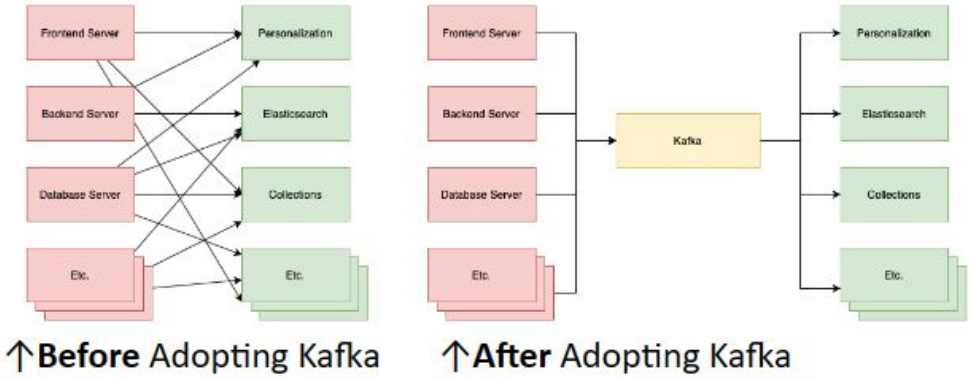
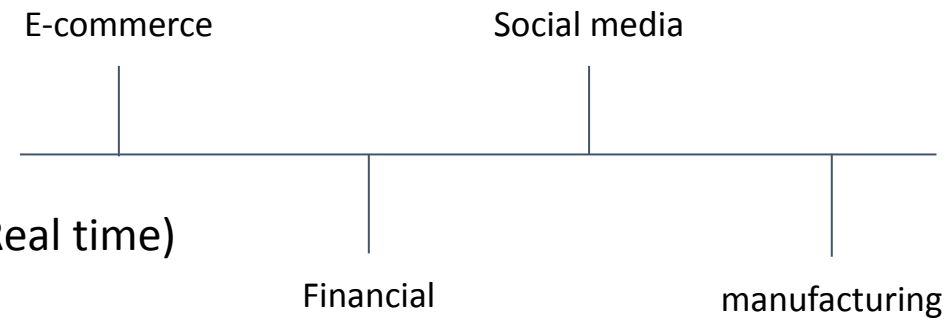
Development Backgrounds 6 Ws (1)

- Story of Why
  - **In 2010, LinkedIn** found **Getting the Big, frequency data from source systems and reliably moving** it around was very difficult.
- Why difficult?
  - Different format and storage methods of data
  - Lost or corrupted during the movement
  - Without real-time messaging solutions. (before kafka)
- What is Kafka
  - As a publish-subscribe system
  - With publishers, topics, and subscribers
  - Partition topics and enable massively parallel consumption (Real time)
- Out scope(Difference between conventional approach)
  - Kafka does not have individual message IDs.
  - Kafka does not track the consumers that a topic has or who has consumed what messages





[Kafka's origin story at LinkedIn]

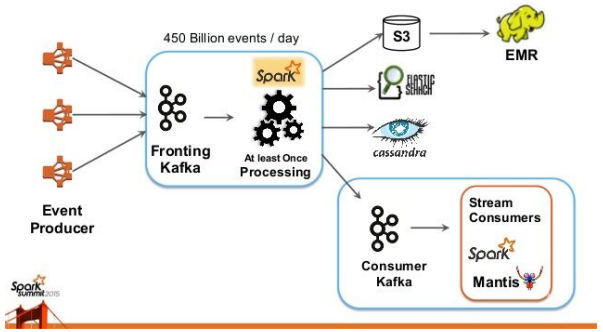
Development Backgrounds 6 Ws (2)

WHO

Apache Software Foundation

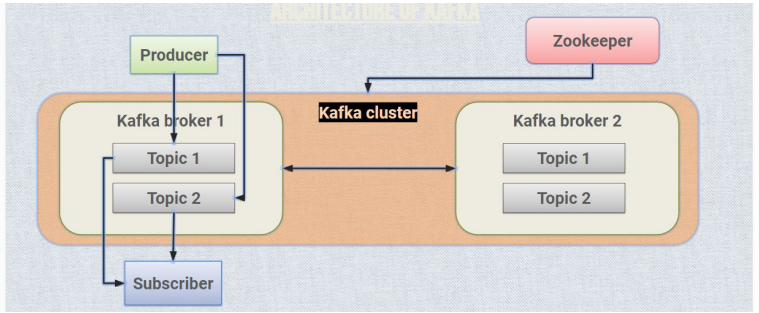
WHY

Real time processing



WHAT

Message Broker



WHEN

Created (2010) Adopted (2011)

LinkedIn

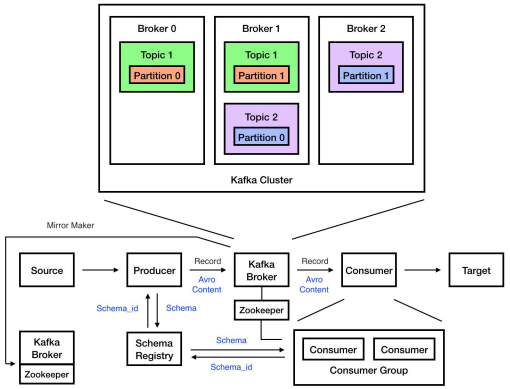
WHERE

Linkedin, paypal

LinkedIn

HOW

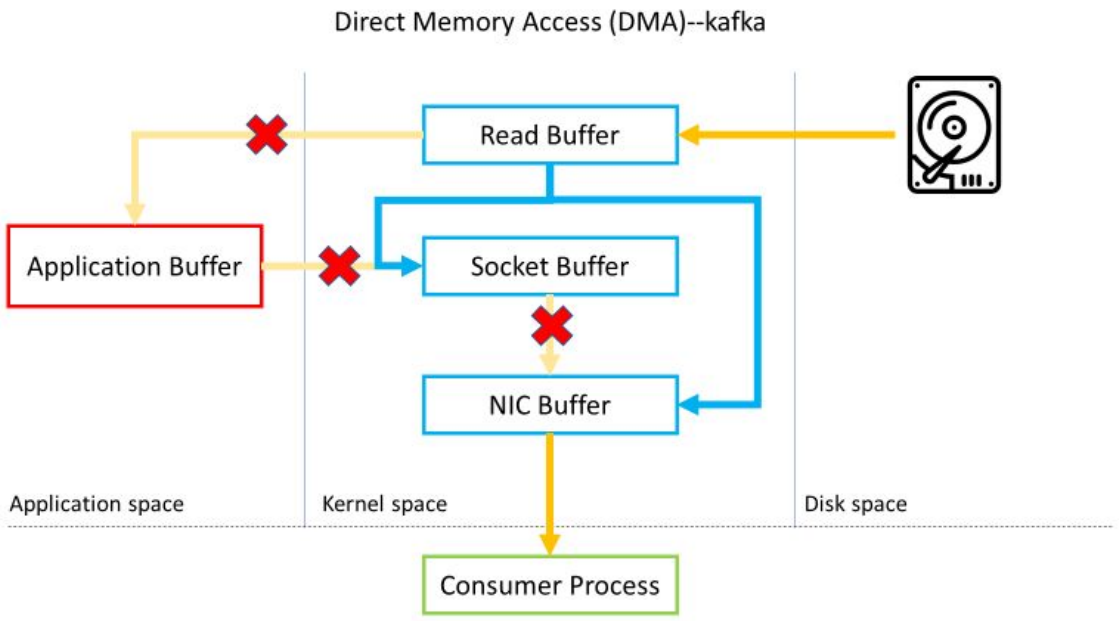
Kafka architecture



Key Features & Benefits (1)

High Throughput

- Capable of handling high-velocity and high-volume data.
- Processes millions of messages per second.



