Apache Kafka

And Event Driven Modelling

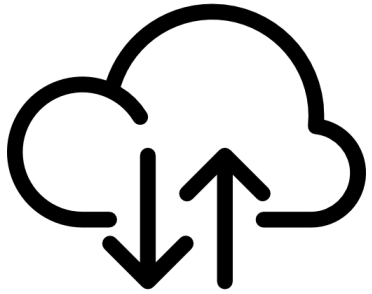
Dr. Felix Aller

Solution Architect

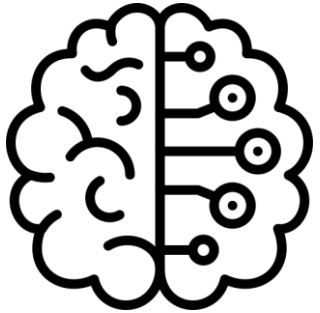
The world of data has changed …

… not recently, but over the past couple of years:

Mobile



Cloud



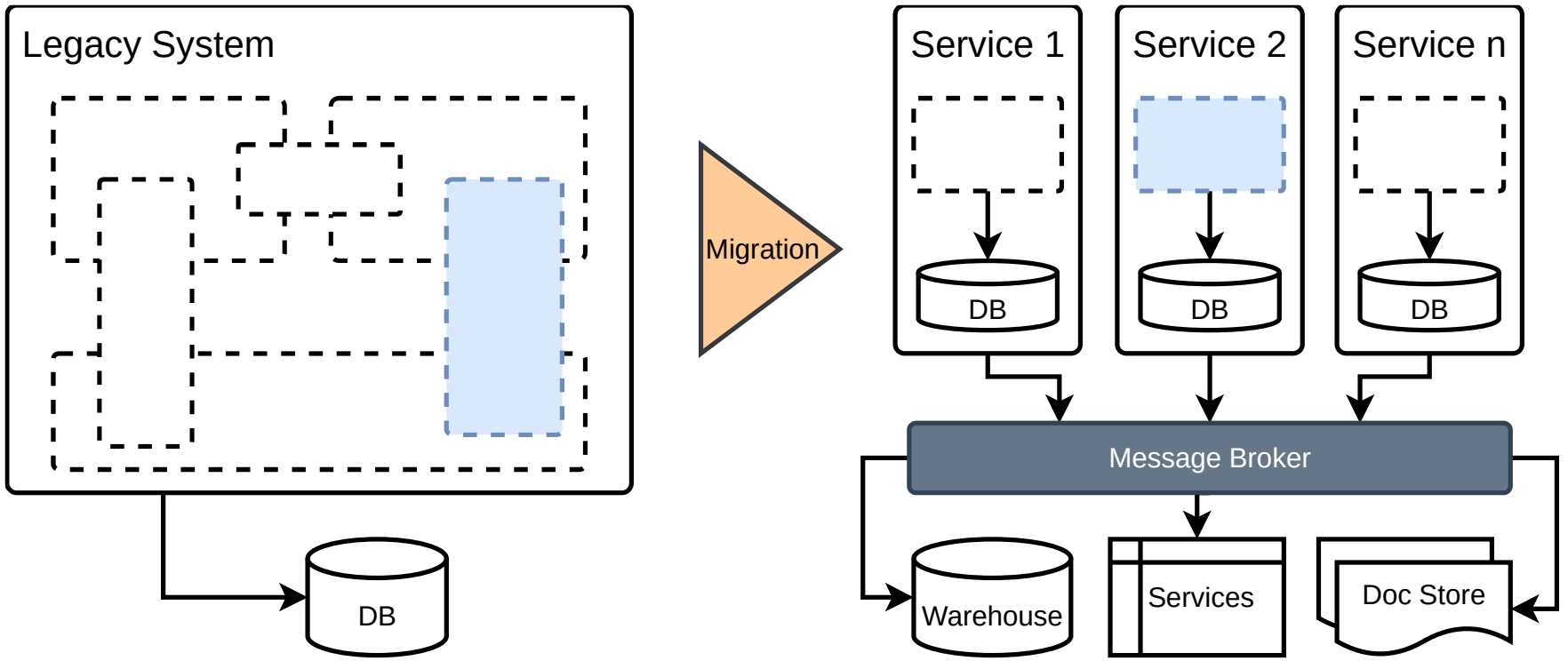
Machine Learning

Microservices

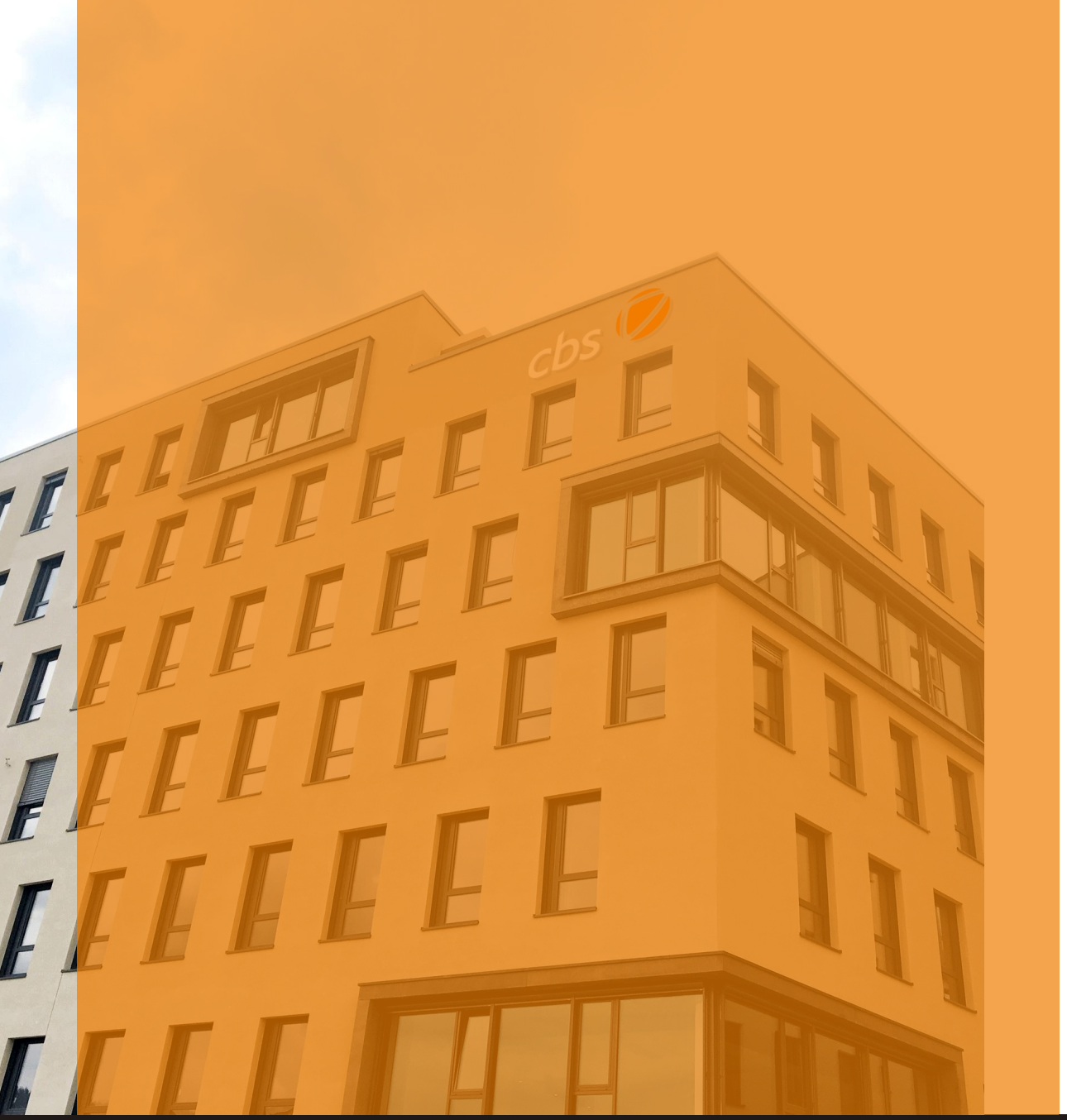
IoT

Digitalization trend drives the need for new requirements with respect to increasing speed, scale and efficiency.

