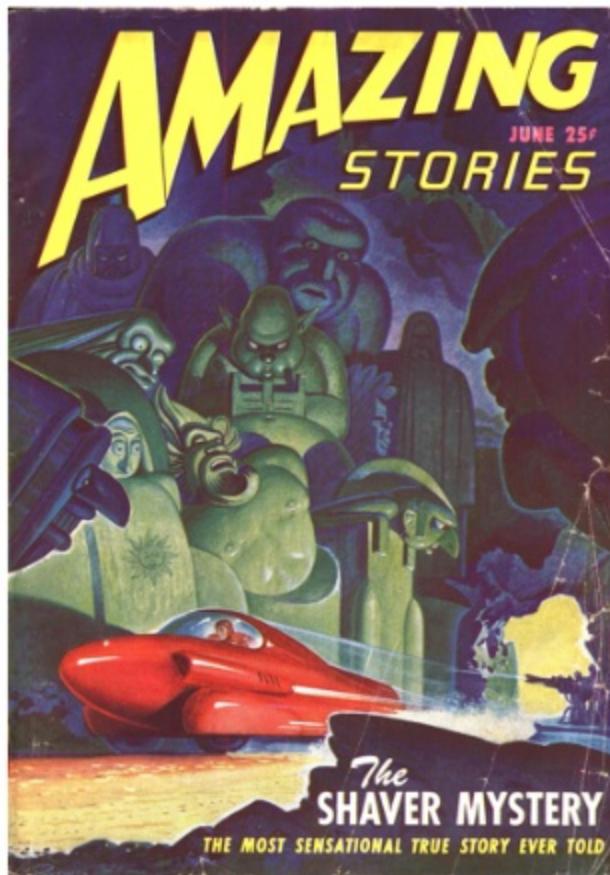
Poll

In which of these environments, if at all, are you using Apache Kafka today?

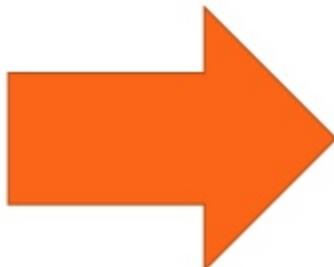
1. Self managed - On premise (Bare metal, VM)
2. Self managed - Public cloud (AWS EC2, etc.)
3. Self managed - Virtual private cloud (Kubernetes, Mesos, etc.)
4. Kafka as a Service – Public cloud
5. We do not currently use Apache Kafka



Disclaimer: This is a fictional story (but not far from reality)...

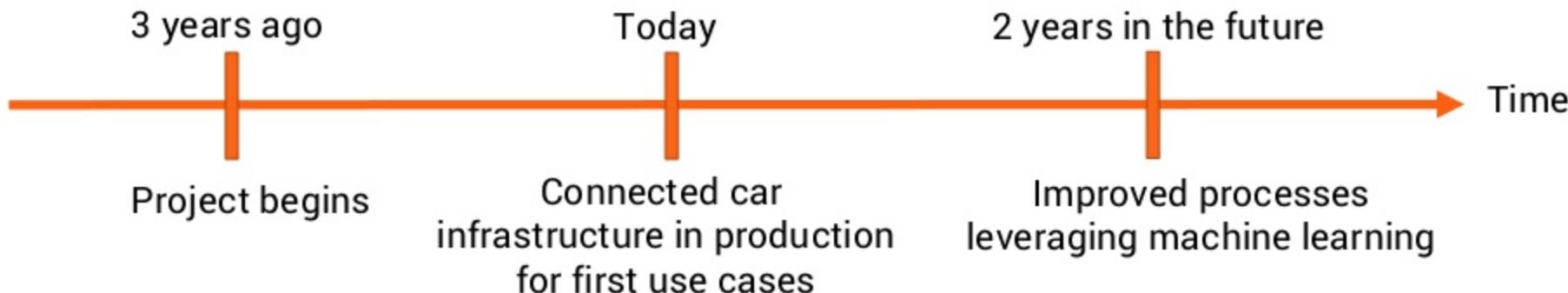


Global automotive company builds connected car infrastructure

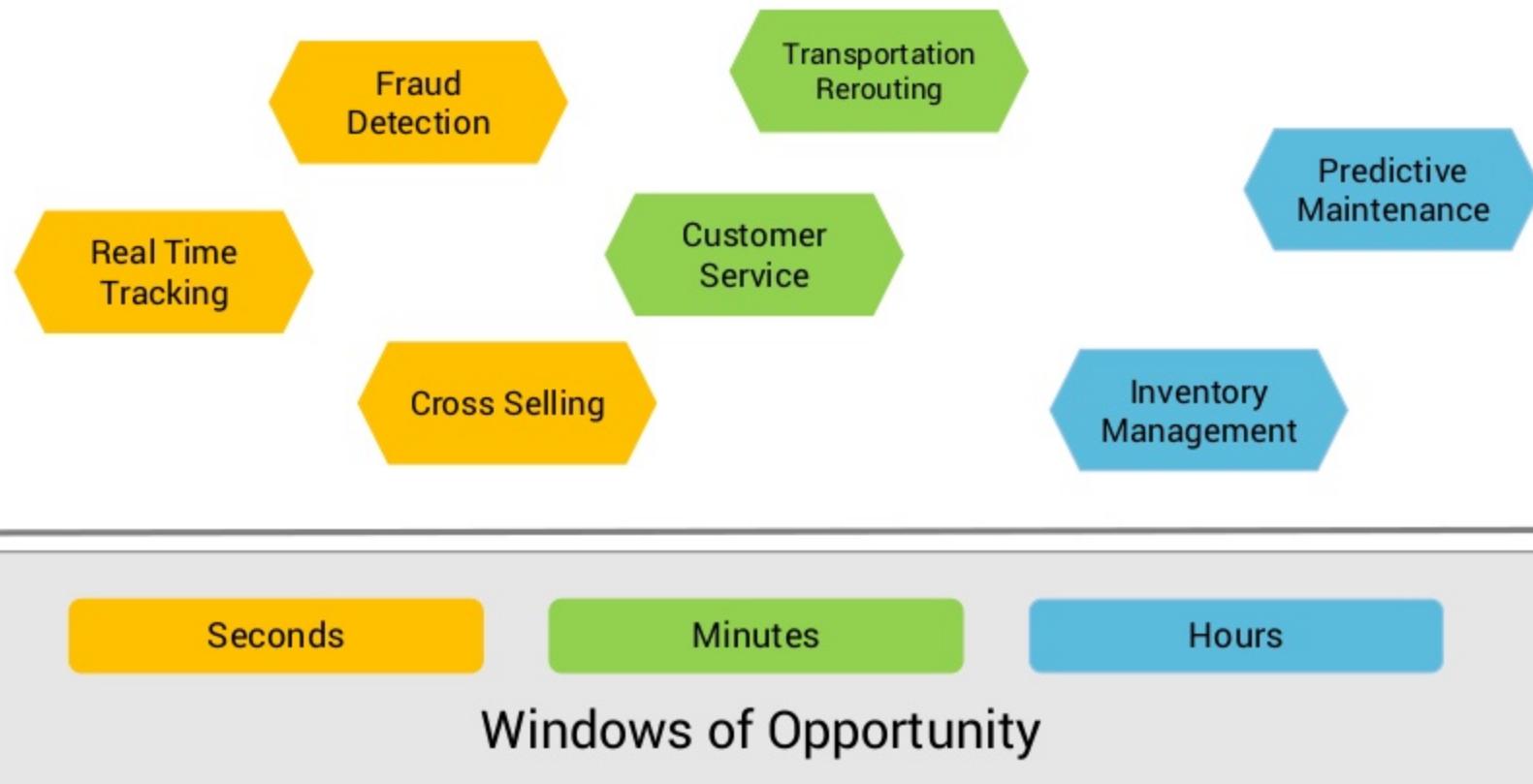


Digital Transformation

- Improve customer experience
- Increase revenue
- Reduce risk

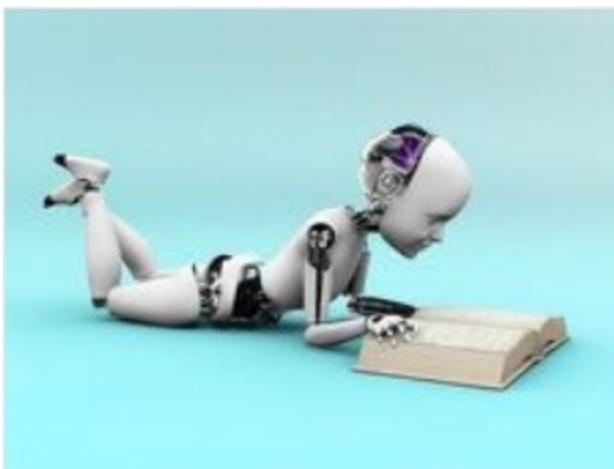


Analyze and act on critical business moments



Machine Learning (ML)

...allows computers **to find hidden insights without being explicitly programmed** where to look.



Machine Learning

- Decision Trees
- Naïve Bayes
- Clustering
- Neural Networks
- Etc.

Deep Learning

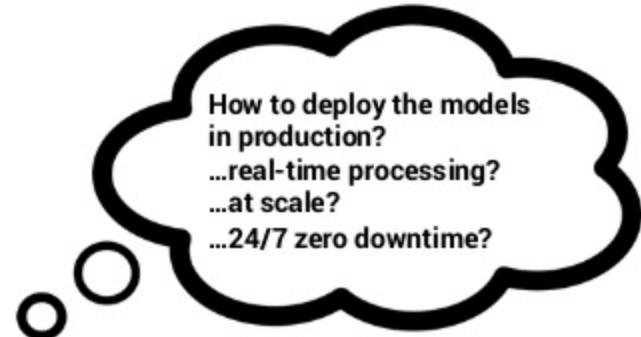
- CNN
- RNN
- Autoencoder
- Etc.