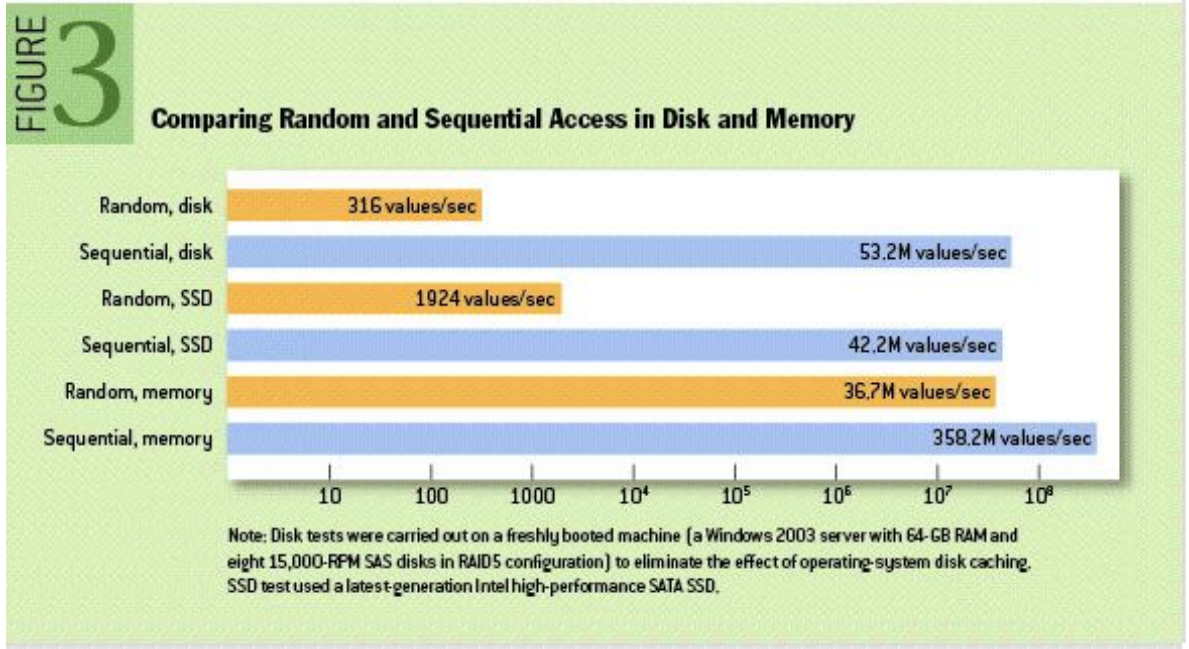
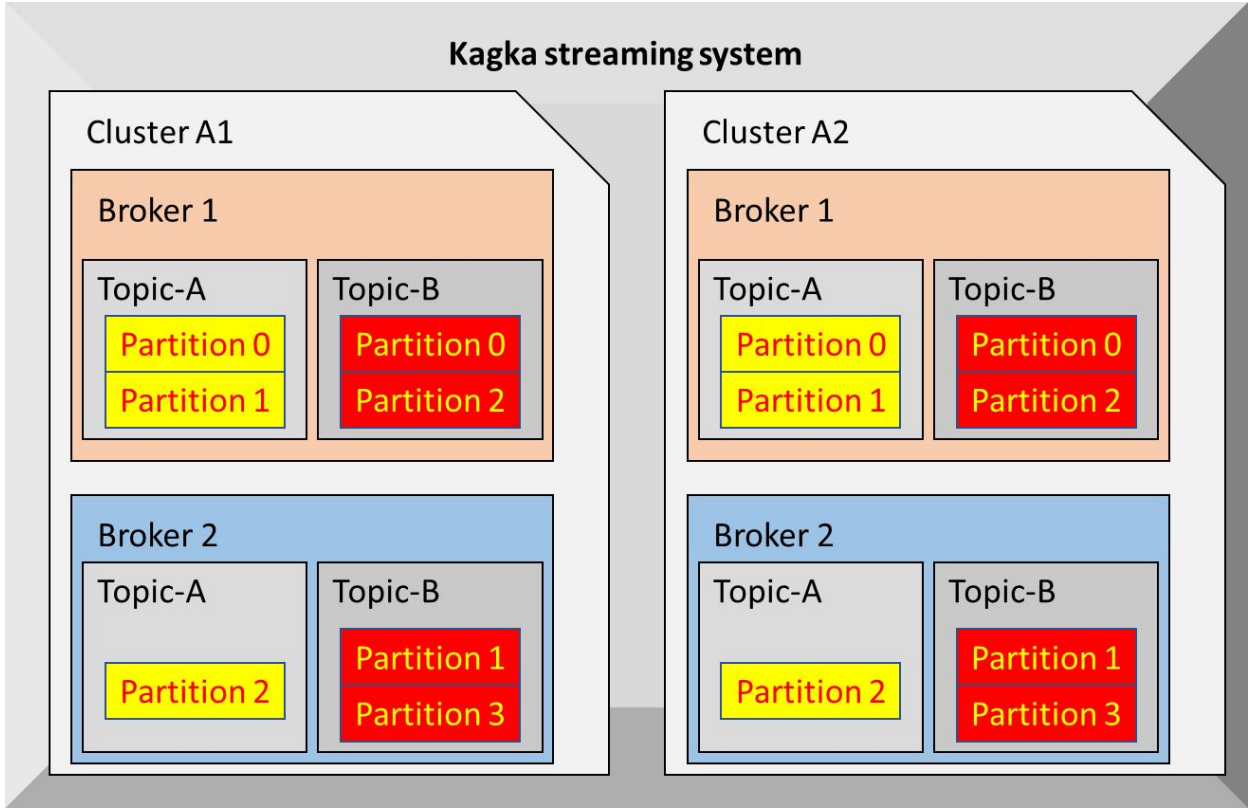


Image Source:https://queue.acm.org/detail.cfm?id=1563874

Key Features & Benefits (2)

High Scalability

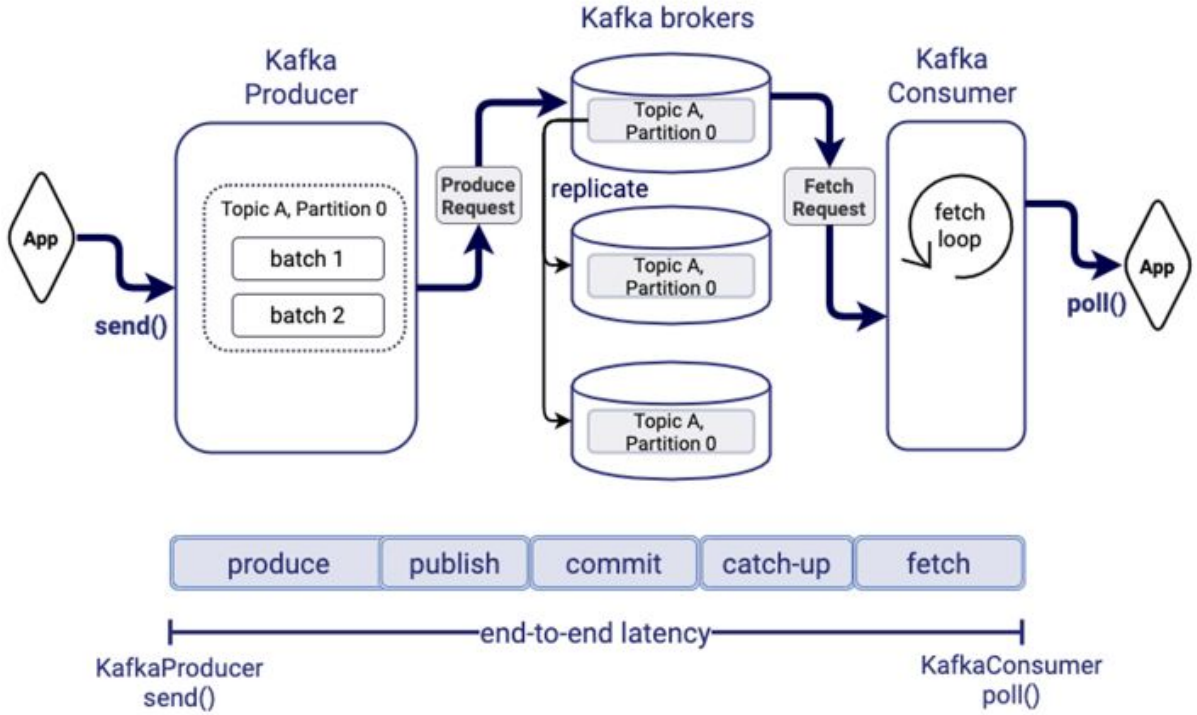
- Scale Kafka clusters up to a thousand brokers.
- Handle trillions of messages per day.
- Manage petabytes of data.
- Support hundreds of thousands of partitions.
- Elastically expand and contract storage and processing.