Modernization of Monolithic Legacy Applications



Separation of source and target systems



- Events
- Event Driven Architecture / History
- Kafka Architecture
- SAP Integration and Kafka

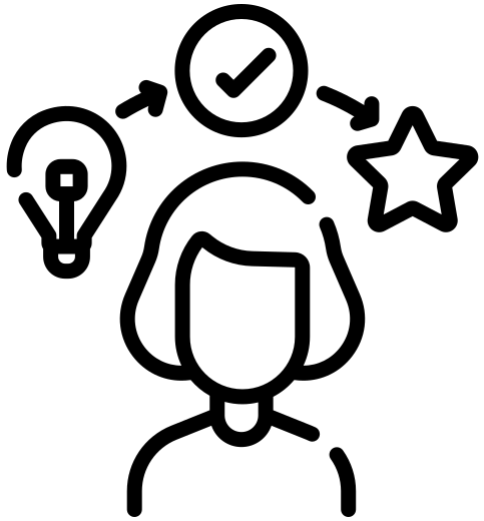
Events

Something happened, but what?

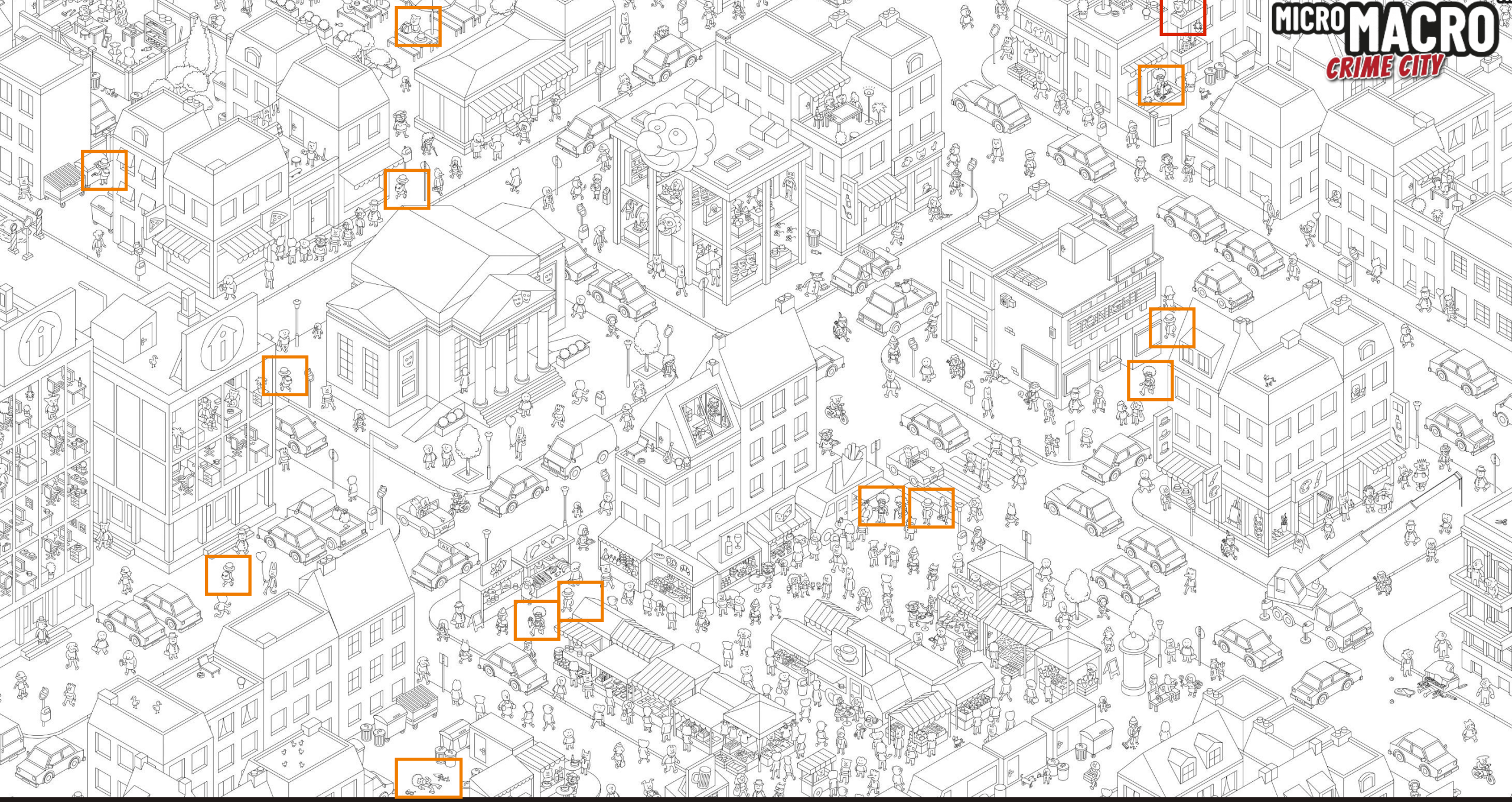
Events

Sale

Invoice



Experience


MICRO MACRO
CRIME CITY
TONIGHT!

Event Sourcing: Thinking of Data in Events

Starting Position

\+

n-Moves

=

Ending Position

CRUD Data Model

Color	Piece	Pos
Black	King	F7
White	Queen	G4
Black	Queen	D7
White	Knight	G5
...		

Event Data Model

Time	Event
11:42	Black Queen to D7
11:44	White Knight to G5
11:45	Black King to G8
11:46	White Queen to D7
...	

