Bukalapak

Building Distributed Semantic Job Queue with Kafka

Software Architecture Conference SARCCOM
Jakarta, October 27th 2018

About Bukalapak
Short Overview

- One of the largest e-marketplace in Southeast Asia
- 2200+ total employees
- 1000+ tech talents
- 70+ tech squads

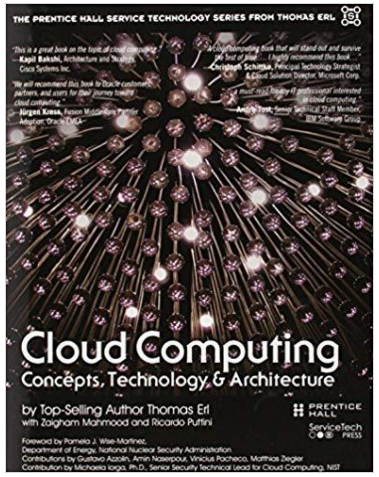
Speaker Profile

@masykurm

Masykur Marhendra Sukmanegara

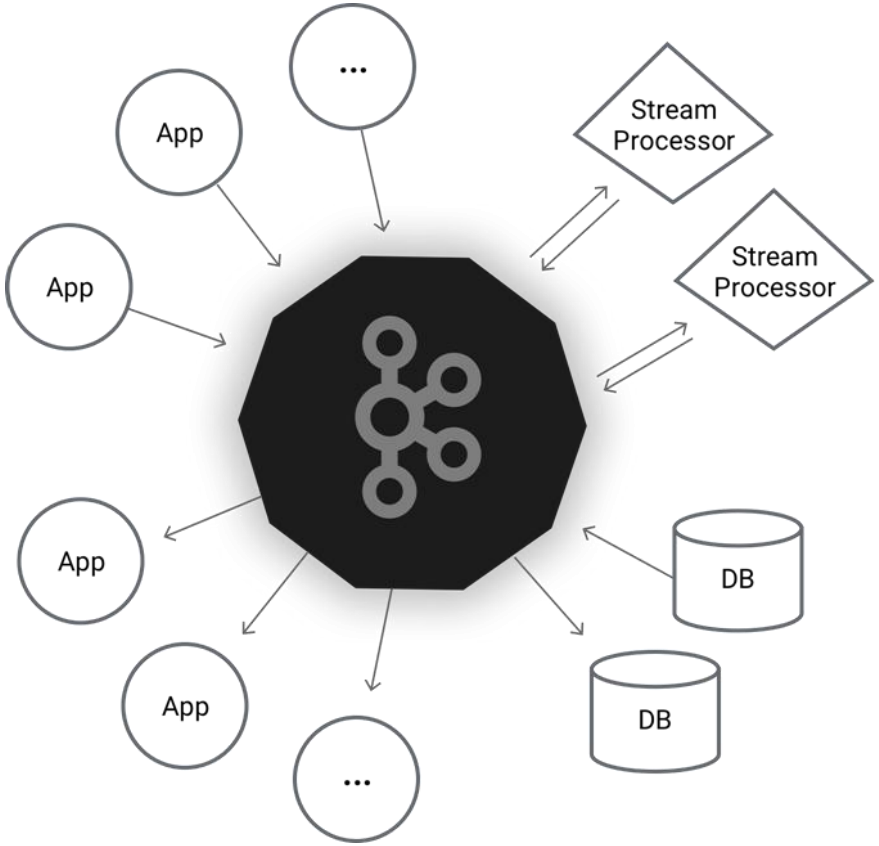
Software Architect - Bukalapak

- Years of experiences in middleware, integration, SOA, Microservices
- Mostly in telco, airlines, bank (a bit), and e-commerce (current)
- Now working on search relevancy improvement, architecture working group, microservices architecture, and more ...
- Prentice Hall Service Technology Books technical reviewers



Apache Kafka Overview

What is Apache Kafka ?