The First Analytic Models

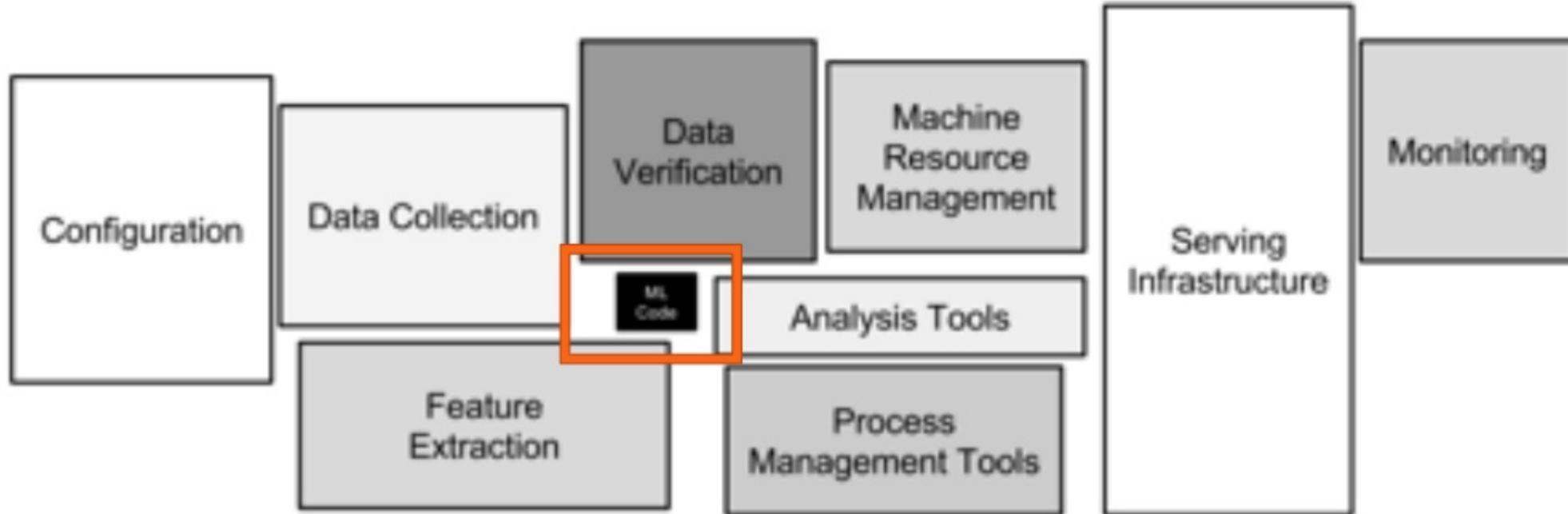
 python

 TensorFlow

```
def nQueens(queens):  
    for i in range(BOARD_SIZE):  
        test_queens = queens + [i]  
        try:  
            validate(test_queens)  
            if len(test_queens) == BOARD_SIZE:  
                return test_queens  
            else:  
                return add_queen(test_queens)  
        except Bailout:  
            pass
```

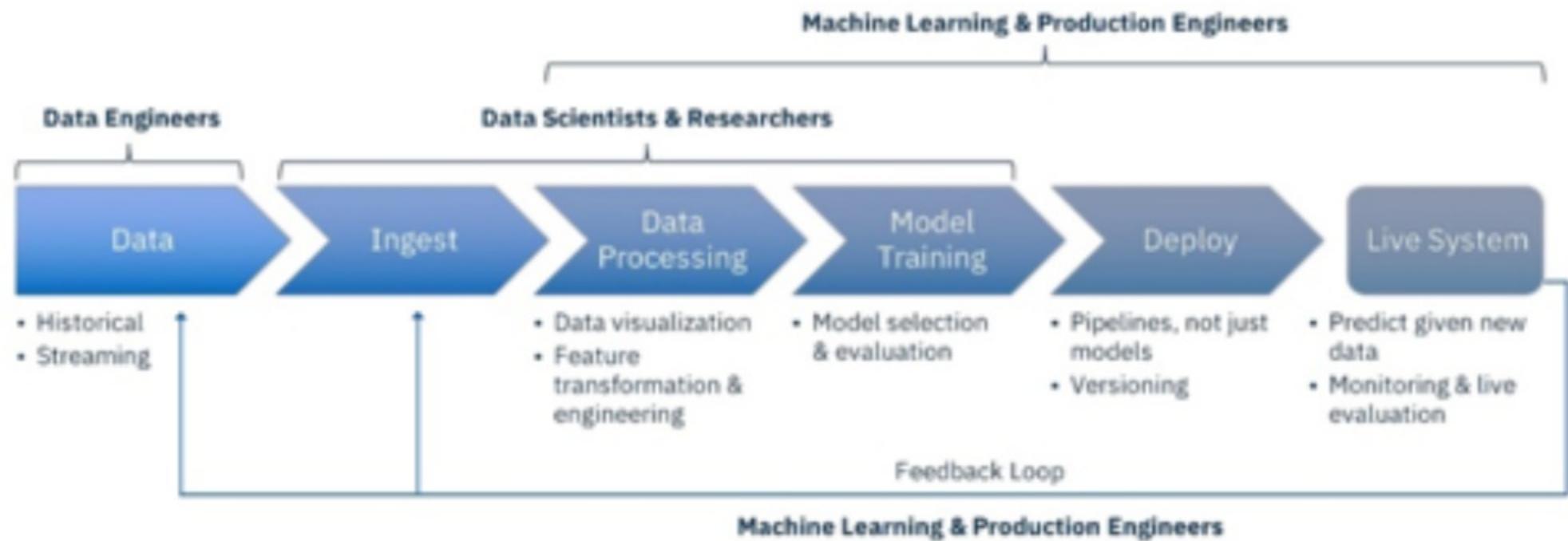


Hidden Technical Debt in Machine Learning Systems



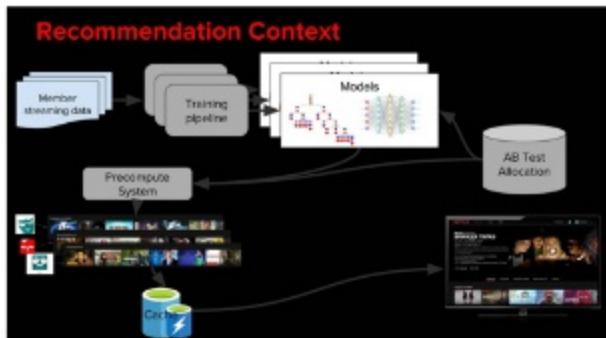
<https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.pdf>

Impedance mismatch between model development and model deployment



<https://www.slideshare.net/NickPentreath/productionizing-spark-ml-pipelines-with-the-portable-format-for-analytics-100788521>

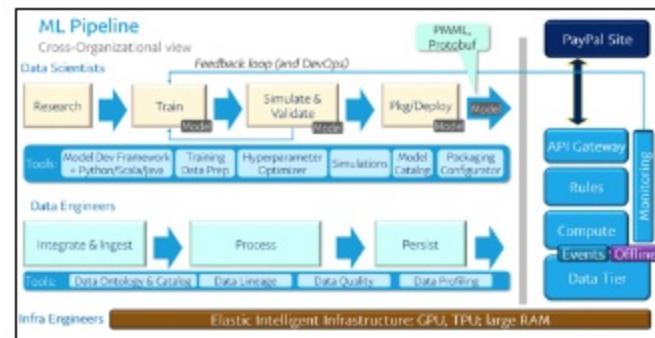
Scalable, Technology-Agnostic ML Infrastructures



Meet Michelangelo: Uber's Machine Learning Platform

By Jeremy Hermann & Mike Del Balso
October 5, 2017

The interface shows various data visualizations including line graphs and bar charts, likely representing model performance metrics over time or different categories.



NETFLIX

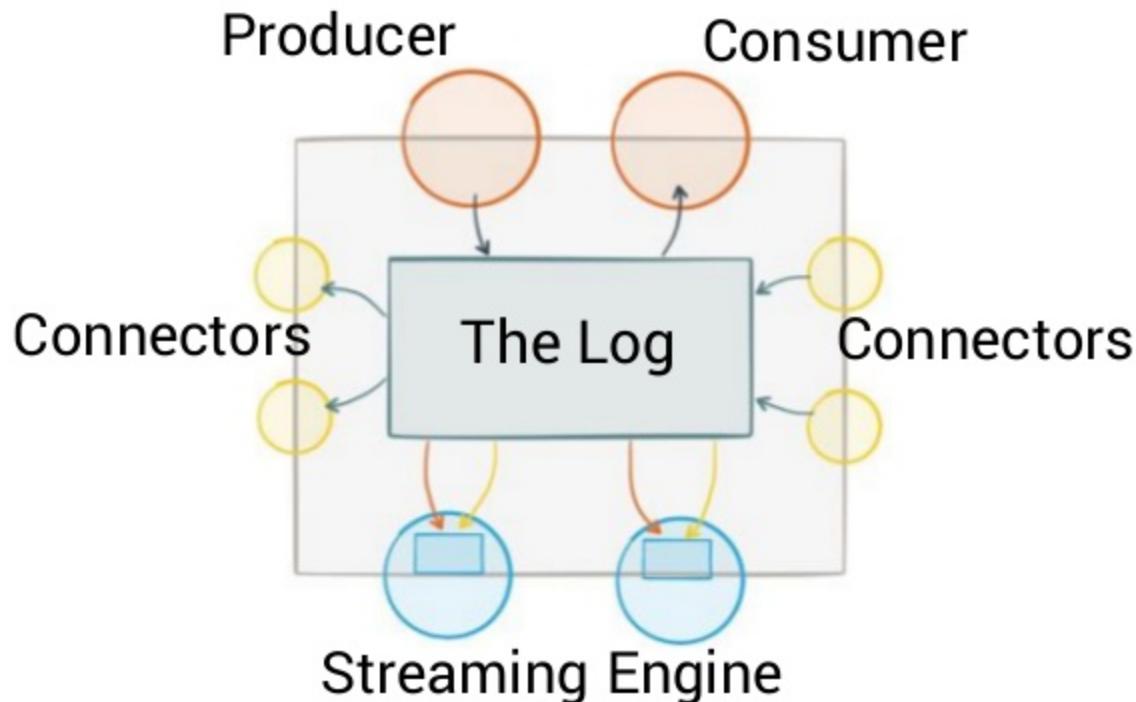
UBER

PayPal

<https://www.infoq.com/presentations/netflix-ml-meson>
<https://eng.uber.com/michelangelo>
<https://www.infoq.com/presentations/paypal-data-service-fraud>

What is this kafka thing used everywhere?