Order of Events

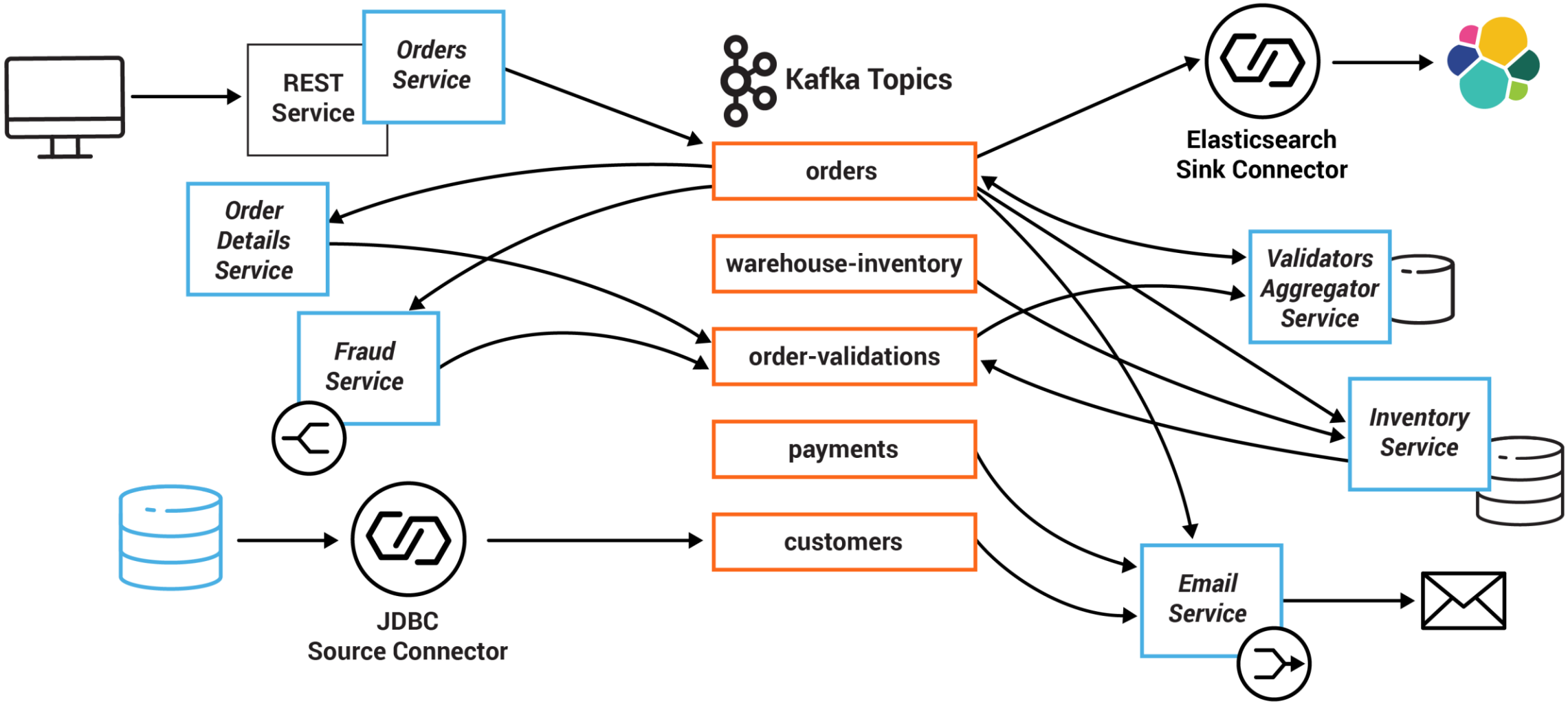
Derive

Event History is retained.

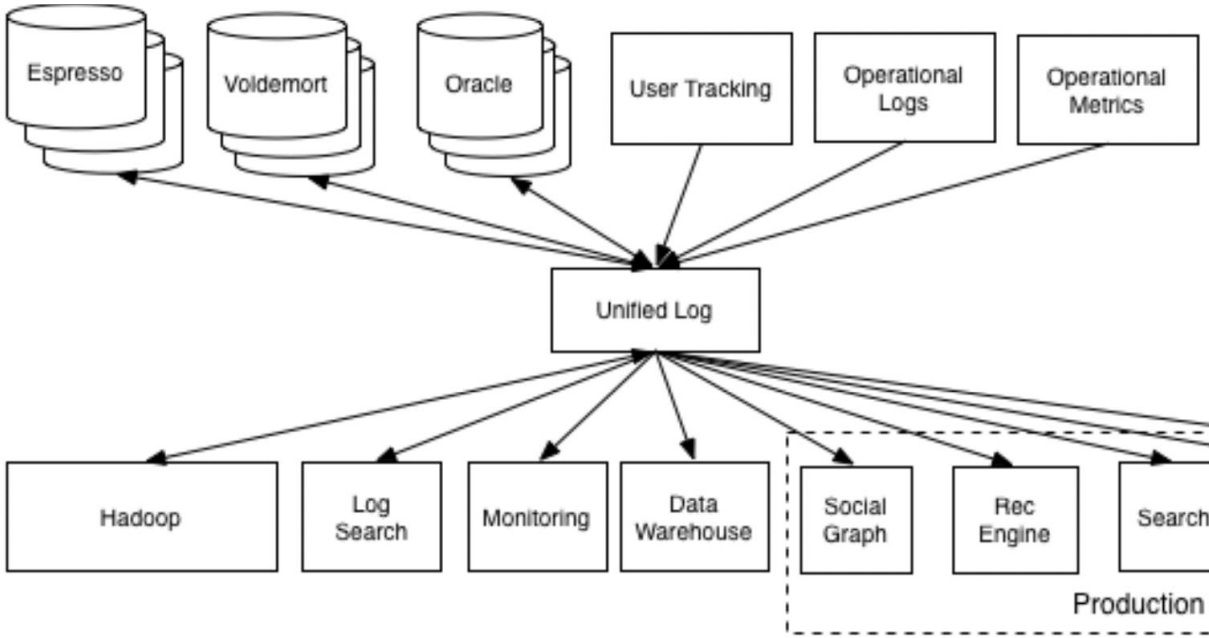
Event Driven Architecture

Using Apache Kafka

Streaming Platform for event-driven architecture



Source https://docs.confluent.io/



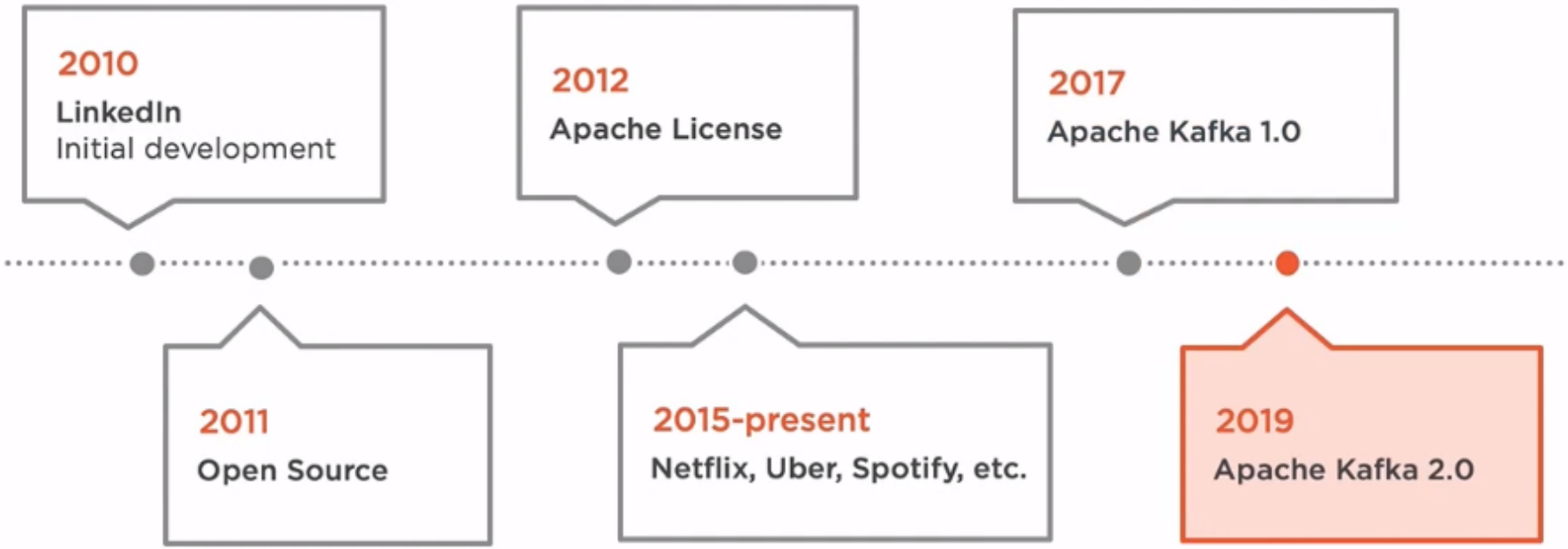
- Developed 12 years ago
- First use case - no other technologies available
- Handle big volumes of log data using a ‘unified log’
- Implemented in Scala and Java

“A unified platform for handling all the real-time data feeds a large company might have.”