Apache Kafka® is ***a distributed streaming platform***

Run as a cluster on one or more servers that can span multiple DC

Stores streams of records in categories called topics. Each record stream consist of a key, a value, and a timestamp

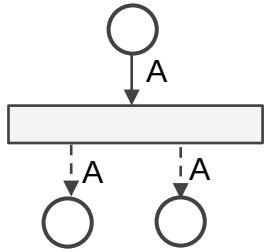
Apache Kafka Overview

Key Usage of Apache Kafka

AS MESSAGING SYSTEM

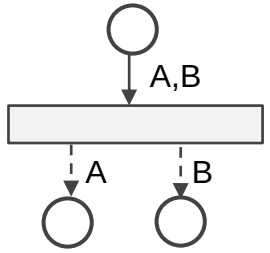
Publish Subscribe

Multiple consumers (subscribers) listen to same message published by a publisher



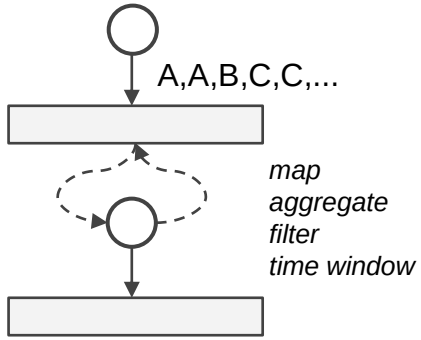
Queuing system

Multiple consumers (subscribers) receiving message alternately published by a publisher

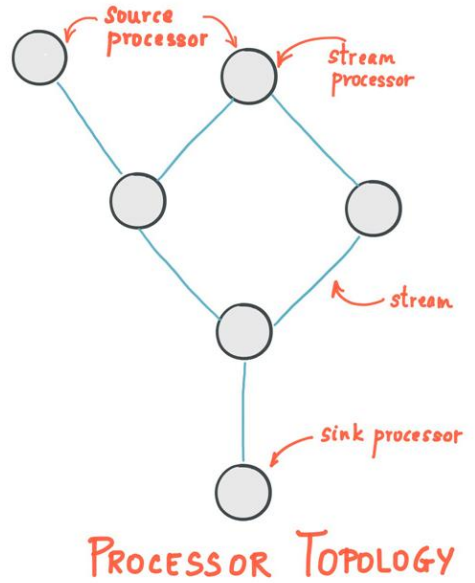


AS STREAM PROCESSING

Real time processing of continuous streams of data from input topics, performs some processing on this input, and produces continual streams of data to output topics