Apache Kafka—The Rise of a Streaming Platform



* Kafka Is not just used by tech giants

** Kafka is not just used for big data



> 4.5 trillion messages / day

NETFLIX

> 6 Petabytes / day

ebay

P PayPal

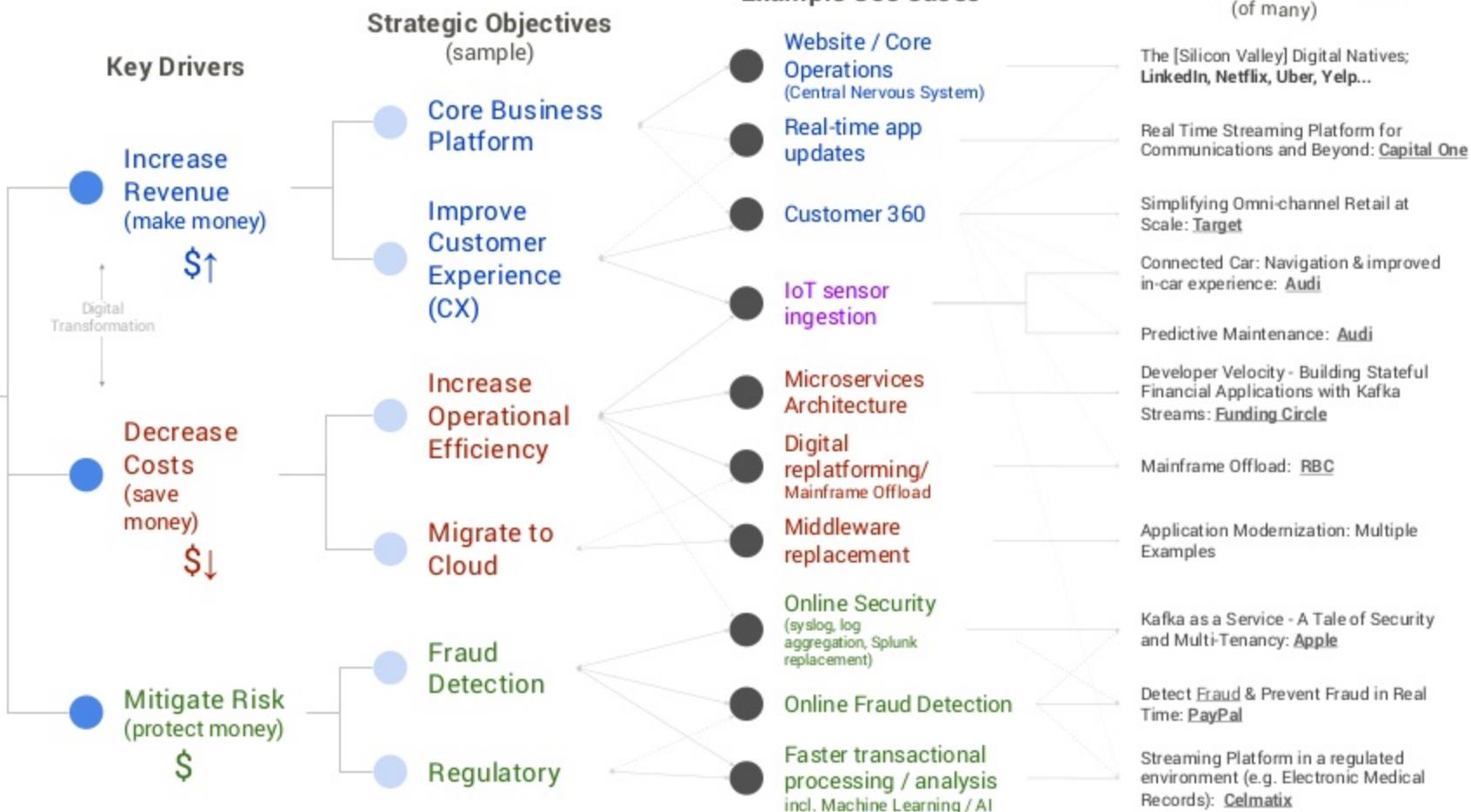


Pinterest "You name it"

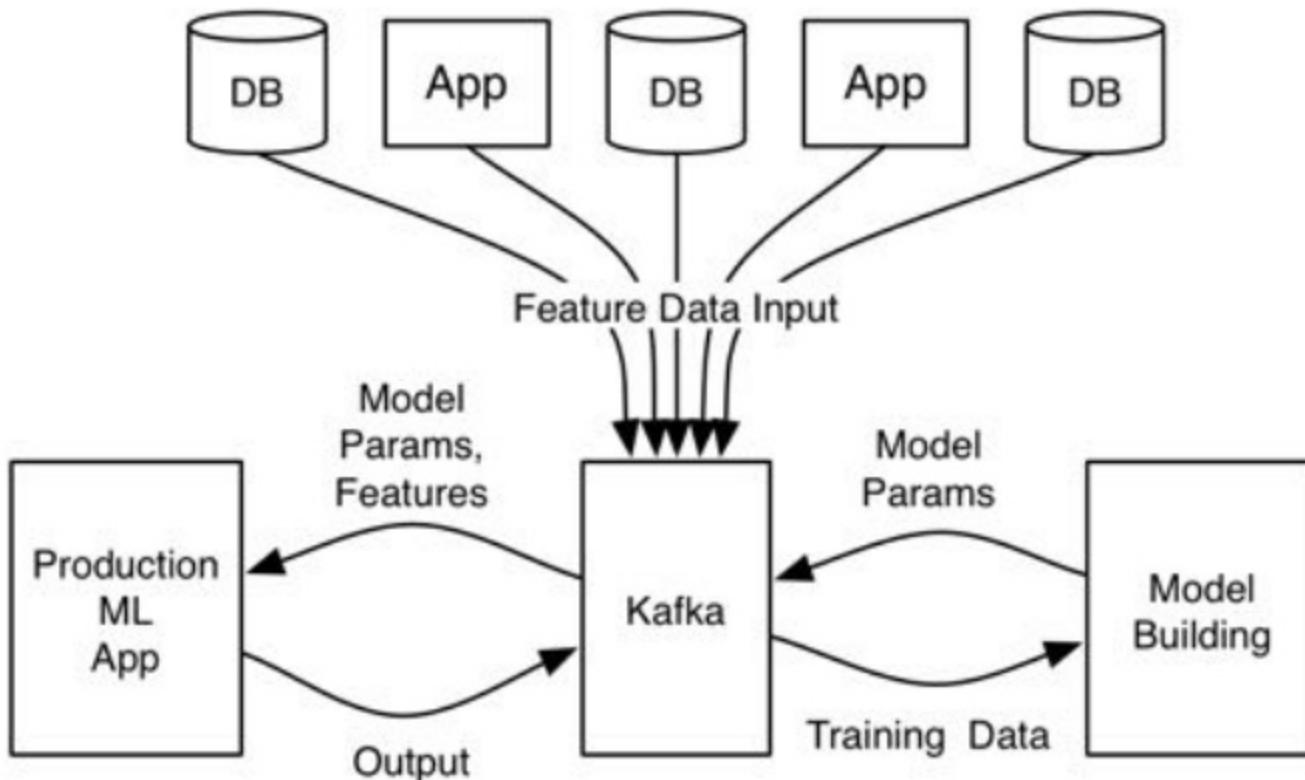
Confluents - Business Value per Use Case