Key Features & Benefits (3)

Low Latency

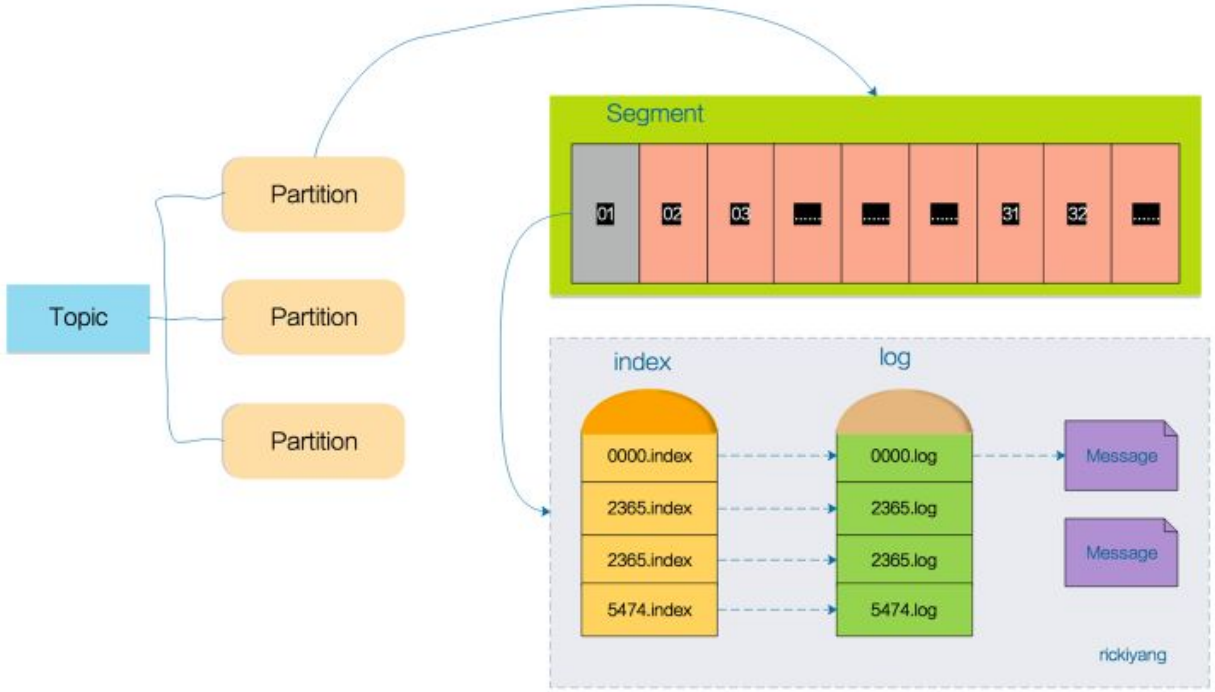
- Deliver high volume of messages.
- Utilize a cluster of machines.
- Achieve low latencies as low as 2ms.



Key Features & Benefits (4)

Permanent Storage

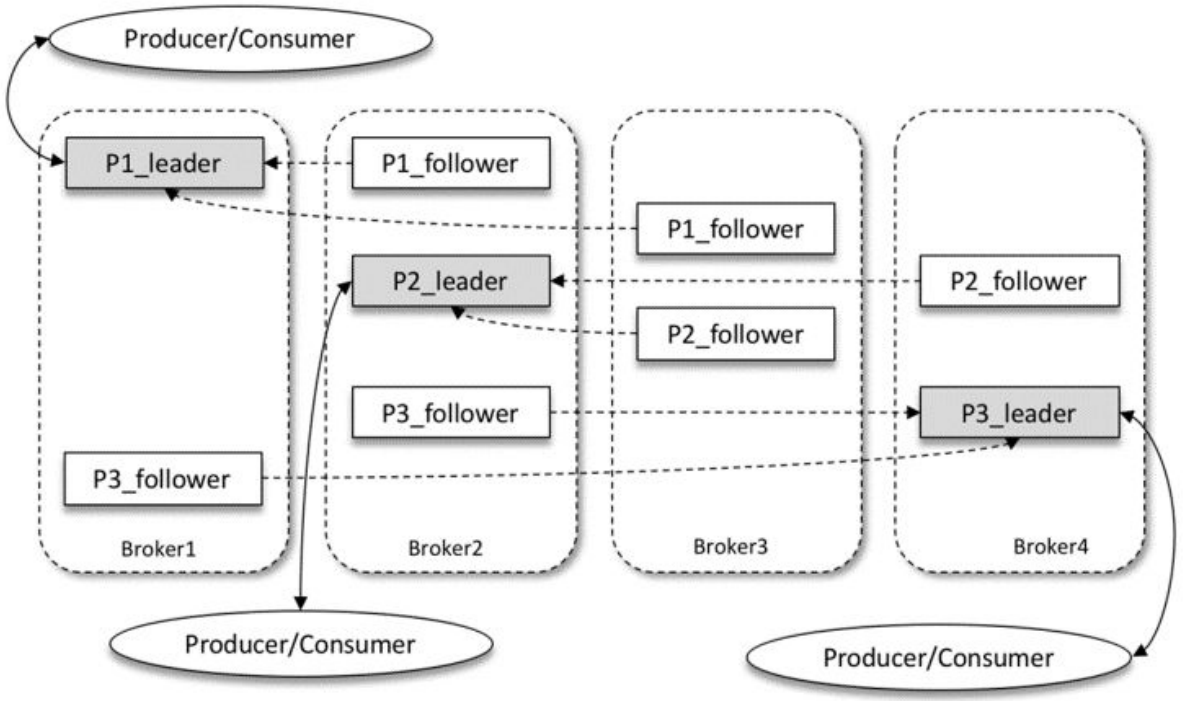
- Safely and securely store streams of data.
- Use a distributed cluster for storage.
- Ensure durability and reliability of data.
- Provide fault tolerance in the cluster.