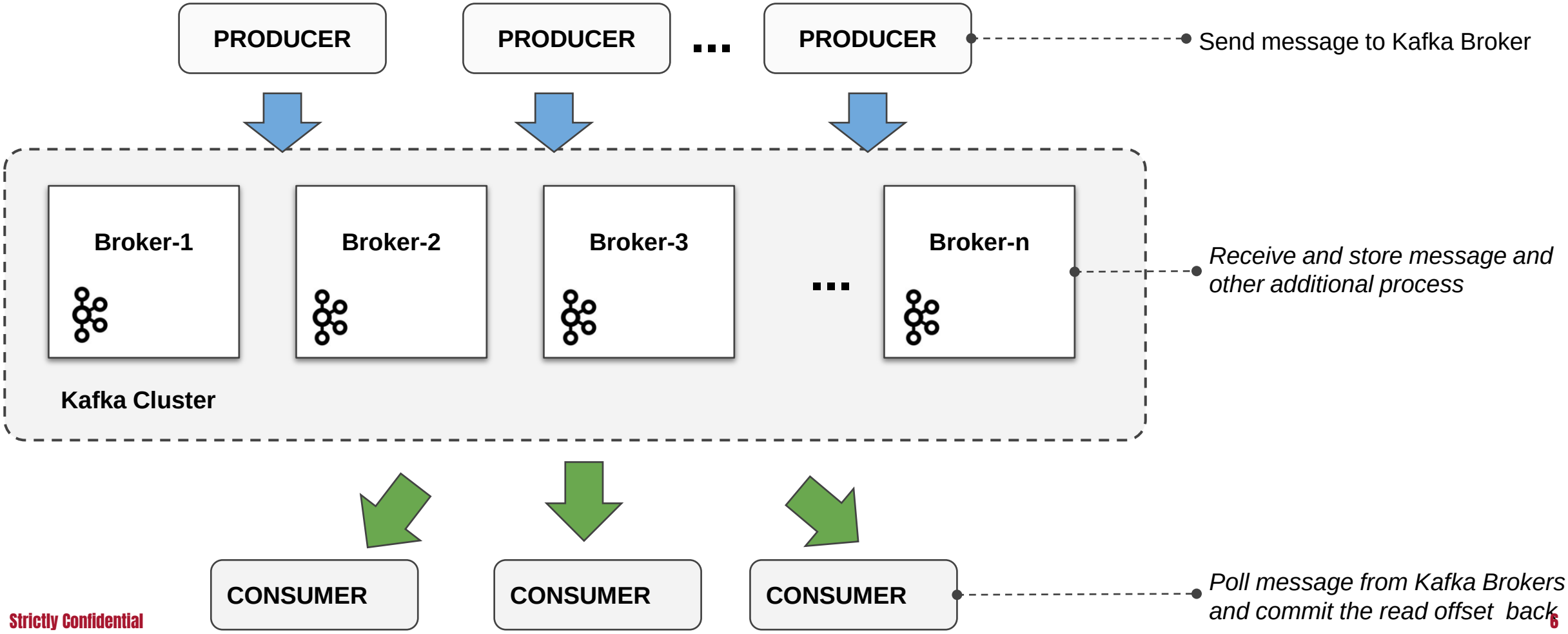


A,2,
B,1
C,2,
...



Apache Kafka Overview

Inside Apache Kafka : Producer, Brokers, Consumer (1/3)



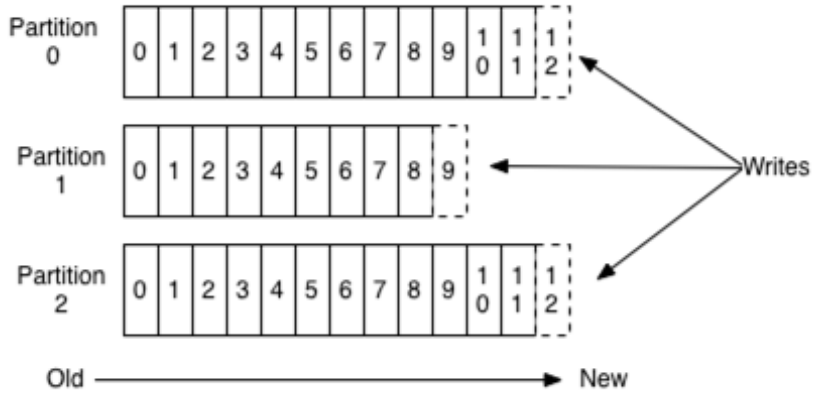
Apache Kafka Overview

Inside Apache Kafka : Topics, Read/Write Operation (2/3)

What is essentially performed inside Kafka Broker

Broker-1

Kafka Topics =
Partitioned Logs



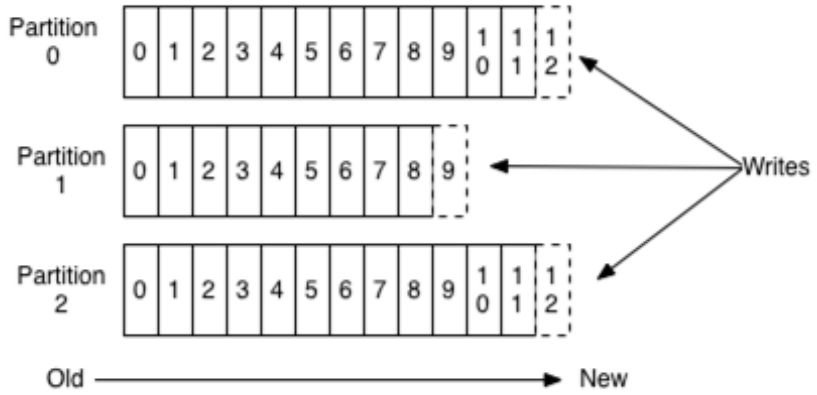
Apache Kafka Overview

Inside Apache Kafka : Topics, Read/Write Operation (2/3)

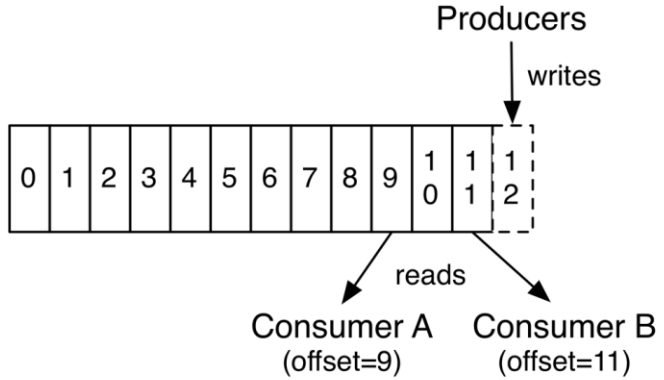
What is essentially performed inside Kafka Broker