LinkedIn Use Case: > 4.5 trillion messages a day

Netflix Use Case: > 6 Petabytes a day

Source https://linkedin.com/



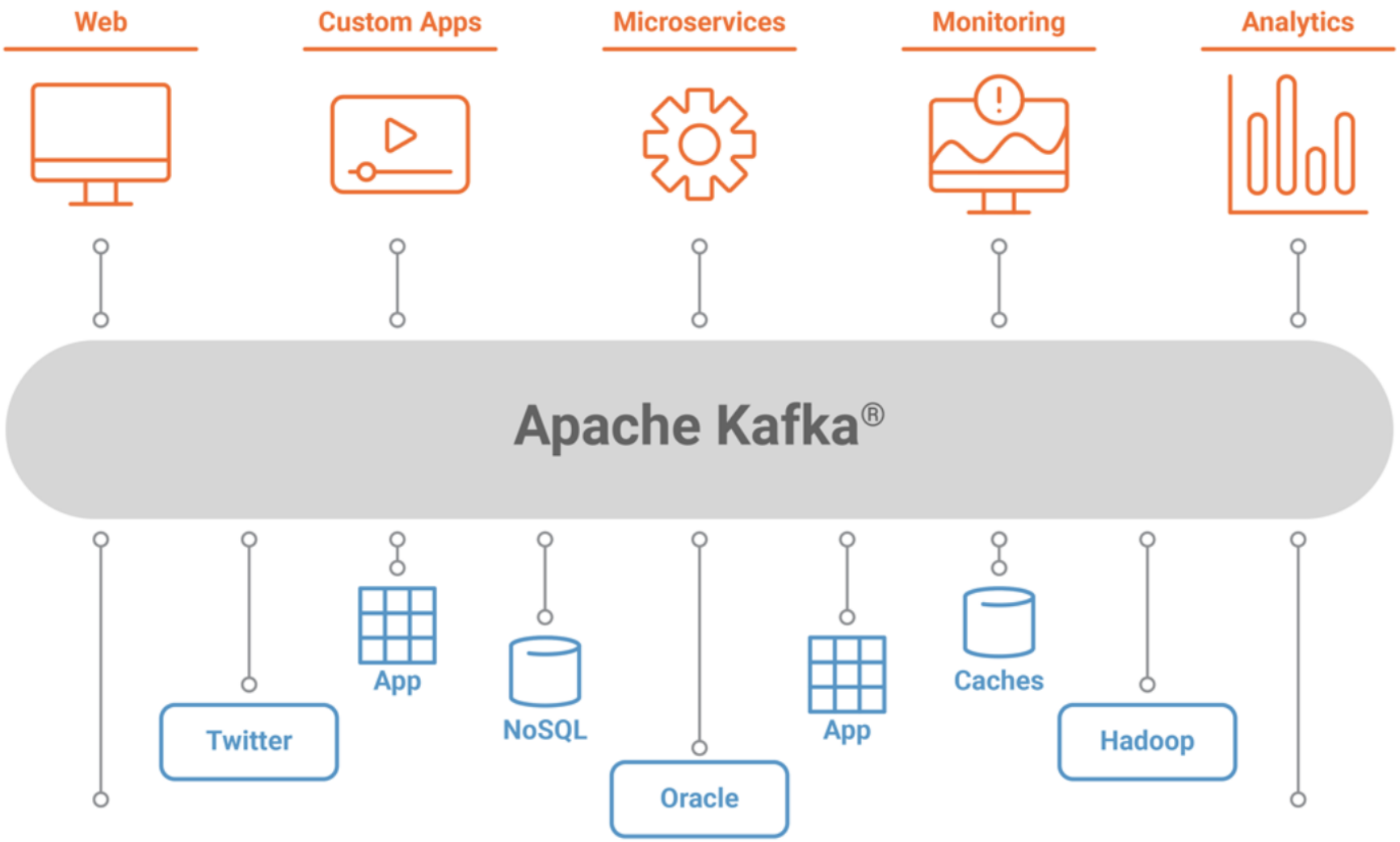
- Originally developed at LinkedIn
- Subsequently open sourced in 2011
- Graduation from the Apache Incubator occurred on 23 October 2012
- Name after author Franz Kafka because it is "a system optimized for writing" - co-creator (Jay Kreps) liked Kafka's work

Kafka central middleware

- Global scale
- Real-time
- Distributed storage
- Persistent storage
- Stream processing