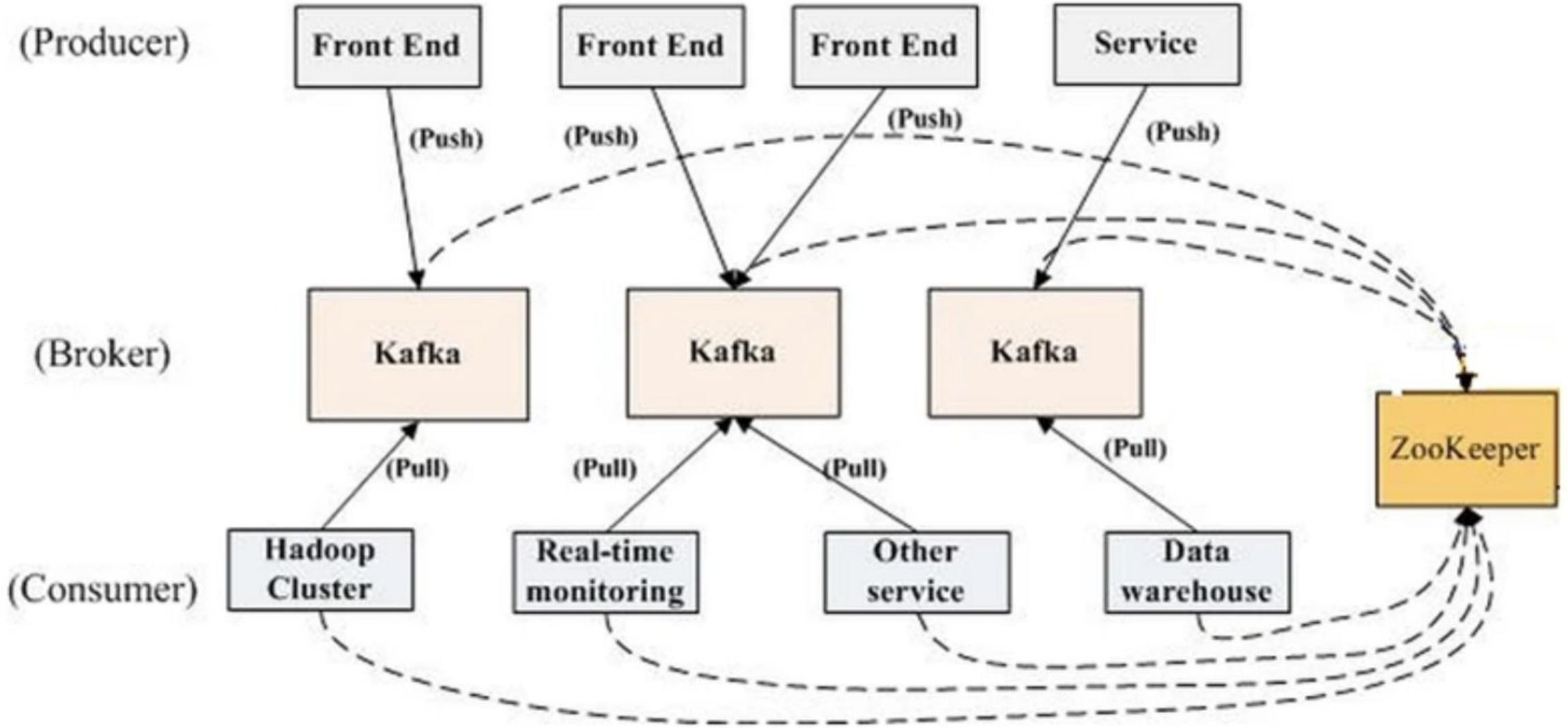
Key Features & Benefits (5)

High Availability

- Efficiently extend clusters over availability zones.
- Connect clusters across geographic regions.
- Ensure high availability and fault tolerance.
- Eliminate the risk of data loss.



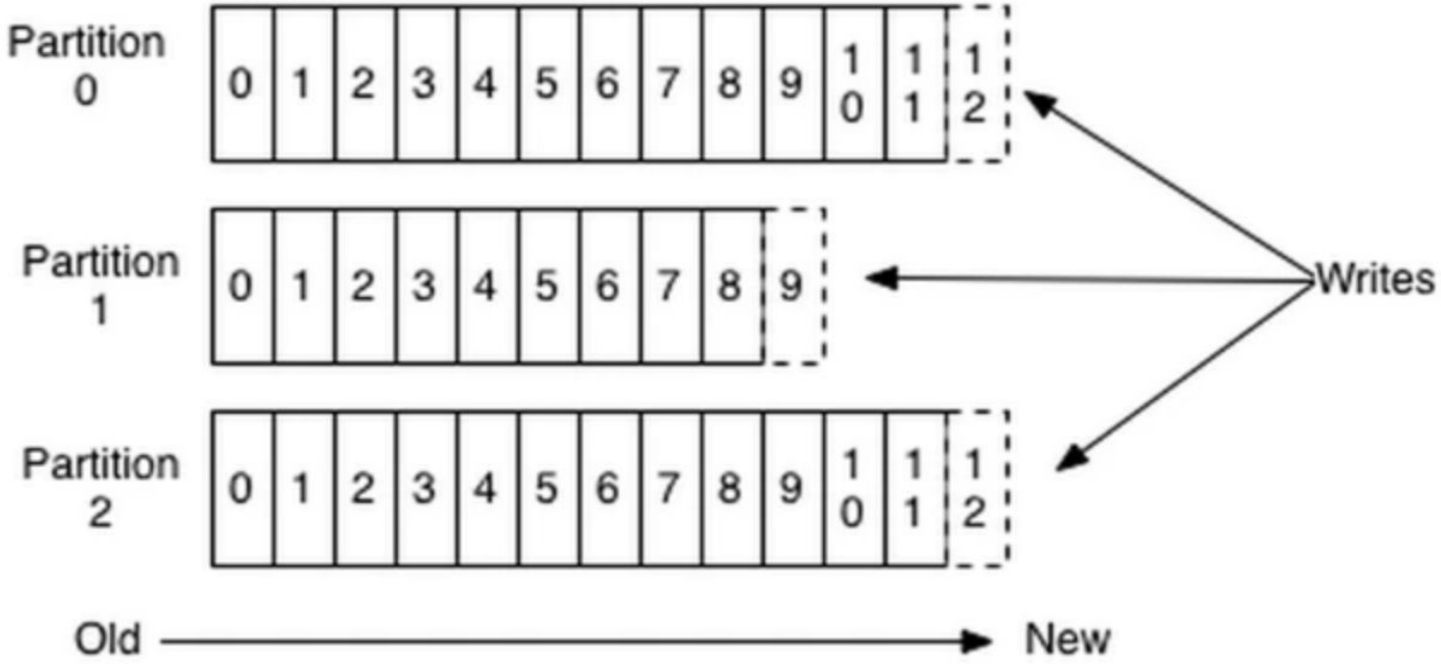
How Does It Work (1)



How Does It Work (2)