Broker-1

Kafka Topics = Partitioned Logs



Read / Write Operation of Kafka Message



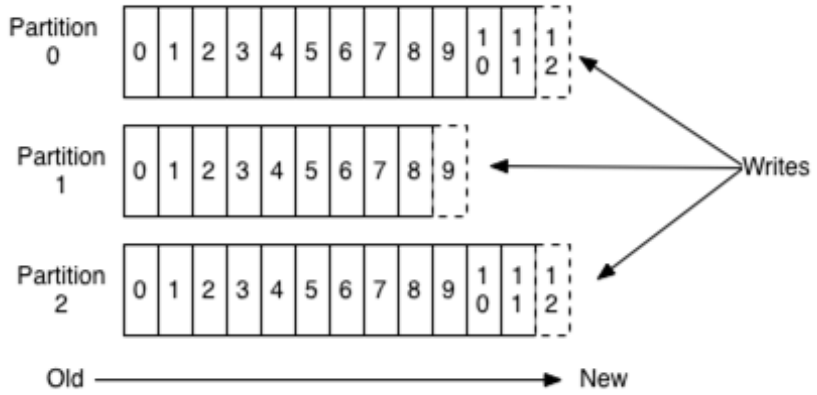
Apache Kafka Overview

Inside Apache Kafka : Topics, Partitions, Read/Write Operation (2/3)

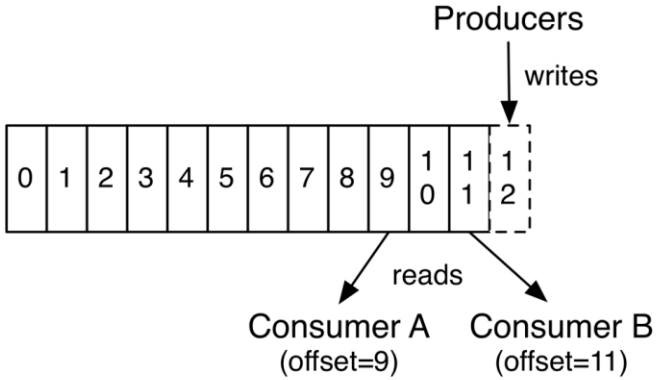
What is essentially performed inside Kafka Broker