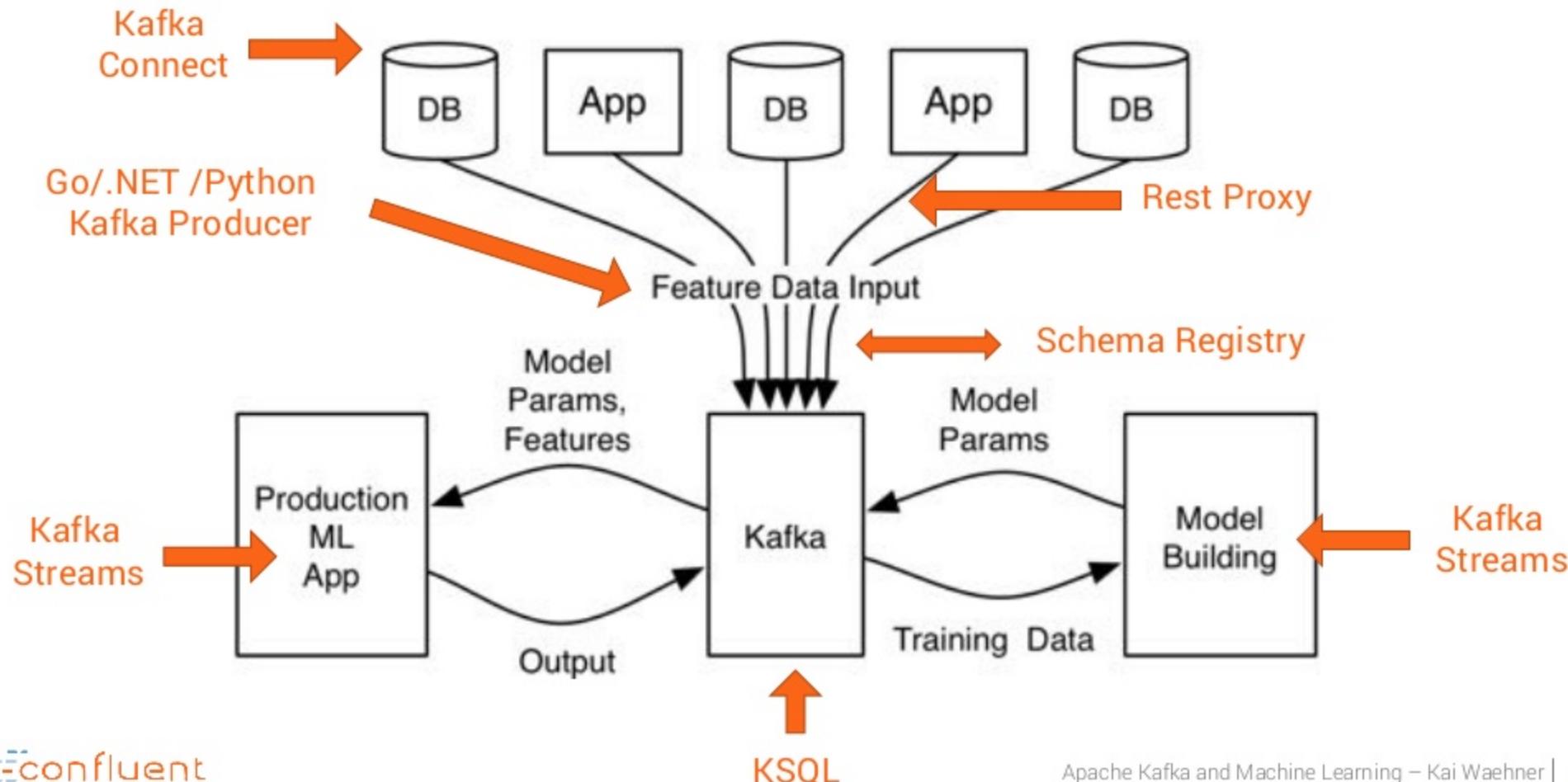
Business Value



Apache Kafka's Open Source Ecosystem as Infrastructure for ML



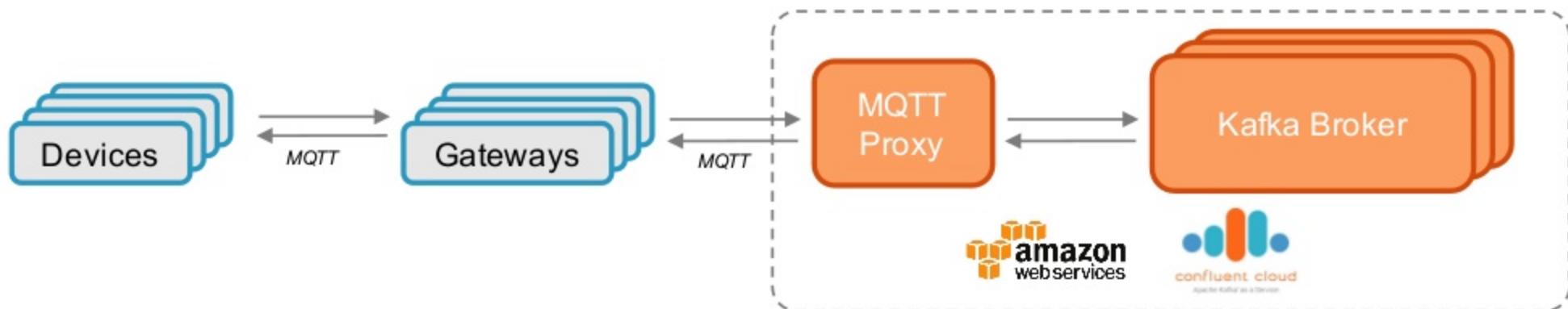
Apache Kafka's **Open Ecosystem** as Infrastructure for ML



Getting Started



Connected Car Infrastructure in Production on AWS



Real time tracking of the cars
to enable new, innovative digital services

Why did they choose Confluent Cloud?



Why Cloud?

- Extreme Scale
- Dynamic Instances
- Special Hardware (GPUs, TPUs,)

Why Confluent Cloud?

- No operations burden
- 99.95 Enterprise SLA, guaranteed high throughput, low ms latency end-to-end
- Confluent Ecosystem, Multi-Cloud + on premise Deployments,
- End-to-End monitoring with Confluent Control Center



Replication of IoT Data from AWS to GCP



Poll

Which of the following use cases would you expect for hybrid Kafka deployments and replication of data?

1. Backup / Separation of Concerns
2. Disaster recovery
3. Active / active deployments
4. Cloud migration
5. We don't anticipate using a hybrid deployment



Data Preprocessing



Google Cloud Platform



- Use KSQL to preprocess data at scale without coding
- Use SQL statements for interactive analysis
+ deployment to production at scale
- Leverage e.g. Python with KSQL REST interface

Data Ready
for
Model Training

Data needs to be
preprocessed at
scale and reusable!

Preprocessing with KSQL

```
SELECT car_id, event_id, car_model_id, sensor_input  
FROM car_sensor c  
LEFT JOIN car_models m ON c.car_model_id =  
m.car_model_id  
WHERE m.car_model_type = 'Audi_A8';
```



The screenshot shows a Jupyter Notebook interface with the following code cells:

```
In [1]: from ksql import KSQLAPI  
In [2]: client = KSQLAPI('http://localhost:8088')  
In [3]: from ksql import KSQLAPI  
client = KSQLAPI('http://localhost:8088')  
query = client.query('SELECT payload, sensor FROM pagesviews_original LIMIT 100')  
In [4]: for item in query: print(item)
```

The output of the fourth cell shows 100 rows of data, each containing a 'payload' and a 'sensor' field.



Data Ingestion into a Data Store

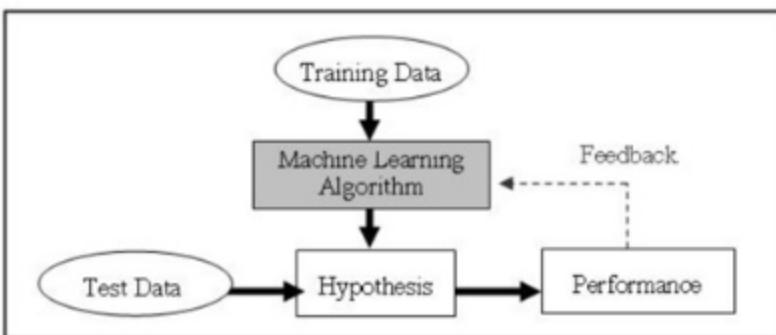
Preprocessed
Data



- "Kafka Benefits Under the Hood"
- Out-of-the-box connectivity
- Data format conversion
- Single message transformation (including error-handling)



Model Training using a Data Store



Let's build some models
at **extreme scale** using
TensorFlow and TPUs!



Google Cloud Storage



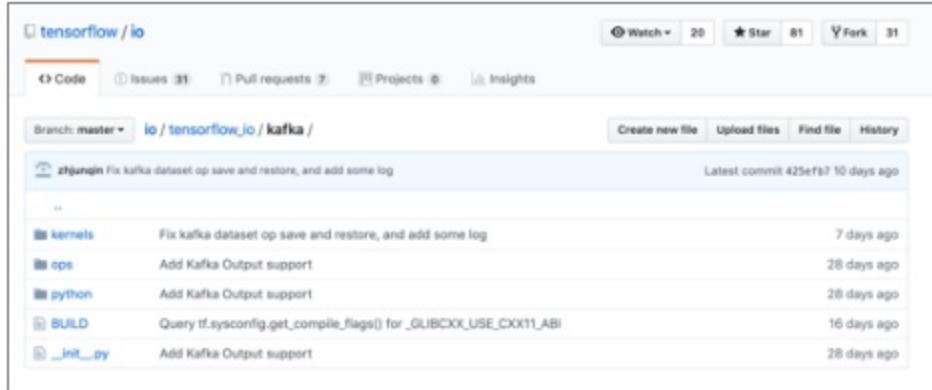
Cloud Machine
Learning Engine

 TensorFlow



Analytic Model

Model Training without additional Data Store



The screenshot shows the GitHub repository page for tensorflow/tensorflow_io. The master branch has 31 pull requests. One pull request by zhjungin is highlighted, showing commits to fix Kafka dataset op save and restore, and add some log. Other commits include fixing kernels, adding Kafka output support for ops, python, and BUILD files, and querying tf.sysconfig.get_compile_flags() for _GLIBCXX_USE_CXX11_ABI.

Commit Message	Time Ago
Fix kafka dataset op save and restore, and add some log	7 days ago
Add Kafka Output support	28 days ago
Add Kafka Output support	28 days ago
Query tf.sysconfig.get_compile_flags() for _GLIBCXX_USE_CXX11_ABI	16 days ago
Add Kafka Output support	28 days ago

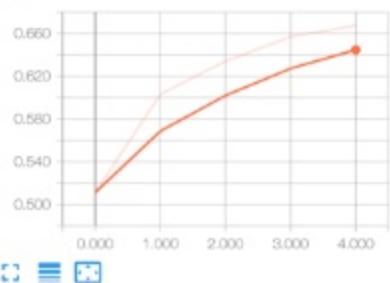


https://github.com/tensorflow/io/tree/master/tensorflow_io/kafka

- Native integration between Kafka and TensorFlow
- KafkaDataSet and KafkaOutputSequence for TensorFlow
- Written in C++ (linked with librdkafka)
- Part of the graph in TensorFlow
- Direct training and inference from streaming data
- No data storage like S3 or HDFS needed

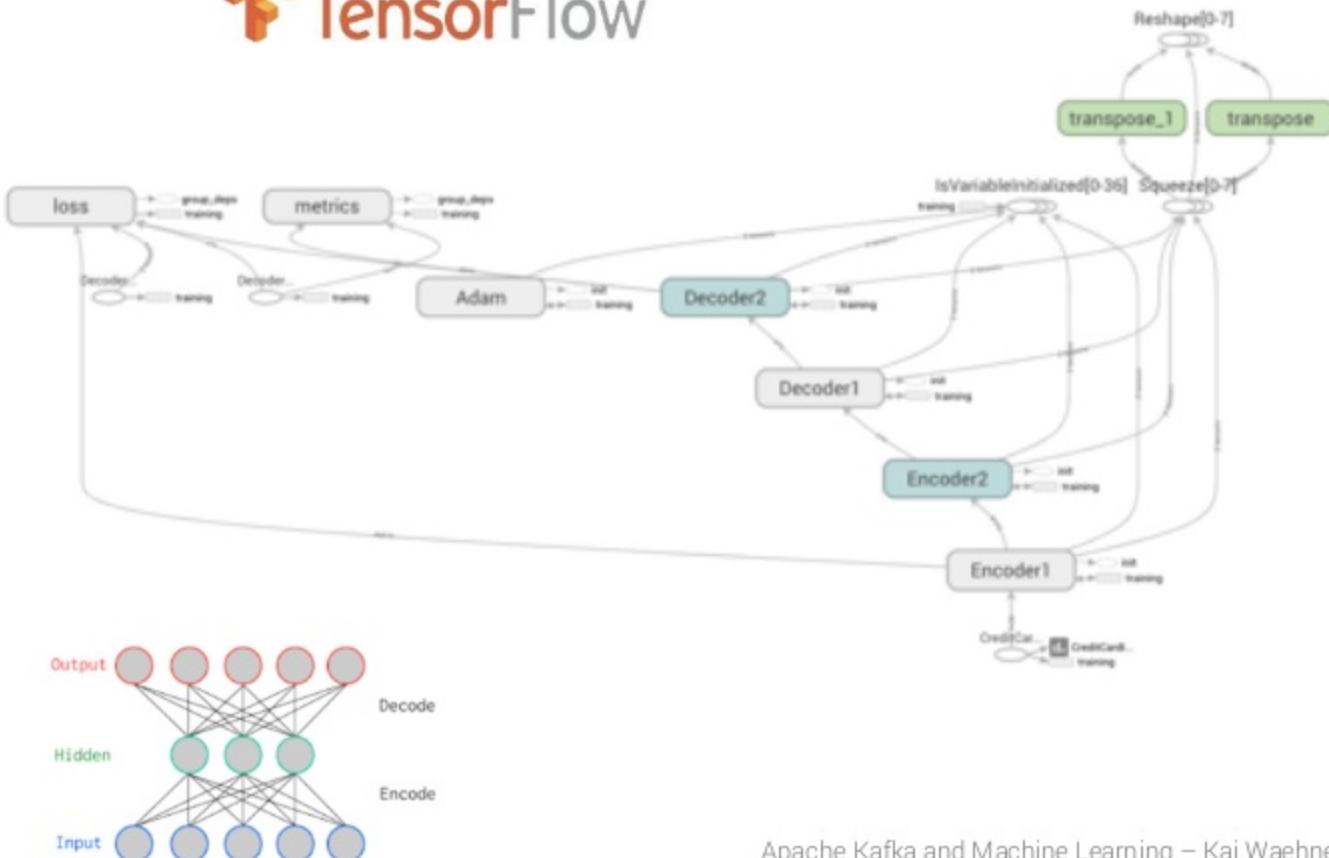
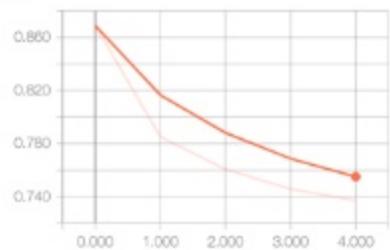
Analytic Model (Autoencoder for Anomaly Detection)