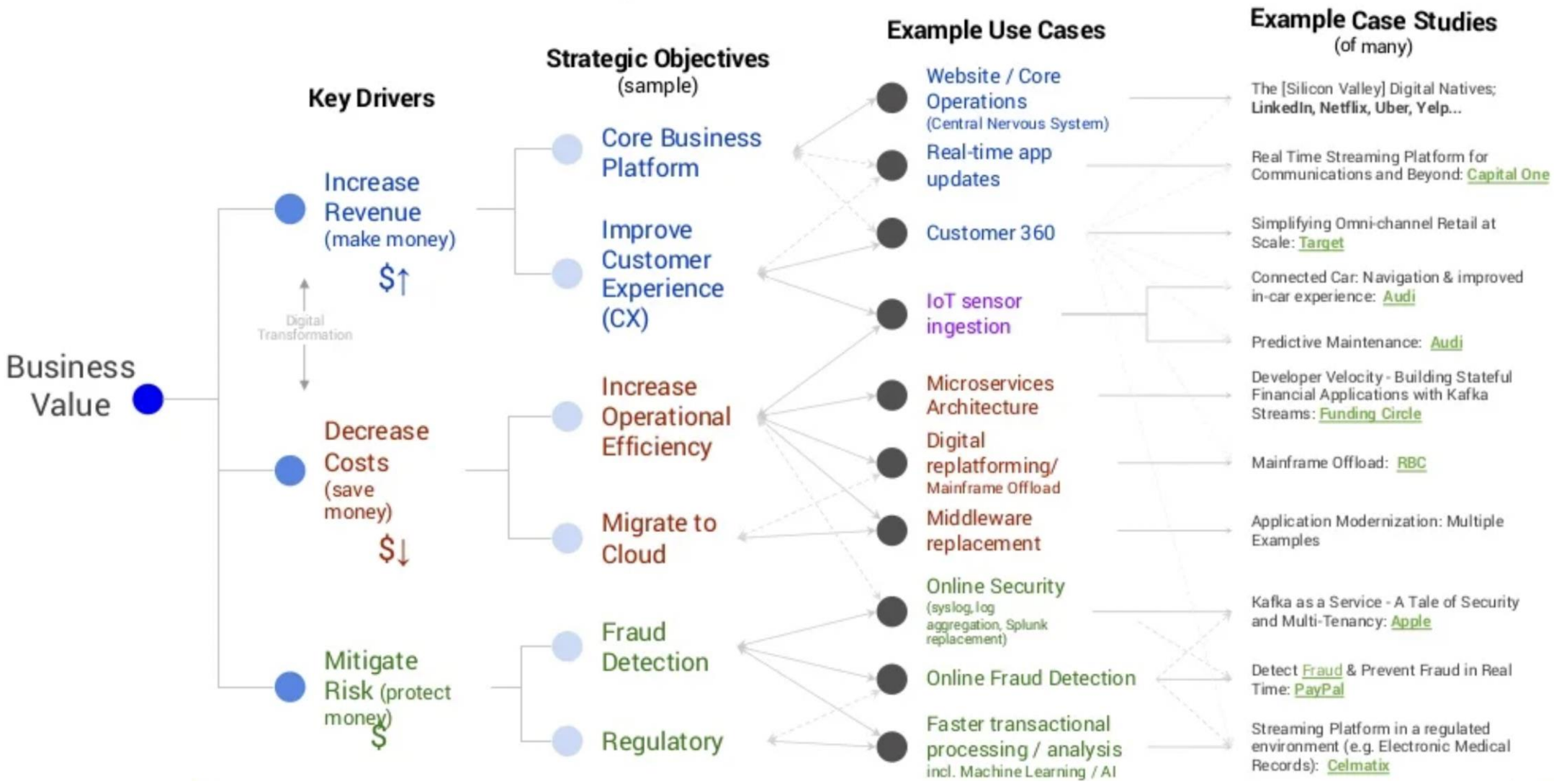
→ Kafka not only for tech giants

→ Kafka not only for big data



Source https://docs.confluent.io/

Event Streaming Value per Use Case

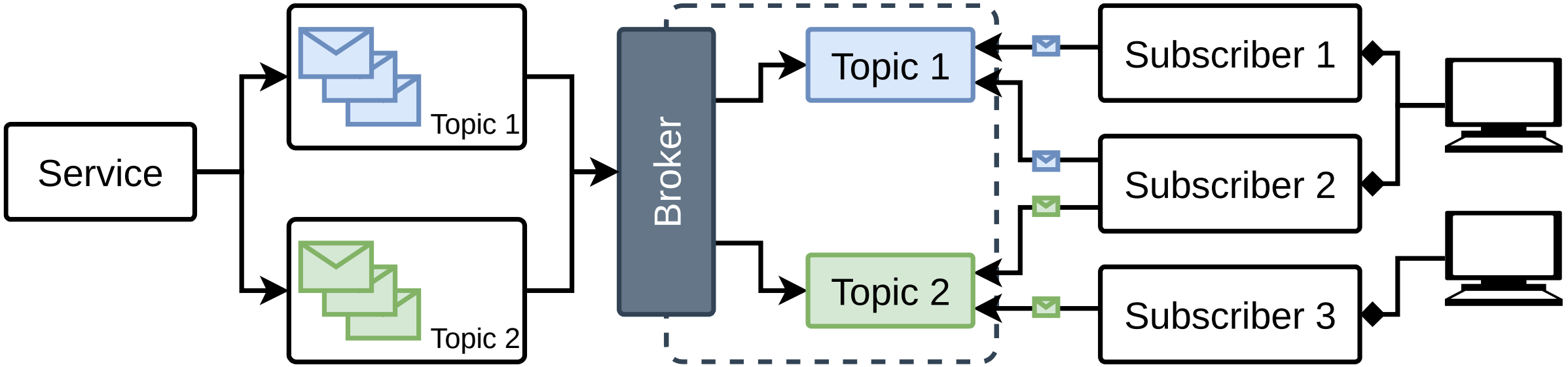


Source https://docs.confluent.io/

Kafka Architecture

From Producer to Consumer

Stream processing is a generalization of batch processing



Messaging, Storing, Processing → Distributed, Immutable, Append-Only Commit-Log

Terminology

Record

- Consits of key, value, timestamp
- Keys optionally assigned
- Keys help the overall partitioning
- Can be published to one partition
- Typically AVRO or JSON
- Records/Events represent the actual information

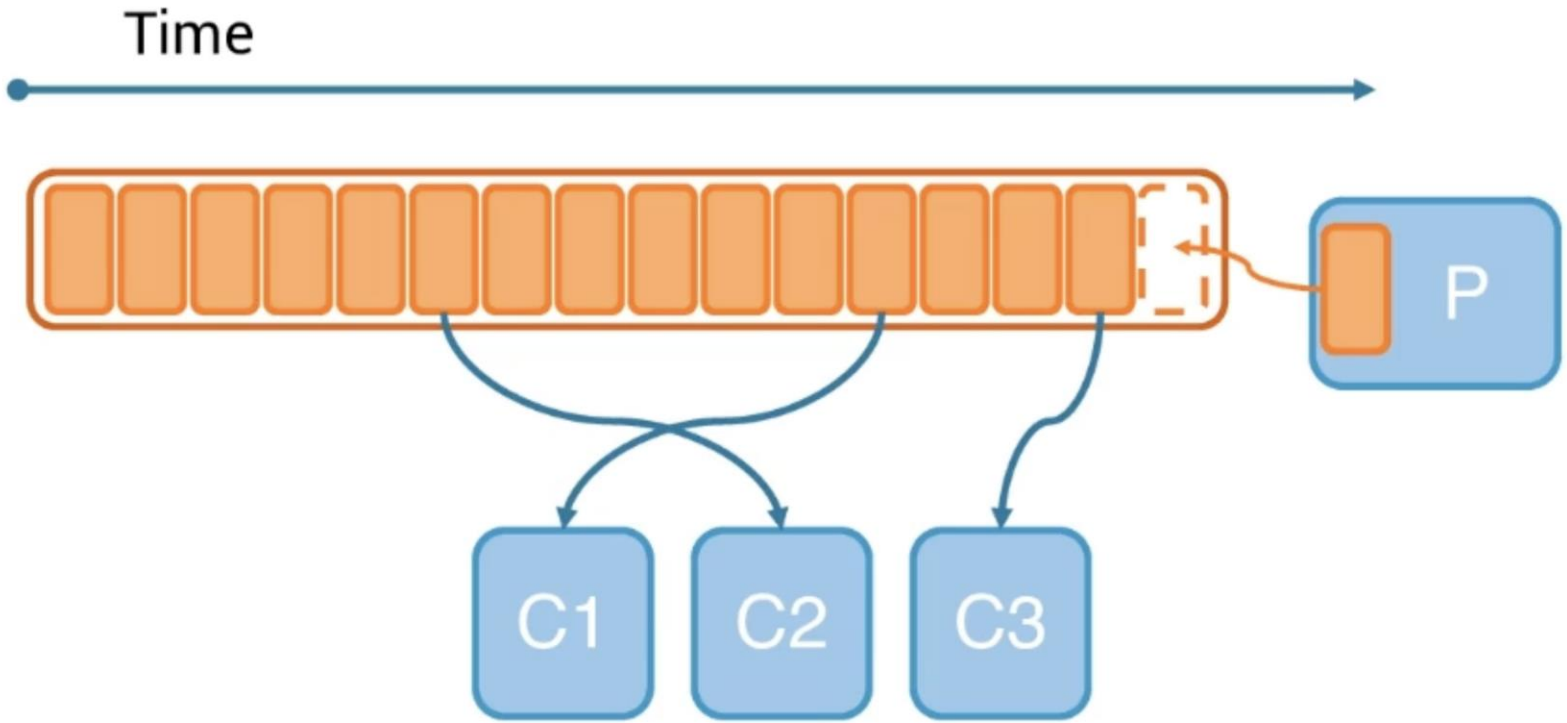
Topic

- Category of records
- Consumer pulls records from topic
- Has one or more partitions
- Each partition is a log of records
- Records are enqueued on the end of the topic and consumed on the same one