Broker-1

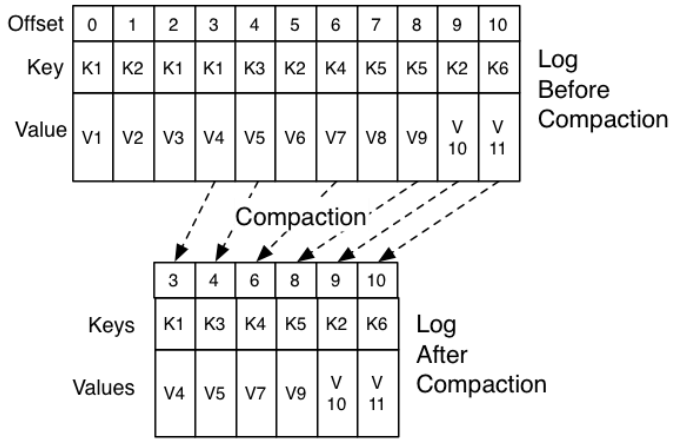
Kafka Topics = Partitioned Logs



Read / Write Operation of Kafka Message

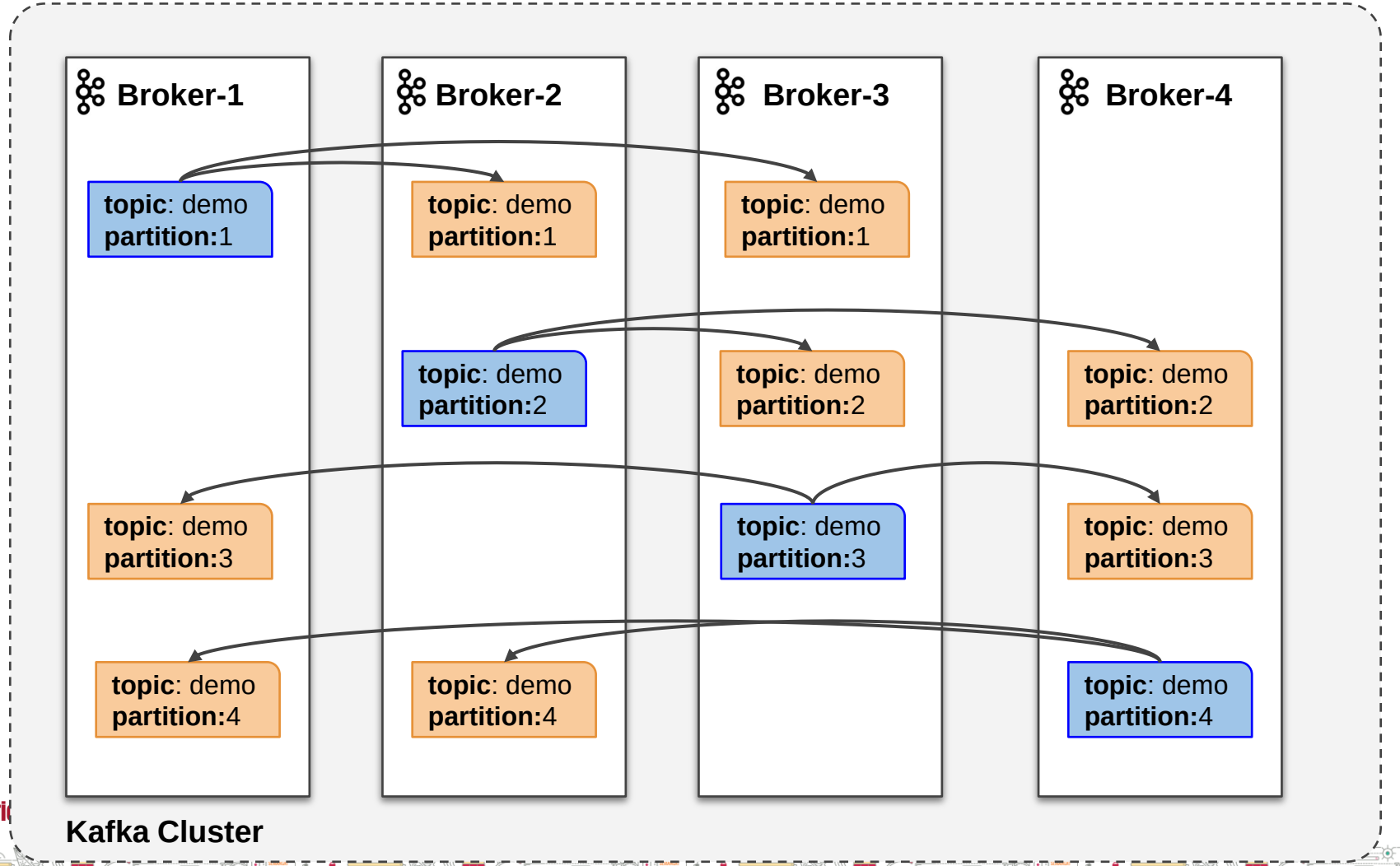


Partitioned Logs Compaction



Apache Kafka Overview

Inside Apache Kafka : Replication (3/3)



Replication of message in topic partition over the kafka cluster

Define partitions and replication factors with right sizing for optimal performances

partition multiply of # of brokers available in the cluster but don't oversize it

More partitions mean a greater parallelization and throughput but partitions also mean more replication latency, rebalances, and open server files

Leader

Followers

Microservices Overview

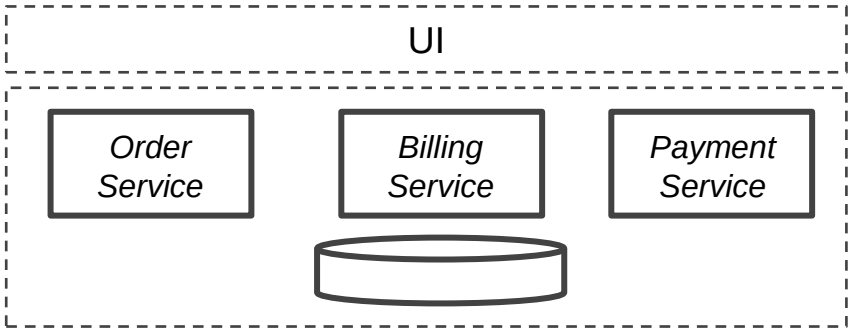
What is Microservices ?