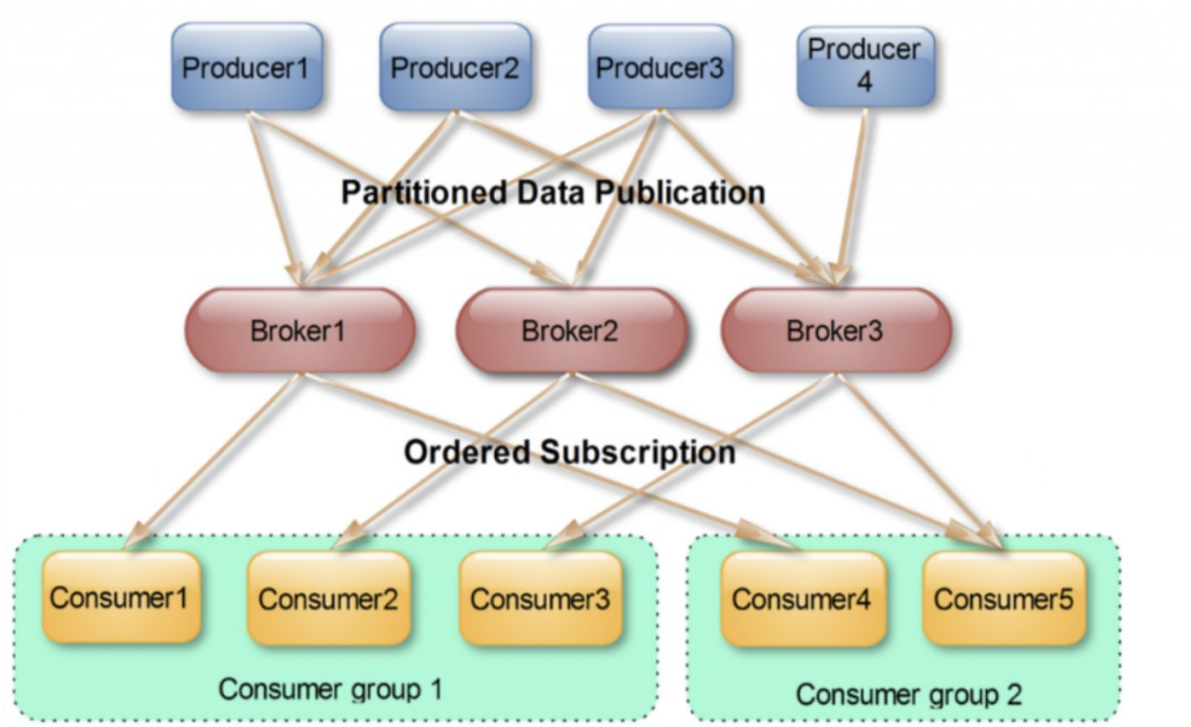
Because each message is appended to the partition, which is sequential disk writing, the efficiency is extremely high. It has been verified that sequential disk writing is even more efficient than random memory writing, which is a crucial factor in ensuring Kafka's high throughput.

For traditional message queues, it is common practice to delete messages that have been consumed. However, a Kafka cluster retains all messages.



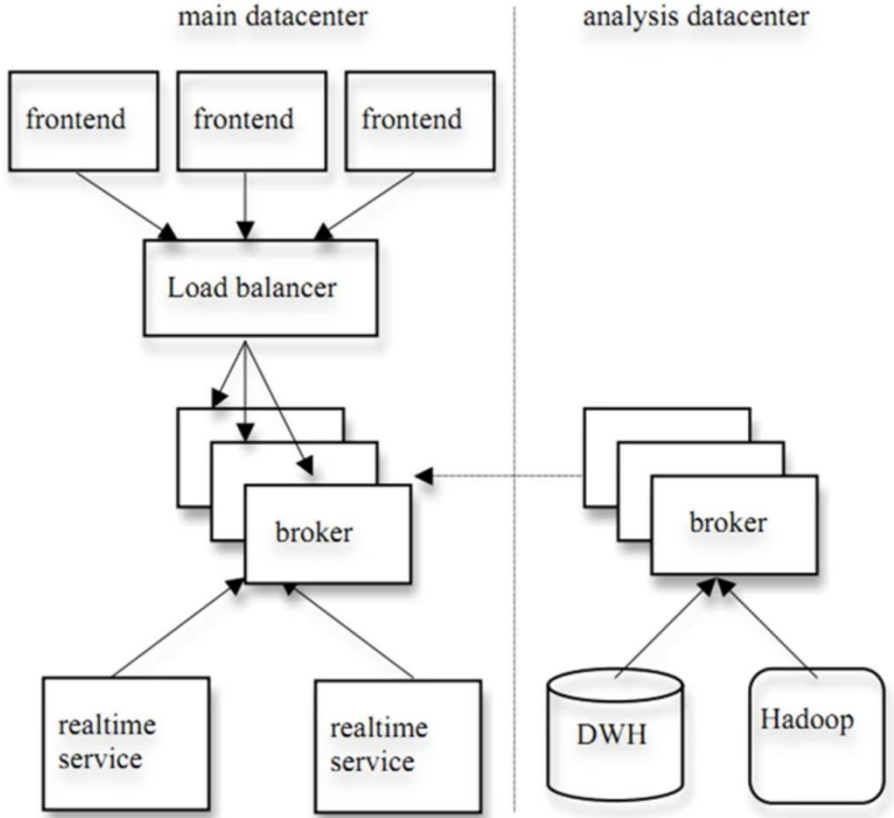
How Does It Work (3)

This is Kafka's means of implementing broadcasting and unicasting of topic messages. A topic can correspond to multiple consumer groups. To achieve broadcasting, each consumer needs to have an independent group. To achieve unicasting, all consumers should be in the same group. By using consumer groups, consumers can be freely grouped without the need to send messages multiple times to different topics.



How Does It Work (4)

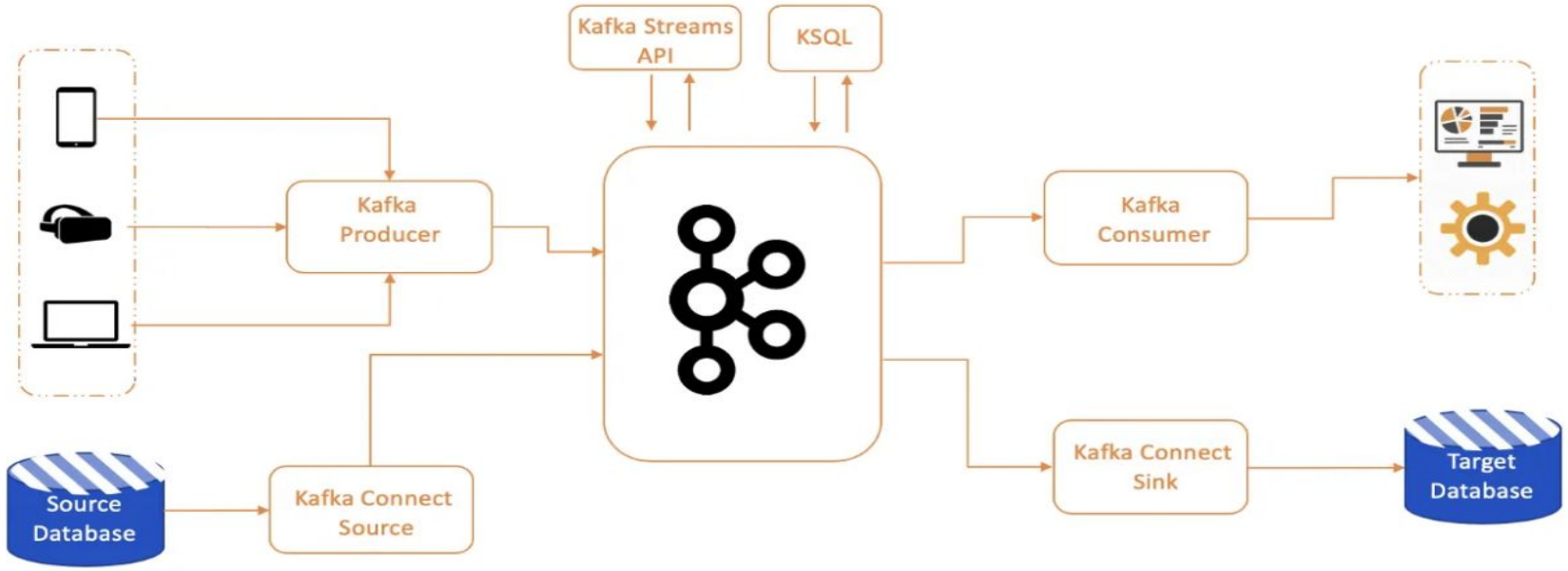
One of Kafka's design principles is to provide both offline processing and real-time processing simultaneously. Based on this feature, one can use a real-time streaming processing system like Storm for real-time online processing of messages, while using a batch processing system like Hadoop for offline processing. It is also possible to simultaneously replicate data in real-time to another data center.