Partition

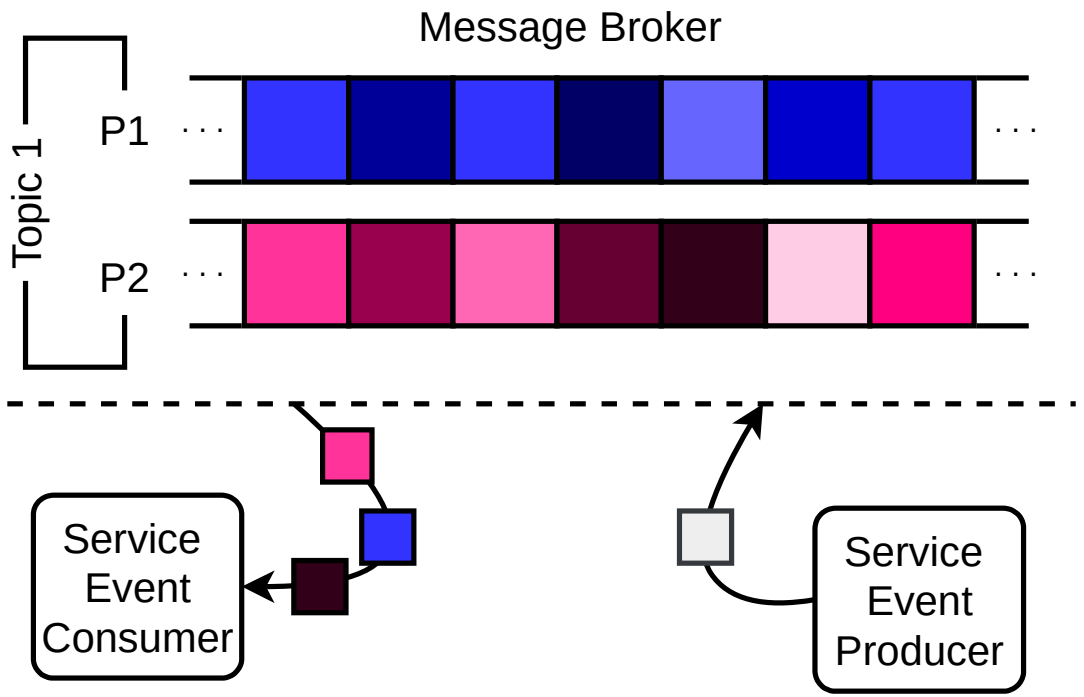
- Topics are spread among partitions
- Partitions enable parallelism
- Should fit into a single Kafka server
- Number of partitions determine the amount of parallelism

Producing to Kafka



Source https://docs.confluent.io/

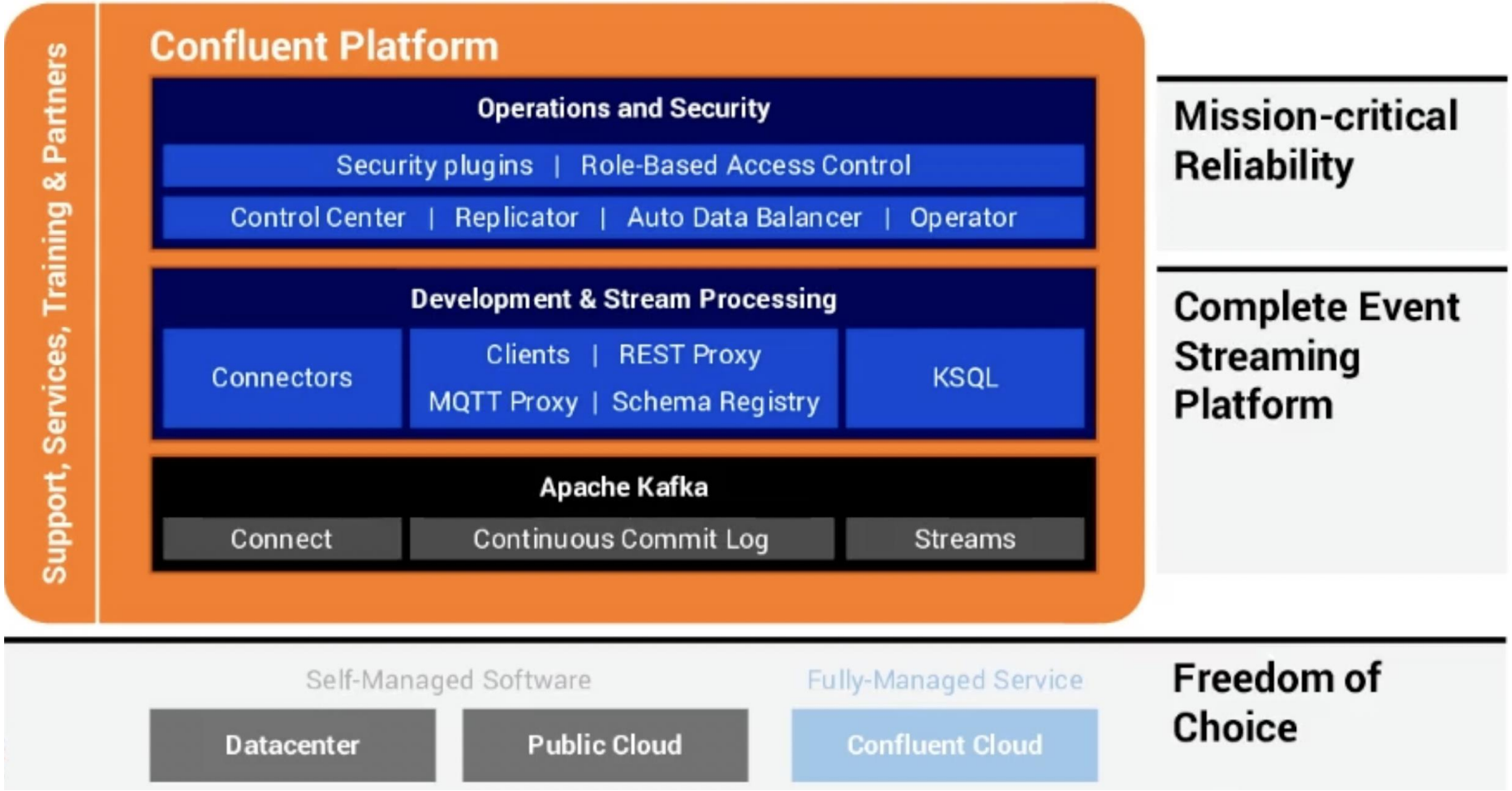
Message Broker Parallelism



Decoupling of Topics:

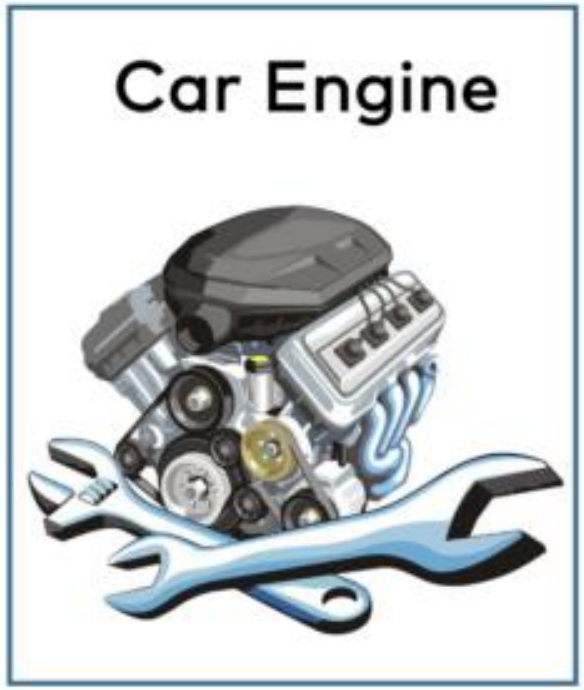
- Messaging
- Scalability
- Event Storage and Replay
- Stream Processing