microservices is ***an architectural style***

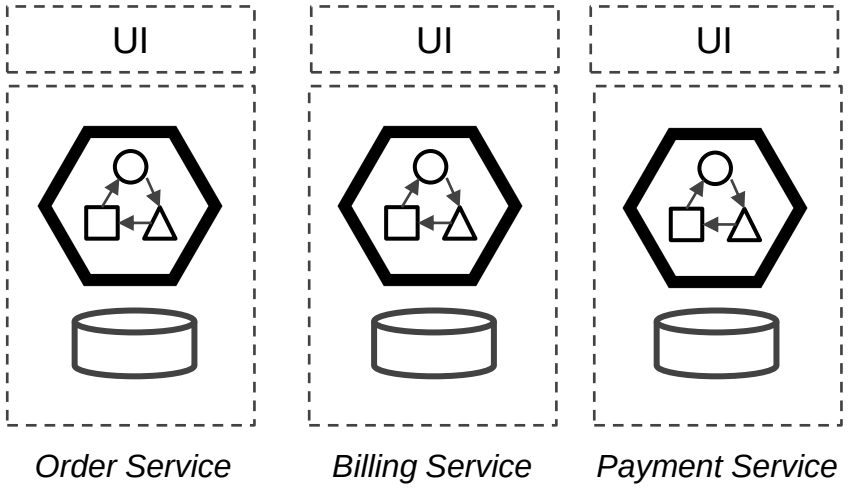
the combination of distinctive features in which architecture is performed or expressed:

Distinctive features of microservices:

- Single applications built as suite of small service
- Built around business capabilities
- Independently deployable
- Decentralized data management
- Low coupling and high cohesion as much as possible