acc

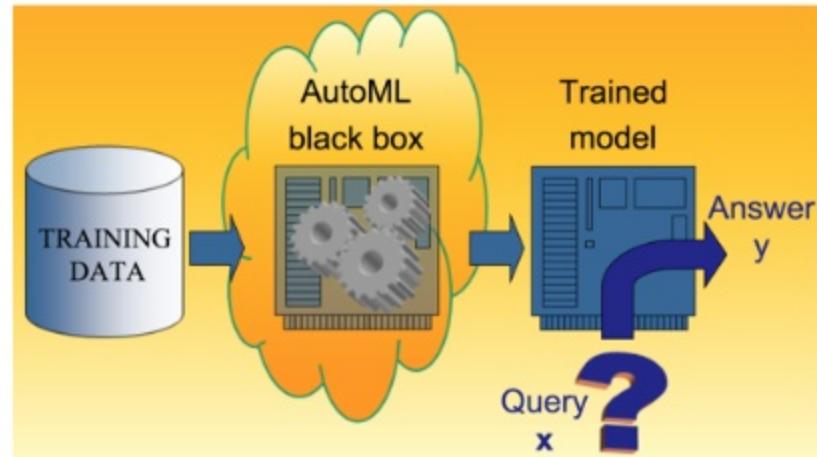


loss

loss



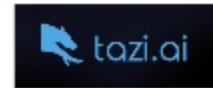
AutoML



Cloud AutoML

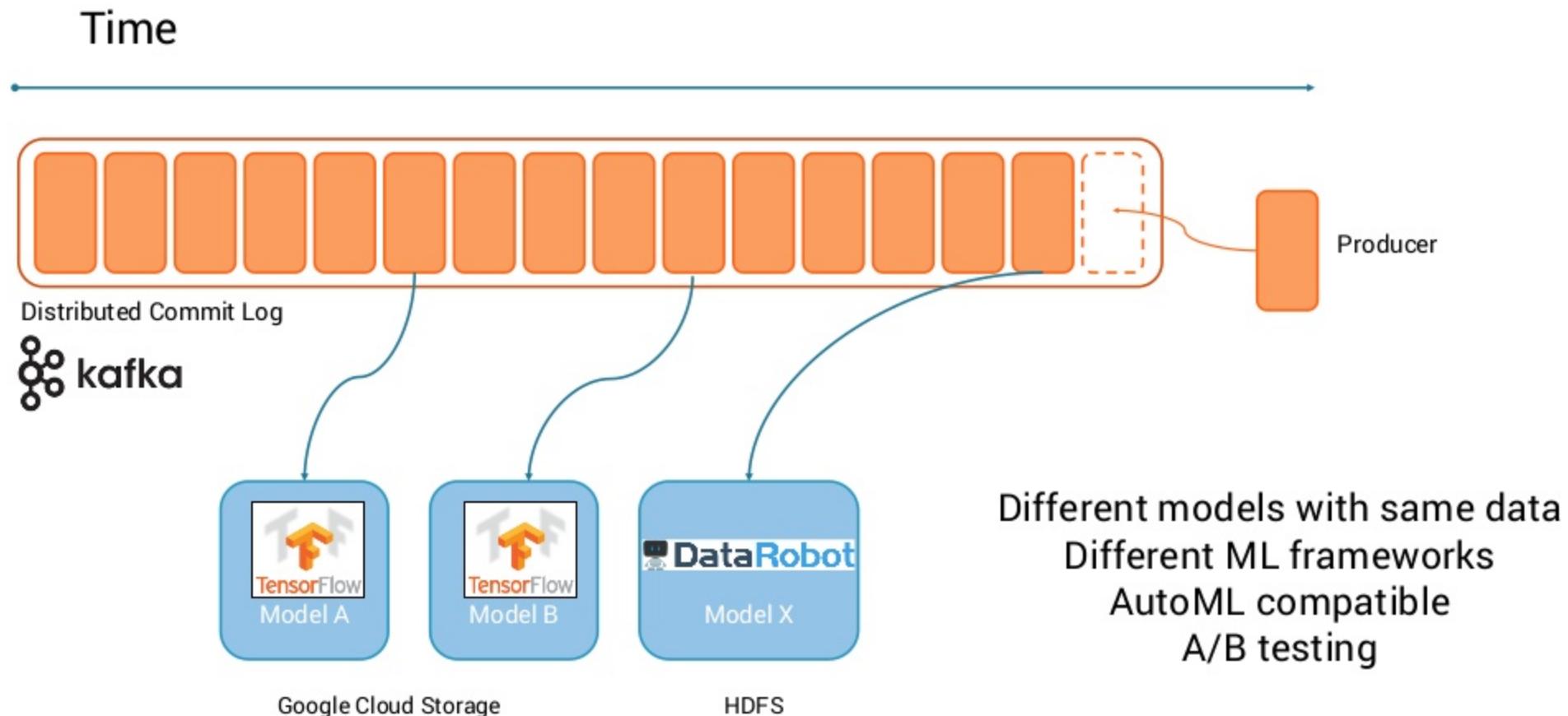


H₂O.ai
Driverless AI: Your Expert System for AI



“One-click Data-in Model-out Simplicity”

Replayability – a log never forgets!



The Need for Local Data Processing



Confluent
Replicator



TensorFlow

Analytic
Model



amazon
web services



Google Cloud Platform

CLOUD

PII data



confluent



Local Processing

ON PREMISE

We are ready to use our
models for predictions,
BUT all the PII data needs to
be processed in our local data
center!



Self managed on premise deployment for model deployment and monitoring



kubernetes



MESOSPHERE



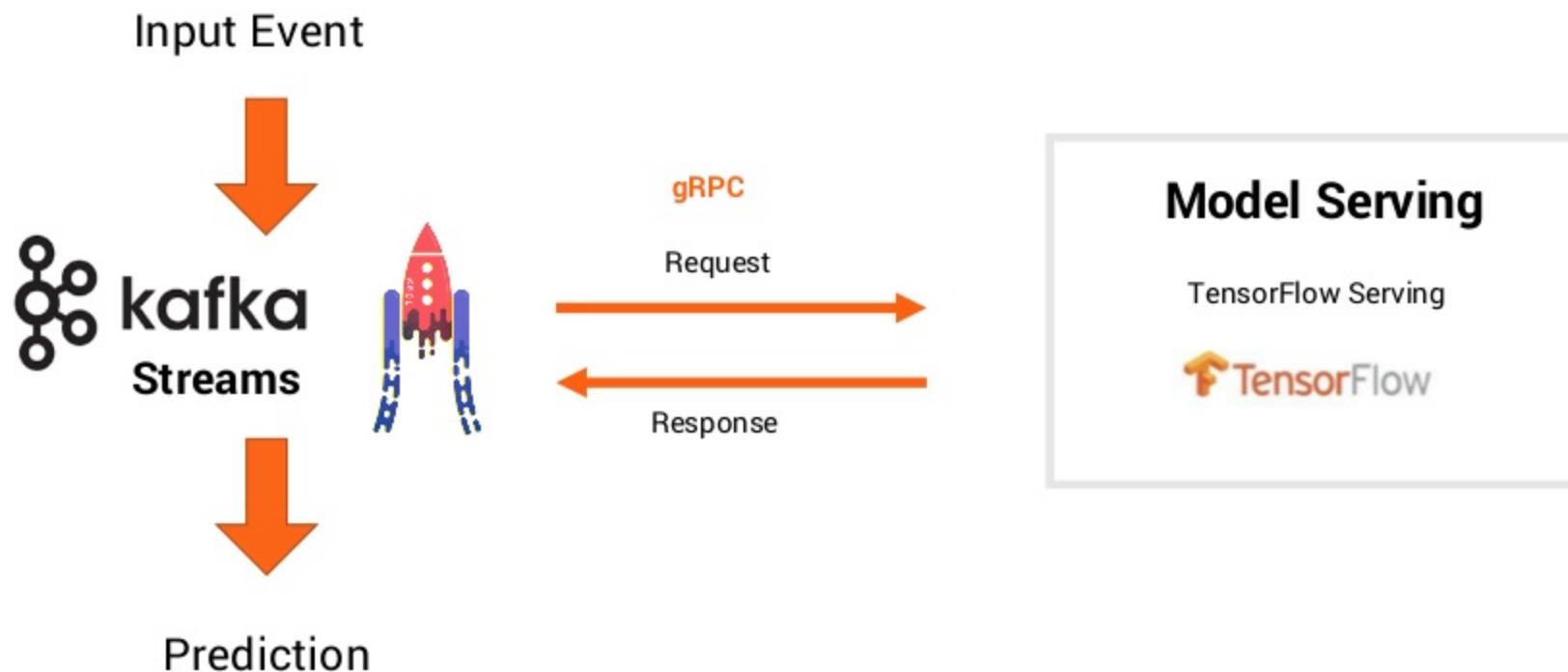
Confluent Operator takes over the challenge of operating Kafka on Kubernetes!

(Automated provisioning, scaling, fail-over, partition rebalancing, rolling updates, monitoring, ...)