Confluent Streaming Platform

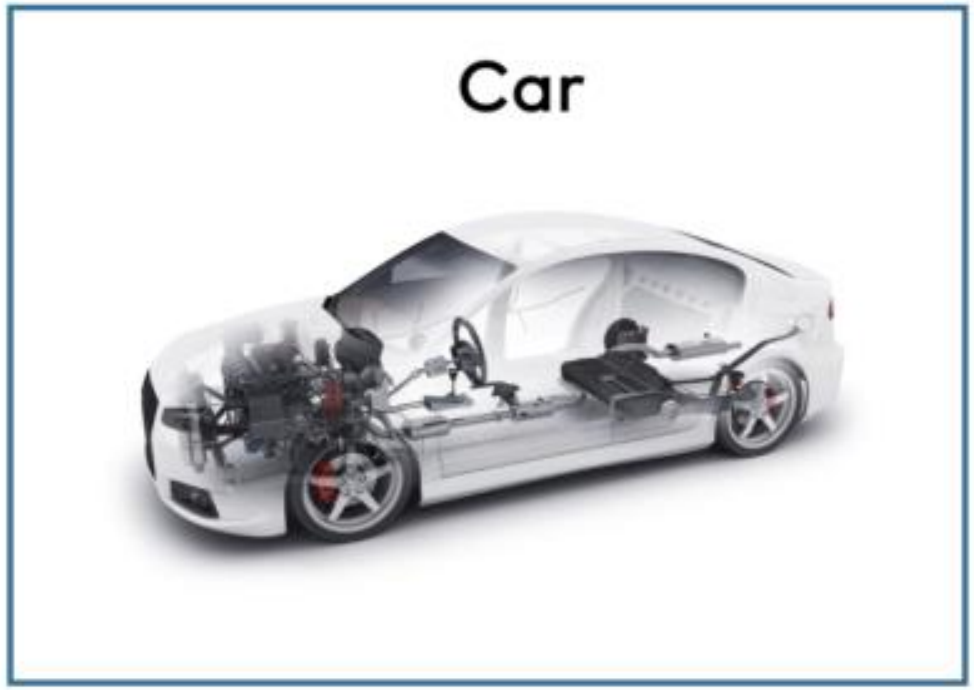


Source https://docs.confluent.io/

Kafka vs Confluent vs Confluent Cloud



Kafka



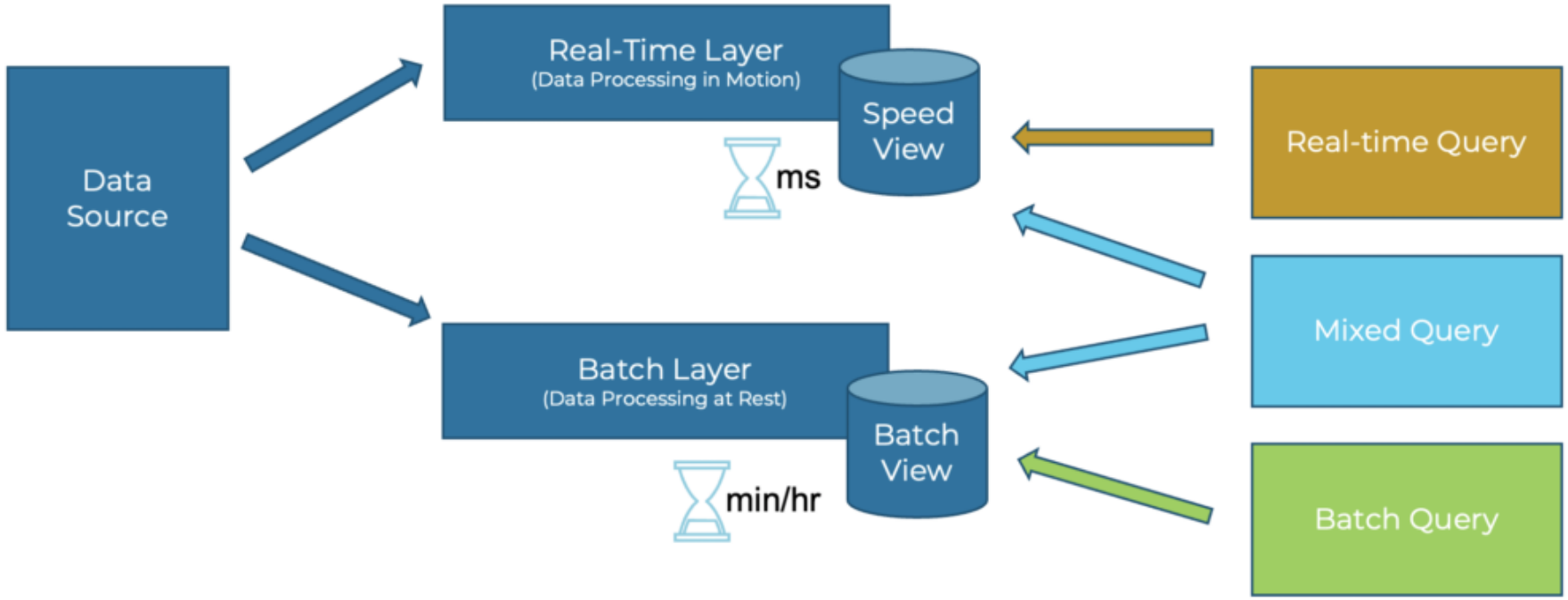
Confluent

Confluent Cloud

Source https://kai-waehner.de/

Lambda Architecture

Option 2: Separate serving layers

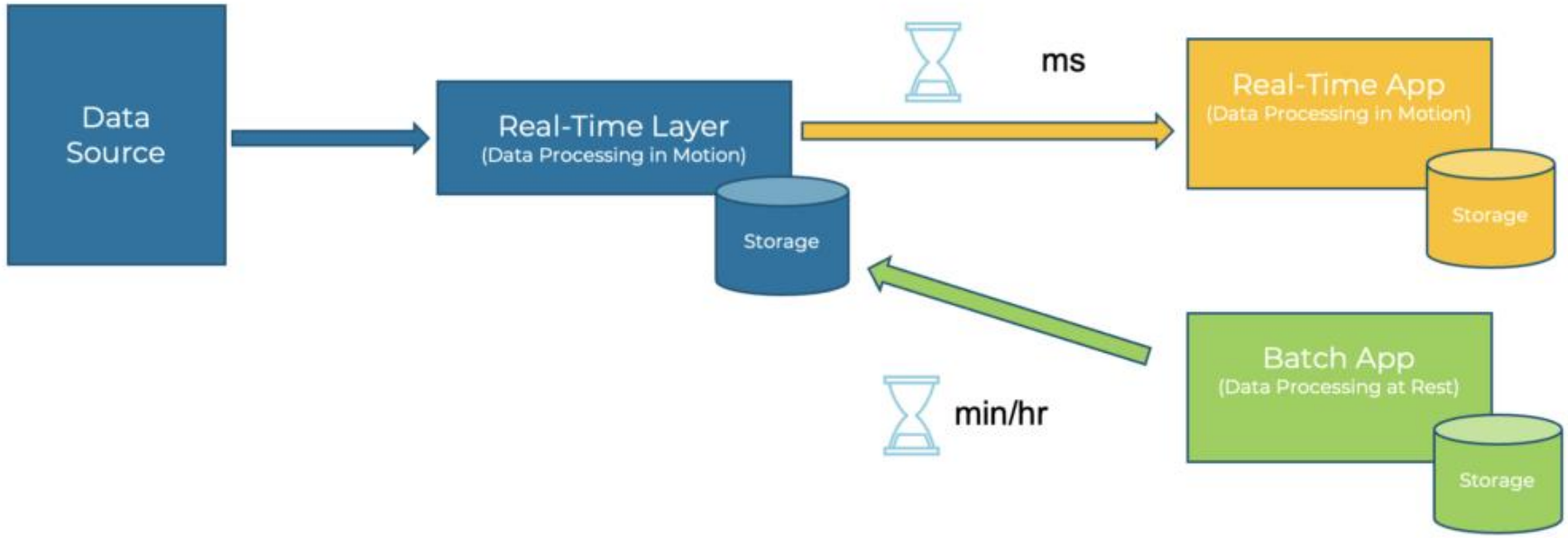


Why not use Lambda:

- Duplicate Code
- Data Quality
- Added Compexity
- Two distributed systems

Kappa Architecture

One pipeline for real-time and batch consumers



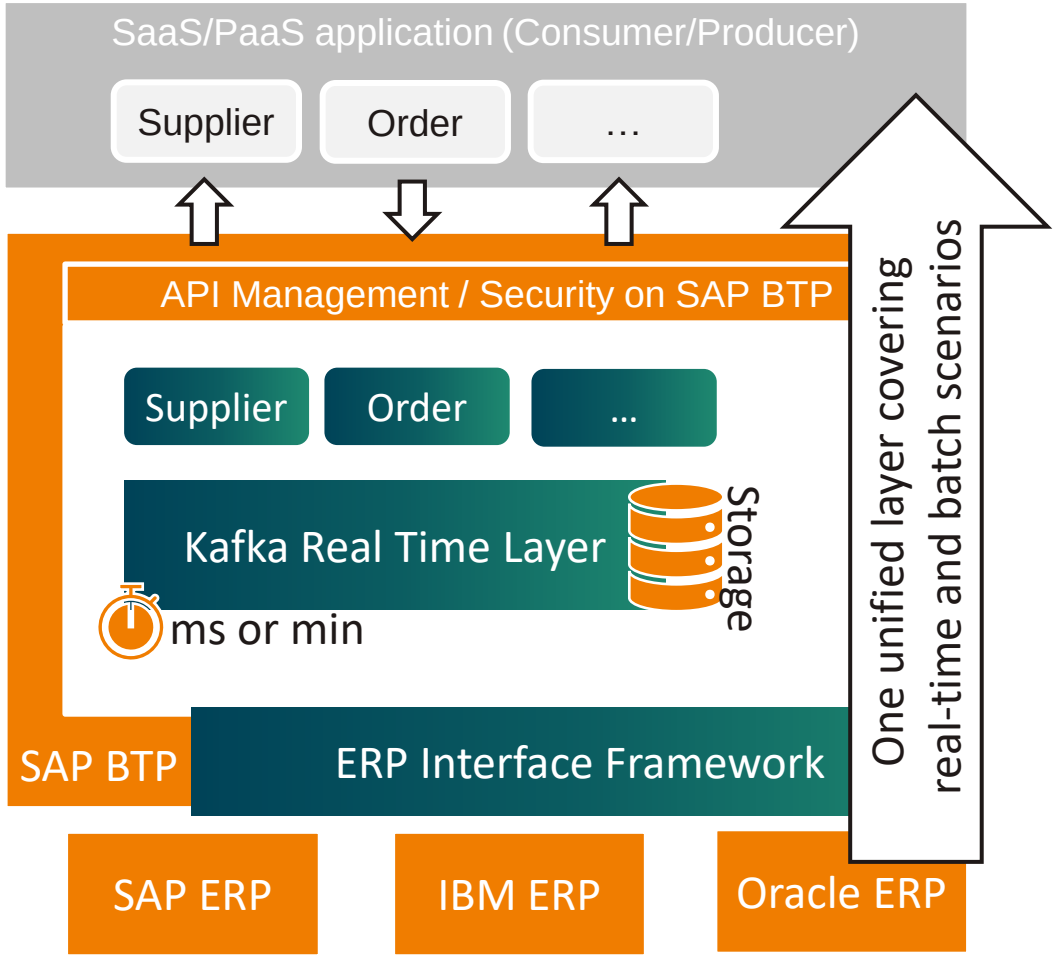
Why use Kappa:

- Handle use cases with single architecture
- One codebase always in synch
- One set of infrastructure and technology
- The heart of the infrastructure is real-time, scalable, and reliable
- Improved data quality with guaranteed ordering and no mismatches
- No need to re-architect for new use cases

SAP Integration and Kafka

Enabler for Enterprise System Generalisation

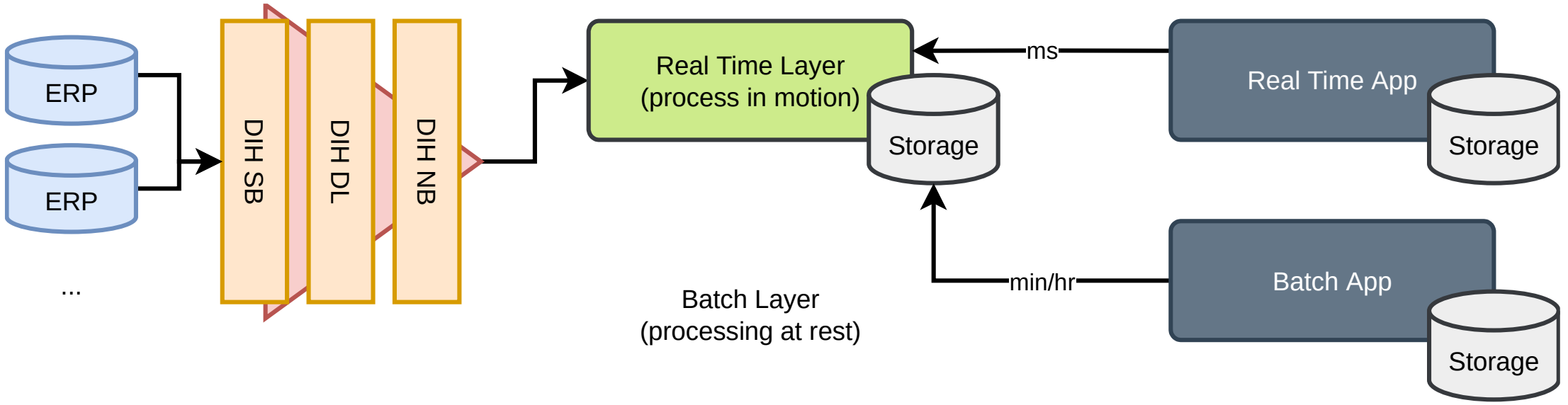
Digital Integration Hub (DIH) and Kafka: Real-Time Data Transfer and Processing



- **Kafka & SAP Synergy:** Enables real-time data processing and transfer of both real-time AND historical data with one infrastructure.
- **Consolidated Business Objects:** Streamlines data organization and access by creating unified business objects in the Kafka-SAP integrated ecosystem.
- **Kafka and SAP Cloud Integration:** ONE platform for future enhancements.

Empowering Real-time Decision-making for Dynamic Business Agility

Objective: Generalized Data Stream



DIH Integration of enterprise systems:

- Based on existing middleware
- DIH connectors well suited and well-proven
- No rewriting of existing functionality

Single Technology Stack

- Handle all use cases
- Codebase always in sync
- Core is real-time, scalable, reliable

Various use cases / processing paradigms:

- Realtime data ingestion at high throughput
- Batch processing for analytical modelling
- Customer interactions for logistics, manufacturing, and many other use cases

Digital Integration Hub on Azure

Design Choices:

- Writing layer as REST API
- Streaming layer based on Writing layer
- Event driven connection to streaming platform
- Serveless functions handles data inquiries
- Data is stored in BLOB storage and as topic information