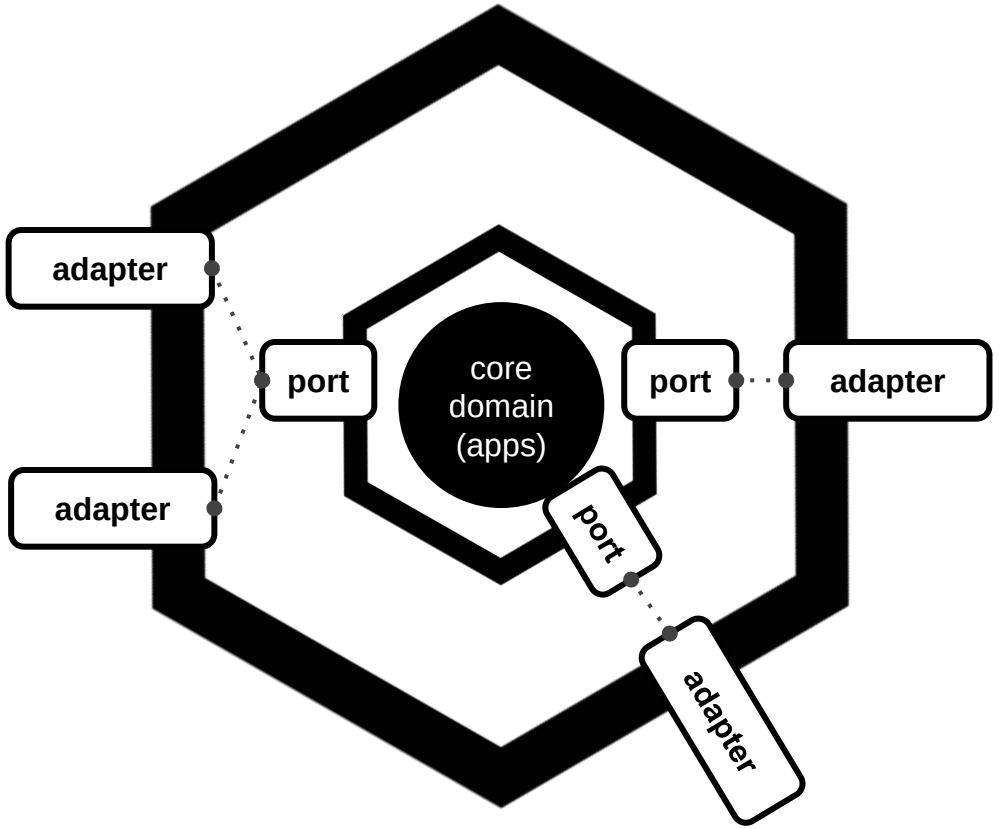
Monolithic architectural style



Microservices architectural style

Microservices Overview

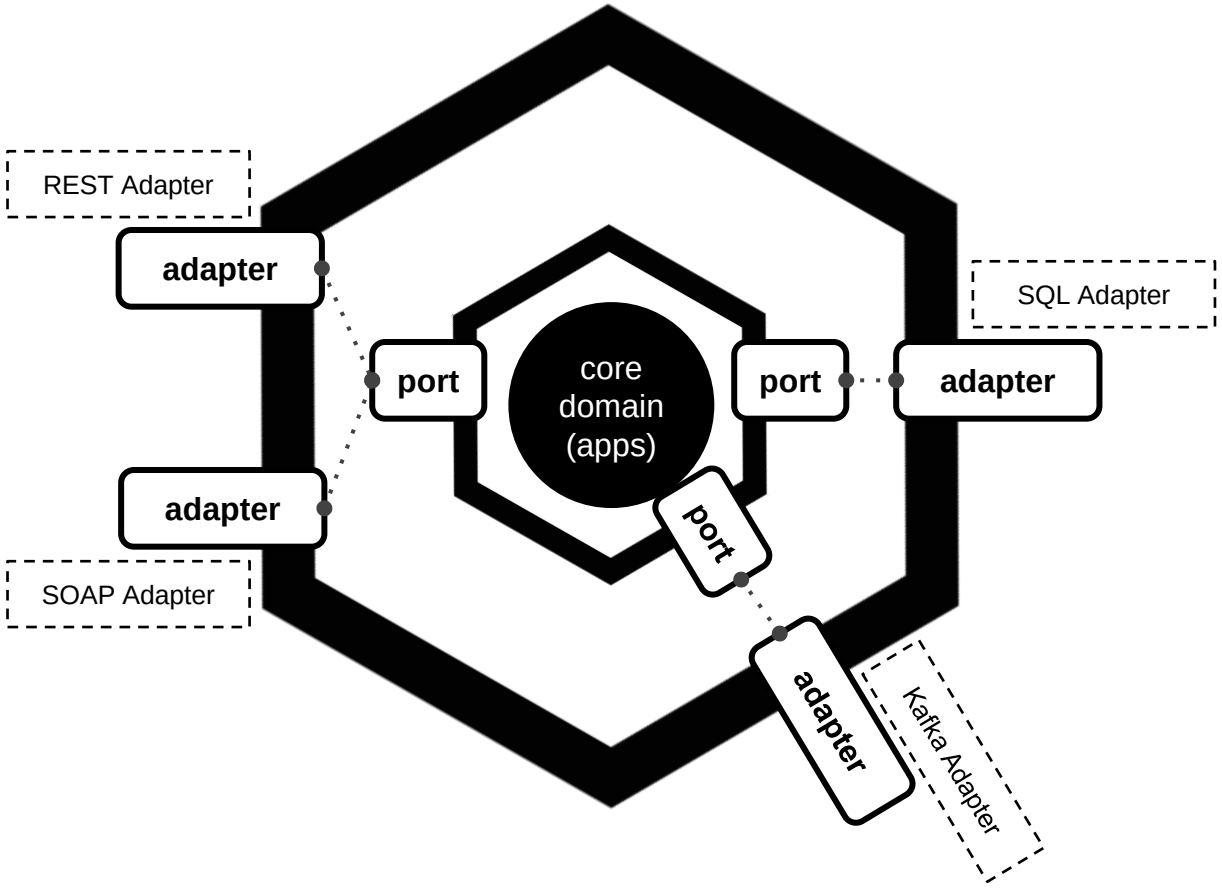
Quick View on Hexagonal Architecture on Microservices



- Decouple business logic (core domain) with the way to connect with outer world (external system) - agnostic to the outside world
- Also known as Port and Adapter pattern
- Ports are entry points of the business logic to the external world - decoupled with “what” are the external world
- Adapter are the method on how and what to connect with external world on both ways communication

Microservices Overview

Quick View on Hexagonal Architecture on Microservices



- Decouple business logic (core domain) with the way to connect with outer world (external system) - agnostic to the outside world
- Also known as Port and Adapter pattern
- Ports are entry points of the business logic to the external world - decoupled with “what” are the external world
- Adapter are the method on how and what to connect with external world on both ways communication

E-Commerce Overview

Understanding e-commerce definition

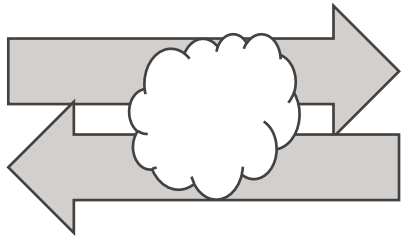
“activity of buying or selling of products on online services or over the Internet”