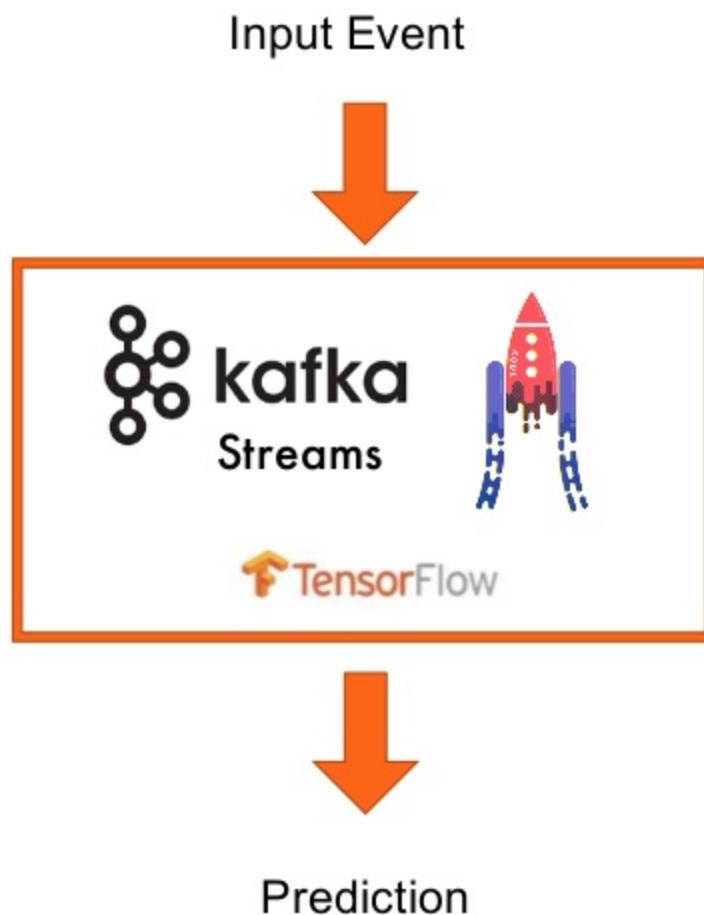
Stream Processing vs. Request-Response for Model Serving



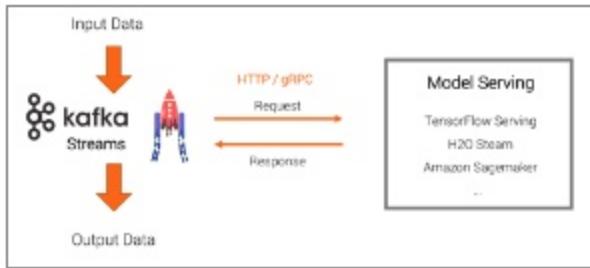
Model Deployment - Option 1: RPC communication to do model inference



Model Deployment - Option 2: Model inference natively integrated into the App



Stream Processing vs. Request-Response for Model Serving



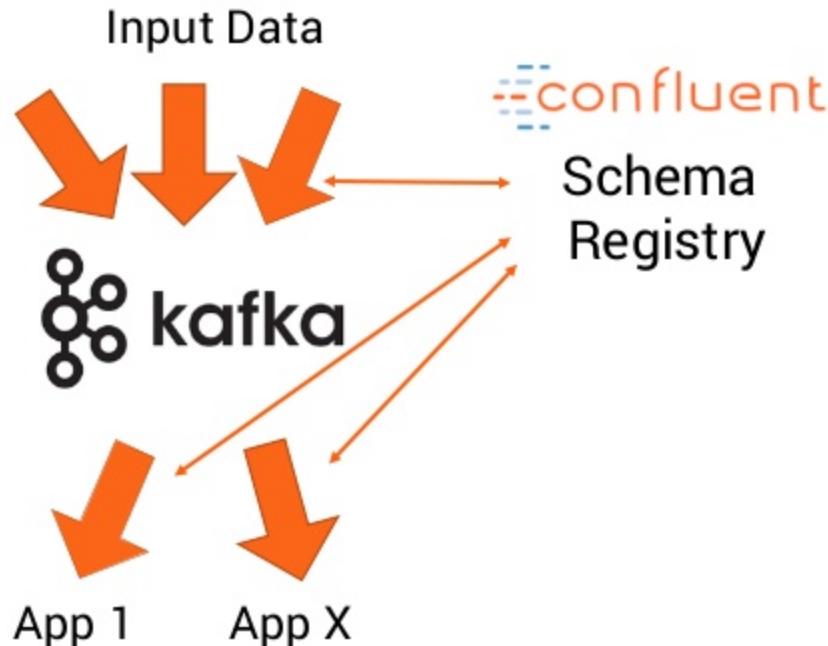
Pros of a Model Server:

- **Simple integration** with existing technologies and organizational processes
- **Easier to understand** if you come from non-streaming world
- **Later migration** to real streaming is also **possible**
- **Model management built-in** for different models, versioning and A/B testing

Cons (= Pros of Deployment in the Streaming App):

- **Worse latency** as remote call instead of local inference
- **No offline inference** (devices, edge processing, etc.)
- **Coupling** the availability, scalability, and latency/throughput of your Kafka Streams application with the **SLAs of the RPC interface**
- **Side-effects** (e.g., in case of failure) **not covered by Kafka processing** (e.g., exactly once)

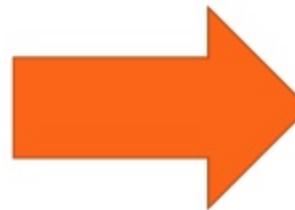
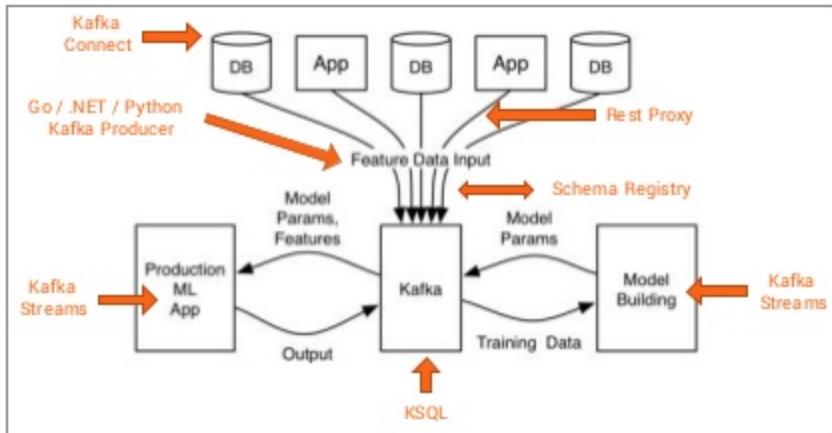
Confluent Schema Registry for Message Validation



- "Kafka Benefits Under the Hood"
- Schema definition + evolution
- Forward and backward compatibility
- Multi data center deployment



Monitoring the infrastructure for ML

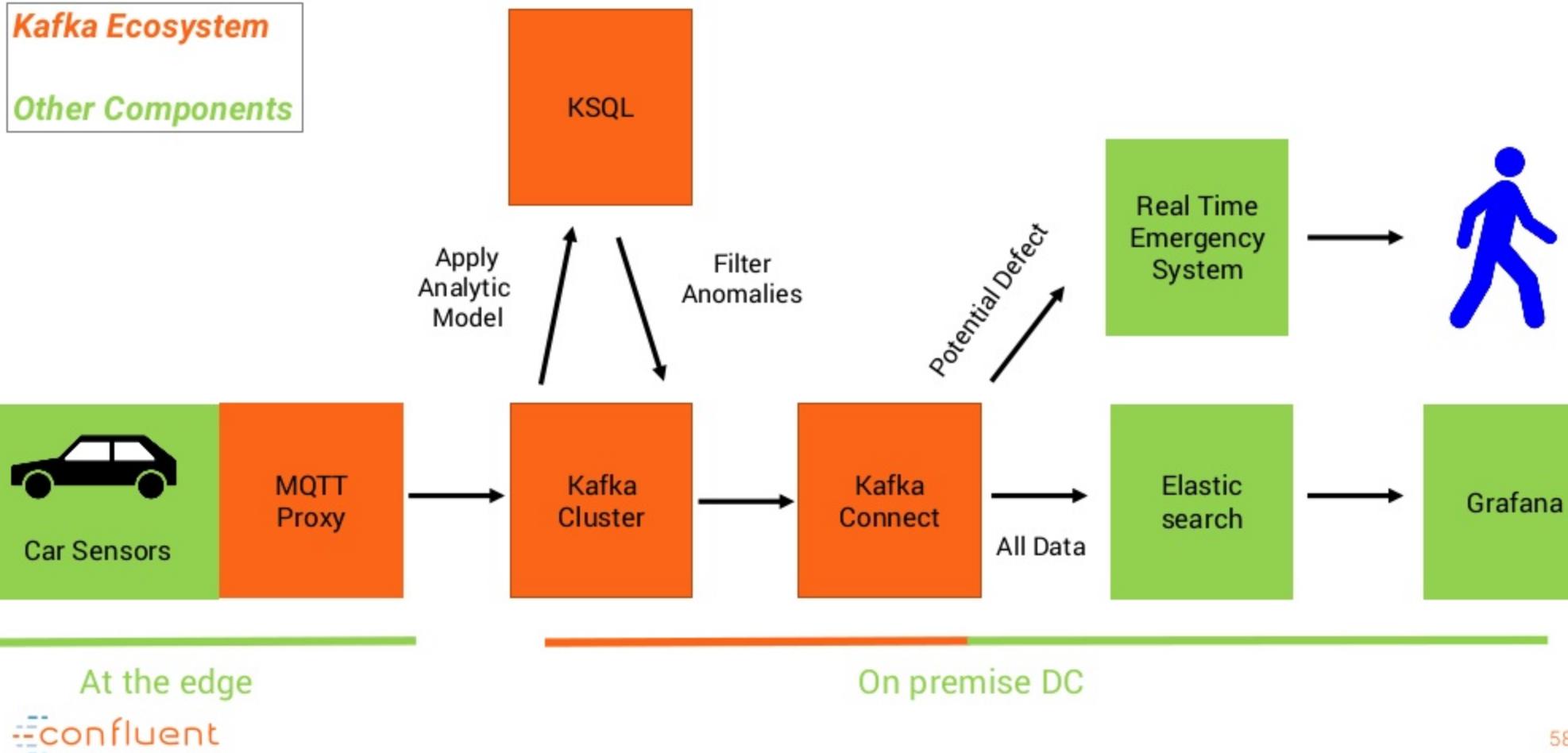


Build vs. Buy
Hosted vs. Managed
Basic vs. Advanced

KSQL and Deep Learning (Auto Encoder) for Anomaly Detection

Kafka Ecosystem

Other Components



Model Training with Python, KSQL, TensorFlow, Keras and Jupyter

Jupyter python-jupyter-apache-kafka-ksql-tensorflow-keras Last Checkpoint: a few seconds ago (autosaved)