How Does It Work (5)

Kafka EcoSystem

Kafka is a horizontally scalable system that can run on a cluster of nodes across multiple regions, enabling each broker to handle messages in the terabyte range without compromising performance. Once the Kafka cluster is set up, external applications can interact with the cluster using the Kafka API to publish, transform, or consume messages.



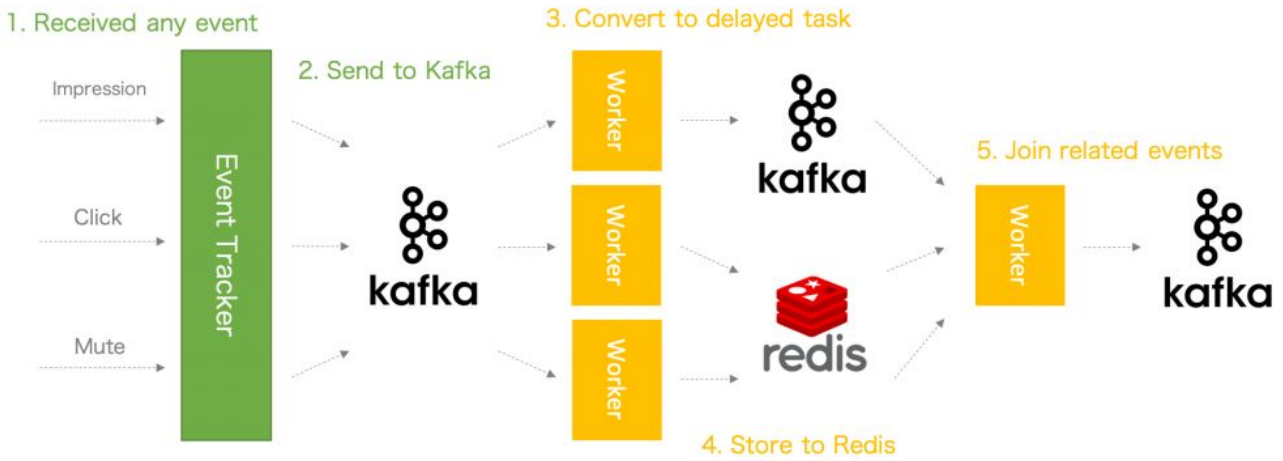
How Does It Work (6)

Kafka API

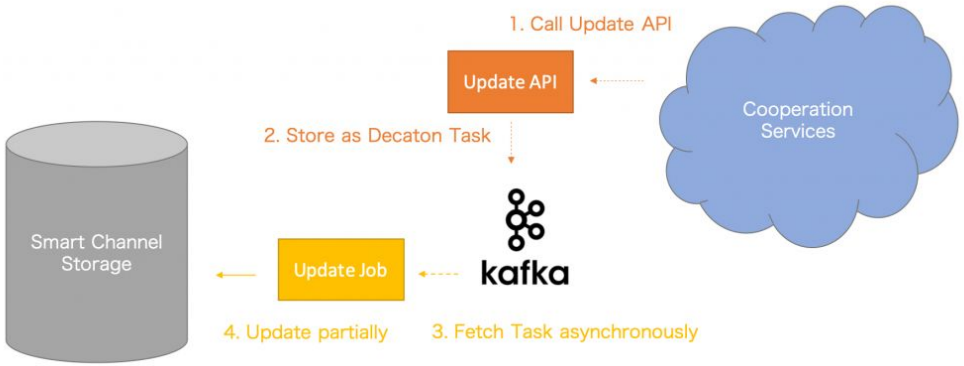
Under Kafka EcoSystem, it's important to understand different APIs for consuming data and corresponding use cases. Some of the most popular Kafka use cases in Industry are:

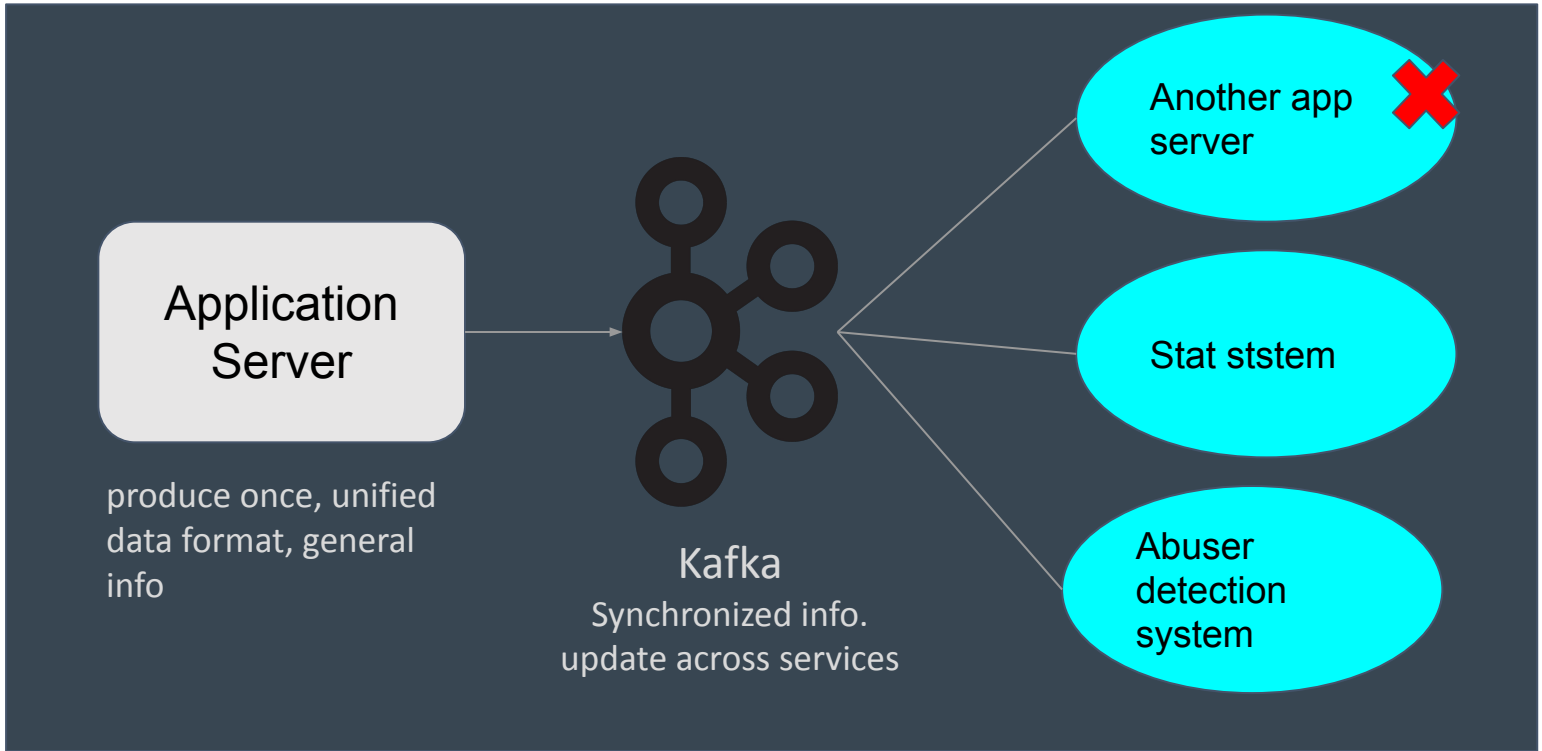
- Building scalable and reliable messaging platforms which deliver messages at high throughput.
- Decoupling system dependencies.
- Build real-time event processing systems with machine learning, AI-based processing.
- Build distributed data processing pipeline to handle ETL tasks.
- Using as a storage system.