File Edit View Insert Cell Kernel Help Trusted Python 3

Setup for the KSQL API:

```
In [1]: from ksql import KSQLAPI
client = KSQLAPI('http://localhost:8088')
```

```
In [3]: client.create_table(table_name='users_original',
columns_type=[ 'regertime bigint', 'gender varchar', 'regionid varchar', 'userid varchar'],
topic='users',
value_format='JSON',
key = 'userid')
```

```
Out[3]: True
```

```
In [6]: client.ksql('show tables')
```

```
Out[6]: [{"#type": "table",
'statementText': 'show tables;',
'tables': [{#type: TABLE,
'name': 'USERS_ORIGINAL',
'topic': 'users',
'format': 'JSON',
'iswindowed': False}]}]
```

Execute sql query and keep listening streaming data:

```
In [4]: query = client.query('select * from users_original')
for item in query: print(item)

("row": {"columns": [[1543510415238, "User_5", 1489124138779, "OTHER", "Region_4", "User_5"]]}, "errorMessage": null, "finalMessage": null}
("row": {"columns": [[1543510415361, "User_8", 1515464455832, "FEMALE", "Region_9", "User_8"]]}, "errorMessage": null, "finalMessage": null}
("row": {"columns": [[1543510415897, "User_2", 1515464455832, "FEMALE", "Region_9", "User_2"]]}, "errorMessage": null, "finalMessage": null}
("row": {"columns": [[1543510416775, "User_3", 1514158220288, "OTHER", "Region_4", "User_3"]]}, "errorMessage": null, "finalMessage": null}
```

Start Backend Services (Zookeeper, Kafka, KSQL)

The only server requirement is a local KSQL server running (with Kafka broker ZK node). If you don't have it running, just use Confluent CLI:

```
! confluent start kafka-server
```

This CLI is intended for development only, not for production
<https://docs.confluent.io/current/cli/index.html>

Using CONFLUENT_CURRENT: /var/folders/0s/zedkbf12yqdnfts11926xle9990gp/T/confluent.0sWkhhs

```
Starting zookeeper
zookeeper is [UP]
Starting kafka
kafka is [UP]
Starting schema-registry
schema-registry is [UP]
Starting ksql-server
ksql-server is [UP]
```

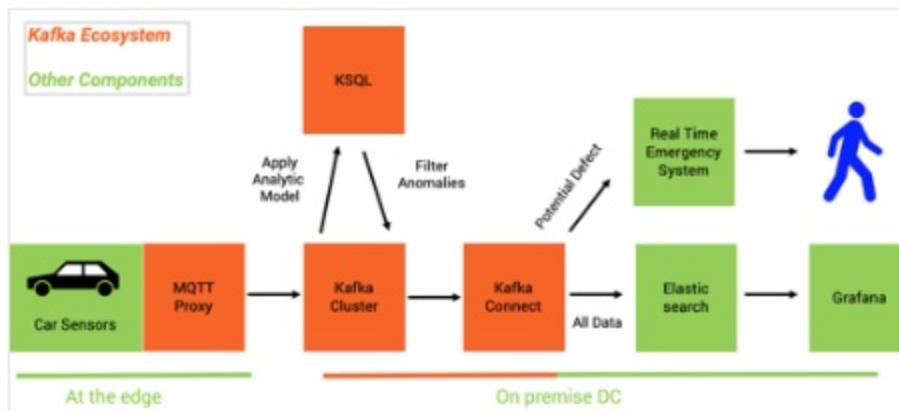
Data Integration and Preprocessing with Python and KSQL

First of all, create the Kafka Topic 'creditcardfraud_source' if it does not exist already:

```
! kafka-topics --zookeeper localhost:2181 --create --topic creditcardfraud_source --partitions 3 --replication-factor 1
```

<https://github.com/kaiwaehner/python-jupyter-apache-kafka-ksql-tensorflow-keras>

Model Deployment with Apache Kafka, KSQL and TensorFlow



```
"CREATE STREAM AnomalyDetection AS  
SELECT sensor_id, detectAnomaly(sensor_values)  
FROM car_engine;"
```

User Defined Function (UDF)

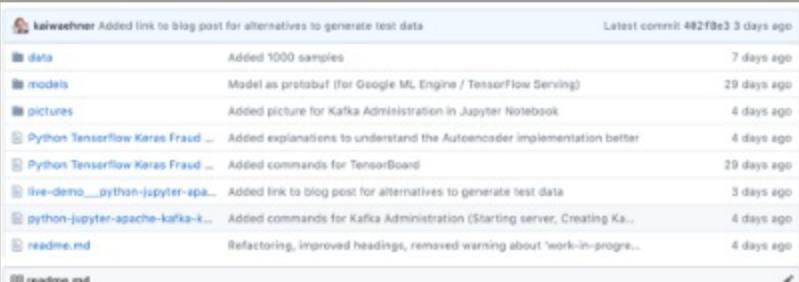




End-to-End Sensor Analytics...

Python, Jupyter Notebook, TensorFlow, Keras, Apache Kafka, KSQL and MQTT

Model Training with Python, KSQL, TensorFlow, Keras and Jupyter

kaiwaehner Added link to blog post for alternatives to generate test data
Latest commit 482f9e3 3 days ago

data Added 1000 samples 7 days ago
models Model as protobuf (for Google ML Engine / TensorFlow Serving) 29 days ago
pictures Added picture for Kafka Administration in Jupyter Notebook 4 days ago
Python Tensorflow Keras Fraud ... Added explanations to understand the Autoencoder implementation better 4 days ago
Python Tensorflow Keras Fraud ... Added commands for TensorBoard 29 days ago
live-demo...python-jupyter-apache... Added link to blog post for alternatives to generate test data 3 days ago
python-jupyter-apache-kafka... Added commands for Kafka Administration (Starting server, Creating Ka... 4 days ago
readme.md Refactoring, improved headings, removed warning about 'work-in-prog... 4 days ago

readme.md

Use Case: Fraud Detection for Credit Card Payments

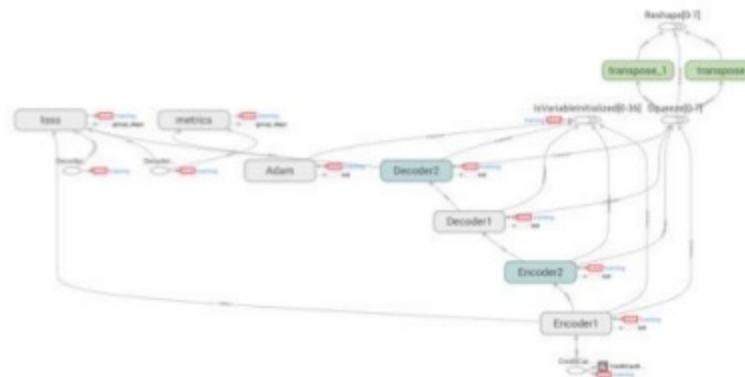
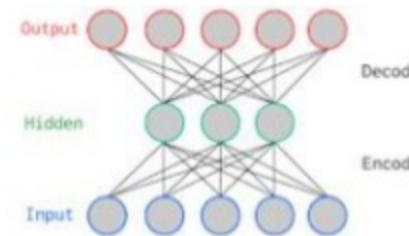
We use test data set from Kaggle as foundation to train an unsupervised autoencoder to detect anomalies and potential fraud in payments.

Focus of this project is not just model training, but the whole Machine Learning infrastructure including data ingestion, data preprocessing, model training, model deployment and monitoring. All of this needs to be scalable, reliable and performant.

Technology: Python, Jupyter, TensorFlow, Keras, Apache Kafka, KSQL

This project shows a demo which combines

- simplicity of data science tools (Python, Jupyter notebooks, NumPy, Pandas)
- powerful Machine Learning / Deep Learning frameworks (TensorFlow, Keras)
- reliable, scalable event-based streaming technology for production deployments (Apache Kafka, Kafka Connect, KSQL).



<https://github.com/kaiwaehner/python-jupyter-apache-kafka-ksql-tensorflow-keras>

Deep Learning UDF for KSQL for Streaming Anomaly Detection of MQTT IoT Sensor Data

<https://github.com/kaiwaehner/ksql-udf-deep-learning-mqtt-iot>

Deep Learning UDF for KSQL for Streaming Anomaly Detection of MQTT IoT Sensor Data

23 commits 1 branch 0 releases 1 contributor Apache-2.0

Kai Waehner and Kai Waehner Fixed order of commands Latest commit df1cc64 14 hours ago

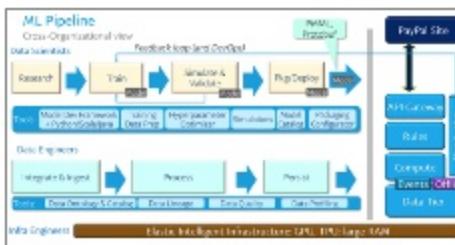
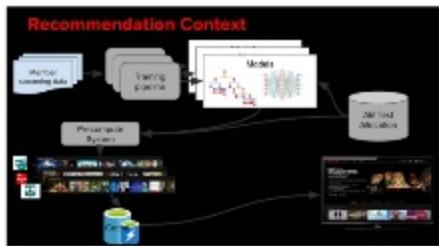
File	Description	Time Ago
pictures	Added picture with MQTT / Kafka architecture	2 days ago
src/main	Removed Kafka Streams code (not needed for the UDF)	2 days ago
LICENSE	Initial commit	8 days ago
README.md	Explained that kafkakat is optional and just a comfortable alternativ...	15 hours ago
live-demo.adoc	Fixed order of commands	14 hours ago
pom.xml	Removed Kafka Streams dependency (not needed for the UDF)	2 days ago
sensor_generator.sh	Bash script which generates new sensor events continuously	15 hours ago

README.md

Deep Learning UDF for KSQL for Streaming Anomaly Detection of MQTT IoT Sensor Data

<https://github.com/kaiwaehner/ksql-udf-deep-learning-mqtt-iot>

Comparing our current project status to others



NETFLIX

UBER

PayPal

Well,
we are not there yet,
but getting closer
every month!