- More than **60 services** use Kafka
- 140+ billions messages /day
- 38+ TB incoming data /day
- 3.5+ million messages /sec on peak



- Multi-services
- Big streaming data
- Real-time Stat.

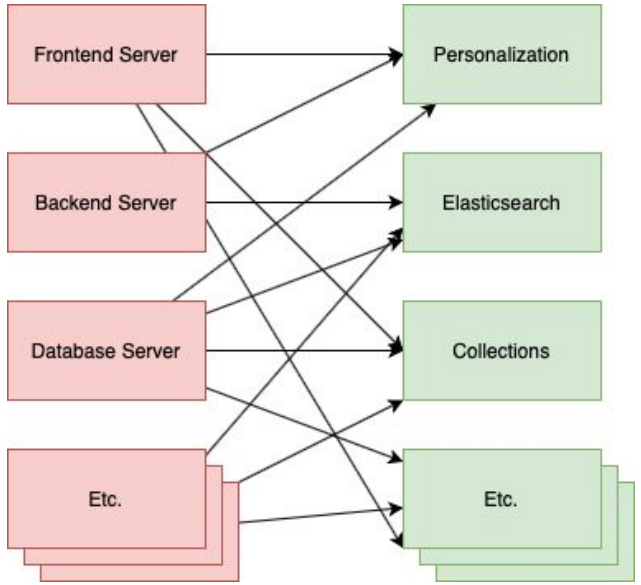




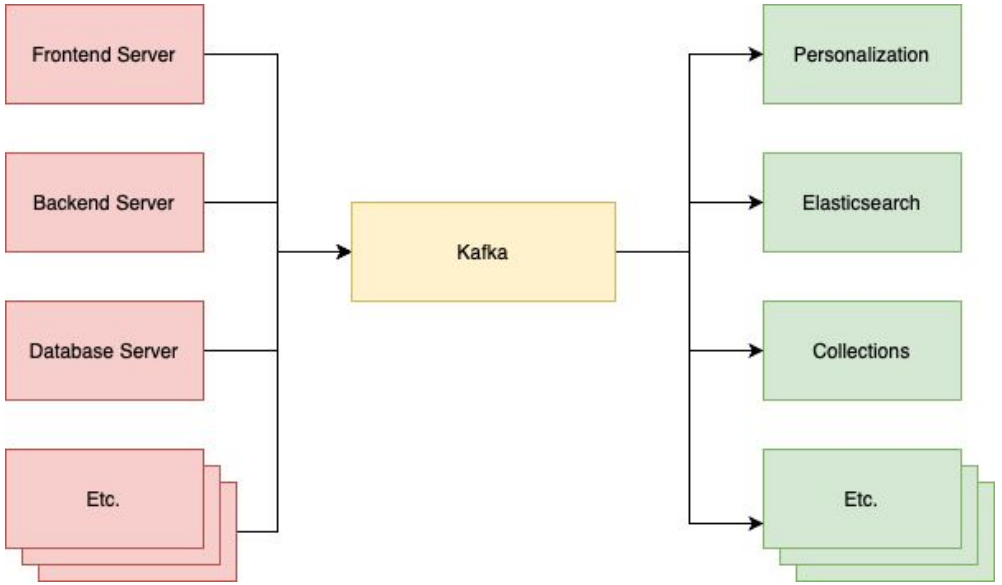
Application Case Study Line with Kafka

Application Case Study (1)

- Apache Kafka was developed by *Linkedin* in 2010.
- To solve **data pipeline** problem.
- Designed for big data streaming processing.
- Capable of handling tens of thousands of requests per second



↑**Before** Adopting Kafka

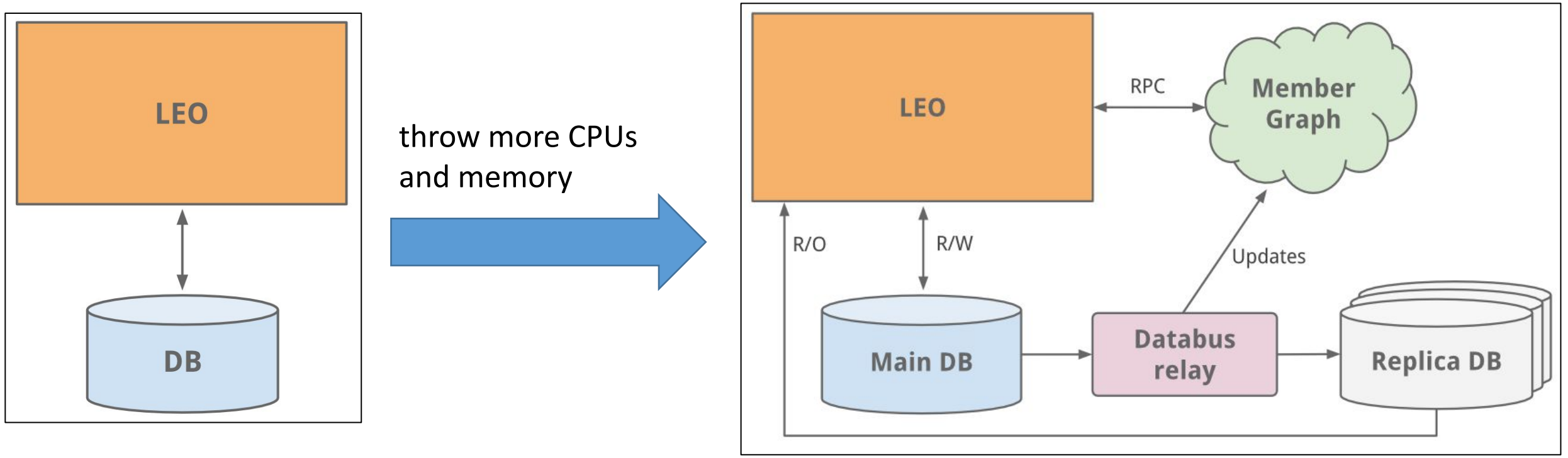


↑**After** Adopting Kafka

Application Case Study (2) - LinkedIn

The early years-LEO

- As a single monolithic application doing it all.
- Hosted web servlets for all the various pages, handled business logic, and connected to a handful of LinkedIn databases.



Application Case Study (3) - LinkedIn

However, as the site began to see more and more traffic

- Leo was often going down in production,
- Difficult to troubleshoot and recover,
- Difficult to release new code
- **High Availability** is critical to LinkedIn.