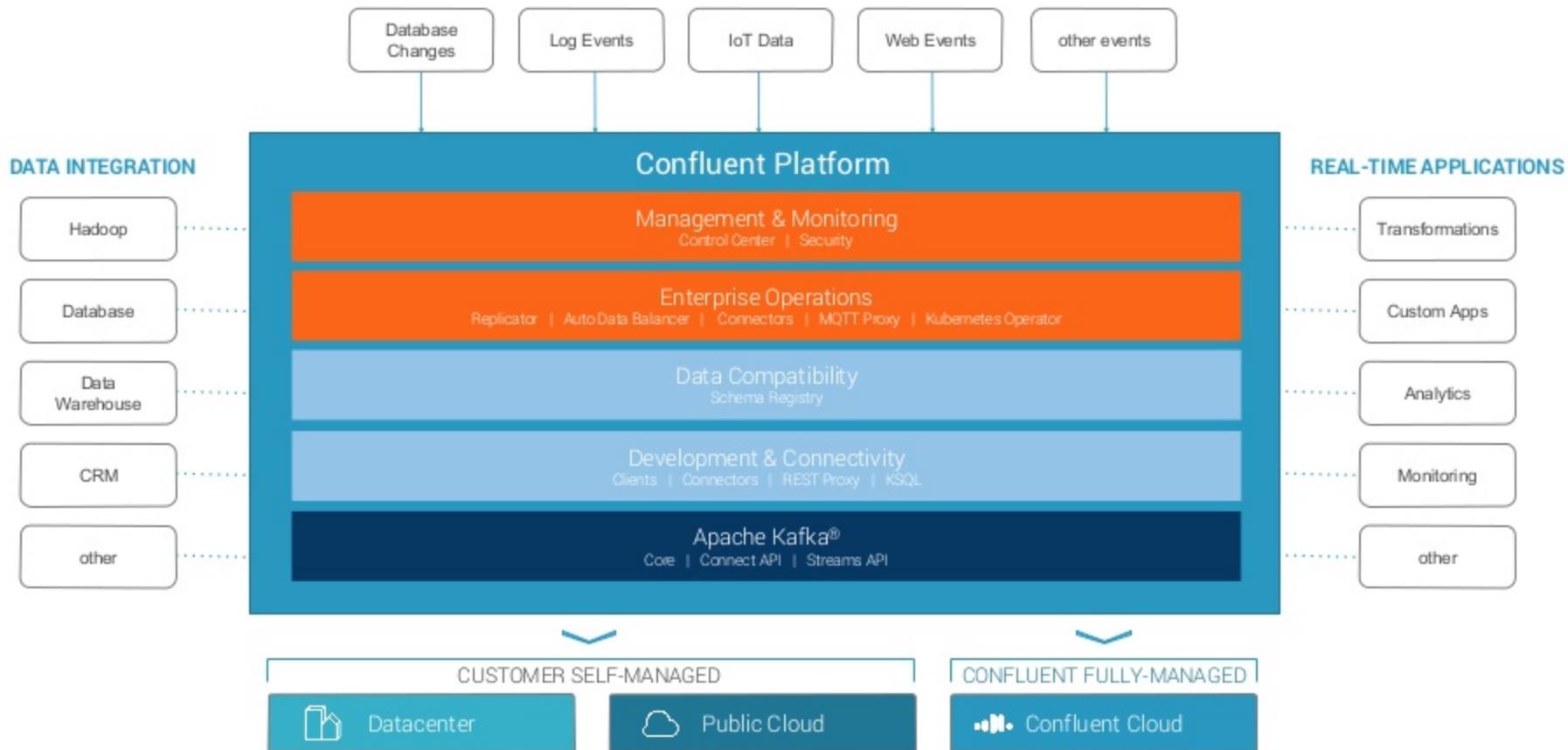
Poll

Which of the following use cases are you most likely to utilize Kafka for over the next year?

1. Data ingestion (processing e.g. batch in Spark)
2. Data pipeline (processing with Kafka)
3. Stream processing (e.g. Kafka Streams, KSQL)
4. Stream processing with machine learning
5. Other (like microservices, event sourcing, storage)



Confluent Delivers a Mission-Critical Event Streaming Platform



Best-of-breed Platforms, Partners and Services for Multi-cloud Streams

SELF MANAGED

FULLY MANAGED



heptio

MCSOSPHERE

Pivotal
Container Service

RED HAT
OPENSHIFT

aws

GoogleCloudPlatform

Microsoft
Azure

confluent cloud
Apache Kafka as a Service



Private Cloud

Deploy on bare-metal, VMs, containers or Kubernetes in your datacenter with **Confluent Platform** and **Confluent Operator**

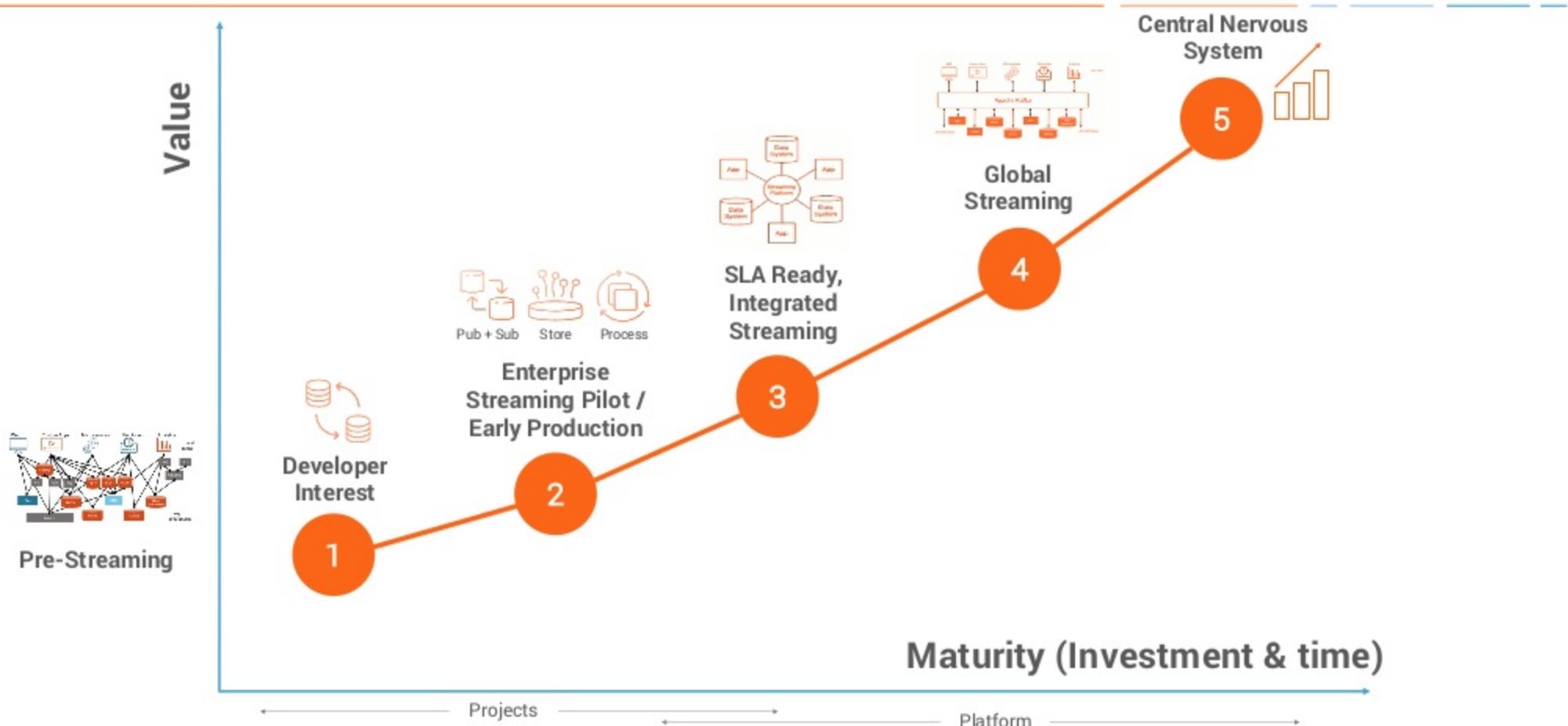
Hybrid Cloud

Build a persistent bridge between datacenter and cloud with **Confluent Replicator**

Public Cloud

Implement **self-managed** in the public cloud or adopt a **fully managed service** with **Confluent Cloud**

Confluent's Streaming Maturity Model - where are you?



Resources and Next Steps



<https://confluent.io>



<https://confluent.io/cloud>



[#confluent-cloud](https://slackpass.io/confluentcommunity)



@confluentinc

Rate today's session → Please rate my session!

Cyberconflict: A new era of war, sabotage, and fear

See passes & pricing

David Sanger (The New York Times)
9:55am-10:10am Wednesday, March 27, 2019
Location: Ballroom

Secondary topics: Security and Privacy

[Rate This Session](#)

We're living in a new era of constant sabotage, misinformation, and fear, in which everyone is a target, and you're often the collateral damage in a growing conflict among states. From crippling infrastructure to sowing discord and doubt, cyber is now the weapon of choice for democracies, dictators, and terrorists.

David Sanger explains how the rise of cyberweapons has transformed geopolitics like nothing since the invention of the atomic bomb. Moving from the White House Situation Room to the dens of Chinese, Russian, North Korean, and Iranian hackers to the boardrooms of Silicon Valley, David reveals a world coming face-to-face with the perils of technological revolution—a conflict that the United States helped start when it began using cyberweapons against Iranian nuclear plants and North Korean missile launches. But now we find ourselves in a conflict we're uncertain how to control, as our adversaries exploit vulnerabilities in our hyperconnected nation and we struggle to figure out how to deter these complex, short-of-war attacks.

David Sanger

The New York Times

David E. Sanger is the national security correspondent for the New York Times as well as a national security and political contributor for CNN and a frequent guest on CBS This Morning, Face the Nation, and many PBS shows.

[Add to Your Schedule](#)
[Add Comment or Question](#)

Session page on conference website

✓ Attending

Notes

Remove

Cyberconflict: A new era of war, sabotage, and fear

9:55 AM - 10:10 AM, Wed, Mar 27, 2019

Speakers



David Sanger
National Security Correspondent
The New York Times

Ballroom

Keynotes

David Sanger explains how the rise of cyberweapons has transformed geopolitics like nothing since the invention of the atomic bomb. From crippling infrastructure to sowing discord and doubt, cyber is now the weapon of choice for democracies, dictators, and terrorists.

SESSION EVALUATION

O'Reilly Events App



Questions? Feedback? Let's connect!

Kai Waehner

Technology Evangelist

kontakt@kai-waehner.de

@KaiWaehner

www.kai-waehner.de

www.confluent.io

[LinkedIn](#)