Application Case Study (4) - LinkedIn

- Demand:

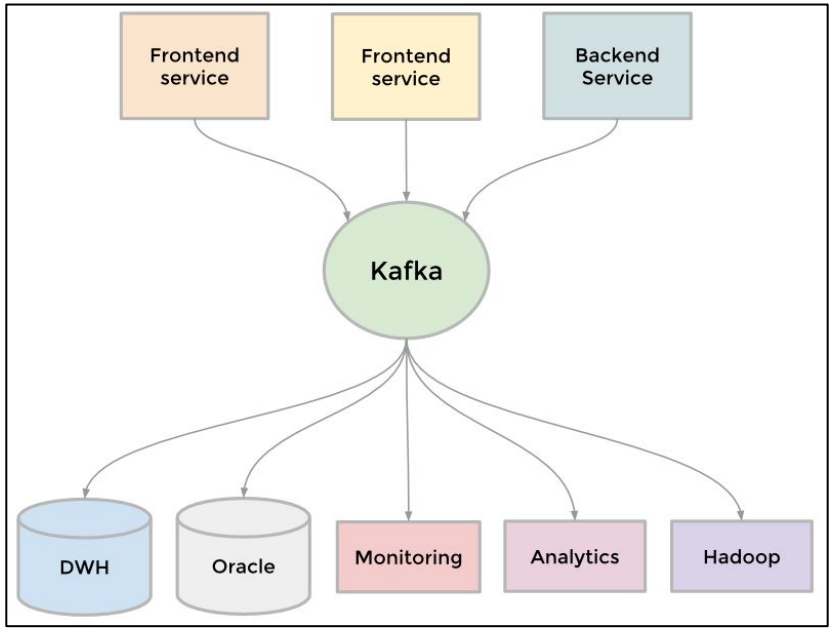
 Sending batches of data into our Hadoop workflow for analytics

 Collecting and aggregating logs from every service

 Collecting tracking events like pageviews

 Keeping search system up to date whenever profile is updated.

- To collect the growing amount of data
  → Result: the development of **Kafka**



Application Case Study (5) - LinkedIn

- Characteristic of Kafka:

 Built around the concept of a commit log with **speed** and **scalability**.

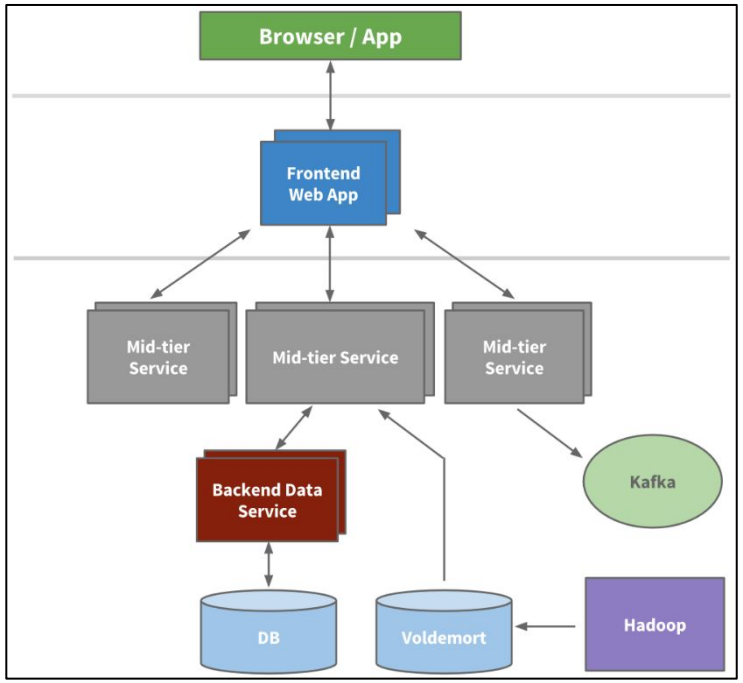
 Enabled near **real-time** access to any data source.

 Empowered our Hadoop jobs.

 Allowed to build **real-time analytics**.

 Improved site monitoring and alerting capability.

→ Kafka handles well over 500 billion events per day.



Application Case Study (6) - LinkedIn

Utilization of Kafka at LinkedIn

- **Monitoring**
  All hosts at LinkedIn emit metrics through Kafka.
  Then collected and processed to create monitoring dashboards and alerts.
- **Messaging**
  Various applications leverage Kafka as a traditional messaging system
  Application Including: Search, Content Feed and Relevance.
- **Analytics**
  Information are sent into a Kafka cluster in each data center.
  Events are collected and pushed onto Hadoop for analysis and daily report.
- As a building block (log) in various distributed applications/platforms
  Kafka is also leveraged as a core building block by other products like Pinot.

Study Comments (20%)

- Your abstract/evaluation?
  - Kafka is an efficient handling of **real-time data streams** with **scalable**, **fault-tolerant**, and **high-throughput** messaging system
- Any impact to you from your study and understanding?
  - It has provided me how social applications like Line, Facebook is worked.
- Any comparison and alternative better technology?
  - Apache Pulsar, RabbitMQ, and Apache ActiveMQ.
- Any questions? How (methodology) to find the answers by yourself?
  - We develop the methodology from application which used Kafka . e.g. Line

Reference(s) (5%)

- Title: Apache Kafka 介紹
  Author: Chi-Hsuan Huang
  https://medium.com/@chihsuan/introduction-to-apache-kafka-1cae693aa85e
  Published Date: Apr 12, 2020
- Title: A Brief History of Scaling LinkedIn
  Author: Josh Clemm
  https://engineering.linkedin.com/architecture/brief-history-scaling-linkedin
  Published Date: July 20, 2015
- Title: Kafka at LinkedIn: Current and Future
  Author: Mammad Zadeh
  https://engineering.linkedin.com/kafka/kafka-linkedin-current-and-future
  Published Date: January 29, 2015
- Title: Kafka's origin story at LinkedIn
  Author: Tanvir Ahmed
  https://www.linkedin.com/pulse/kafkas-origin-story-linkedin-tanvir-ahmed
  Published Date: Sep 20, 2019
- Title: 細說 Kafka Partition 分區
  Author: iriniland
  細說 Kafka Partition 分區 - 閱坊 (readfog.com)
  Published Date: May 7, 2021
- Title:【 Broker 】Apache Kafka 簡介
  Author:Archer
  【 Broker 】Apache Kafka 簡介 (learningsky.io)
  Published Date: Aug 29, 2019

Reference(s) (5%)

- Title: kafka副本机制
  Author: Edgar
  https://edgar615.github.io/kafka-replicas.html
  Published Date: Feb 05, 2020
- Title: 直观理解：Kafka零拷贝技术（Zero-Copy）
  Author: 老羊_肖恩
  https://www.jianshu.com/p/0af1b4f1e164
  Published Date: Nov 30, 2021
- Title: Kafka消息存储机制
  Author: 星河之码
  https://www.modb.pro/db/126424
  Published Date: Oct 04, 2021
- Title:Low Latency Data Streaming with Apache Kafka and Cloud-Native 5G Infrastructure
  Author: Kai Waehner
  https://www.kai-waehner.de/blog/2021/05/23/apache-kafka-cloud-native-telco-infrastructure-low-latency-data-streaming-5g-aws-wavelength/
  Published Date: May 23, 2021
- Title:Using Apache Kafka as a Scalable, Event-Driven Backbone for Service Architectures
  Author: Ben Stopford
  https://www.confluent.io/blog/apache-kafka-for-service-architectures/
  Published Date: Jul 19, 2017
- Title:Kafka高可用，高吞吐量低延迟的高并发的特性背后实现机制
  Author: 明智谈生活
  https://www.163.com/dy/article/HL40DHP20552ZN8B.html
  Published Date: Nov 17, 2022
- Title:『卡夫卡的藏書閣』- 程序猿必須懂的Kafka開發與實作系列
  Author: daniel_ho
  https://ithelp.ithome.com.tw/articles/10264758
  Published Date: Sep 15, 2021

Reference(s) (5%)

- Title: Kafka-based job queue library 'Decaton' examples
  Author: Kazuki Matsushita
  Kafka-based job queue library 'Decaton' examples (linecorp.com)
  Published Date: Dec 10, 2020

TBD

- reference >> 不算
- how does it work >> 等排版
- key feature >> v
- 減少頁數(不能超過20)
- case study 業界青睞 產品發布簡報 客戶應用報告 業務簡報
- **Study Comments (20%)**