- Wikipedia

“buying and selling of goods and services, or the transmitting of funds or data, over an electronic network, primarily the internet”

- searchcio.techtarget.com

BUYERS



SELLERS

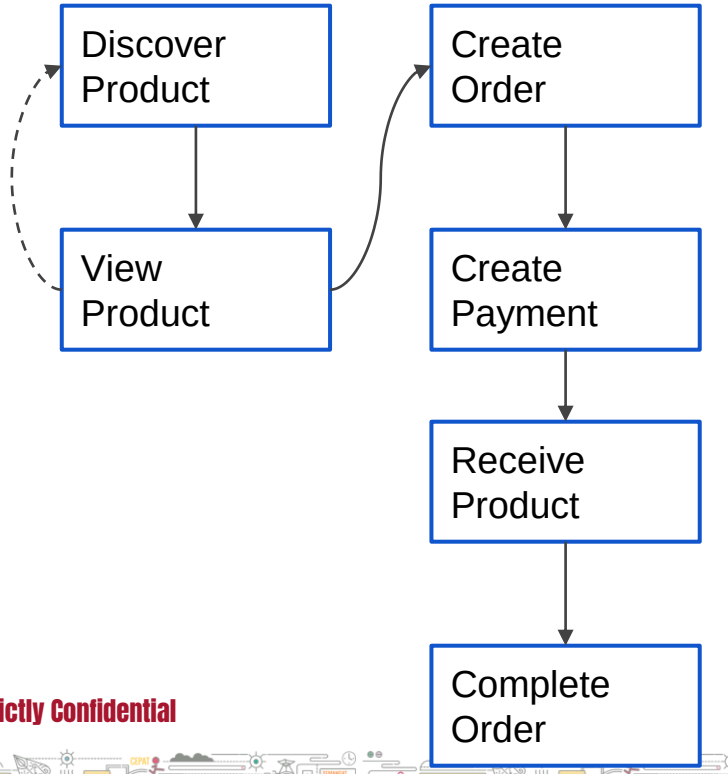
- **B2B** (Business to Business)
- **B2C** (Business to Consumer)
- **C2C** (Consumer to Consumer)

E-Commerce Overview

Common Process of an E-Commerce / Marketplace

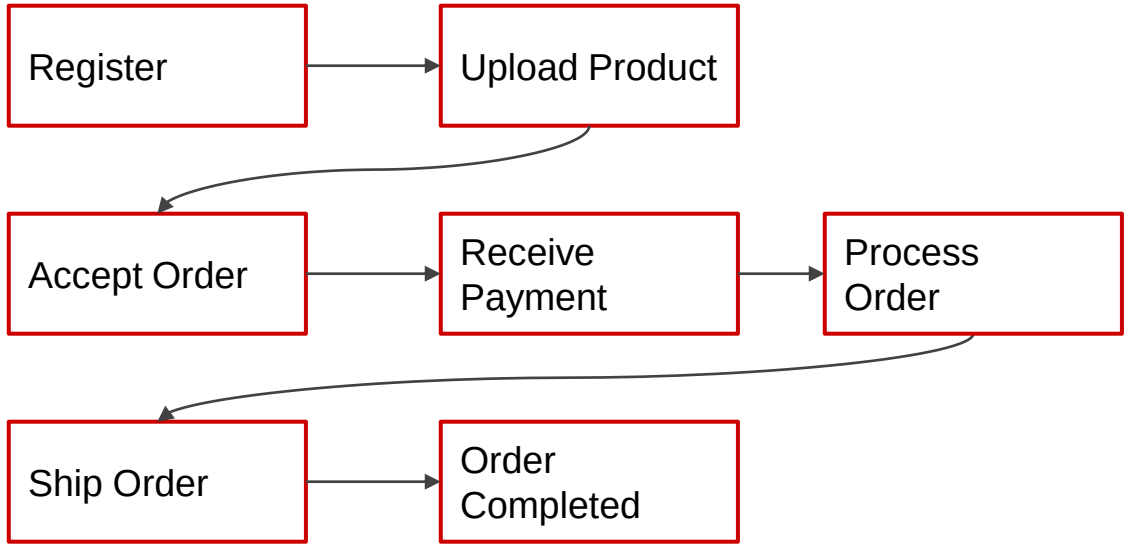
BUYERS

This actor use e-commerce as platform / media to help them find their daily needs



SELLERS

This actor use e-commerce as platform / media to help them sell their products via online for wider reach

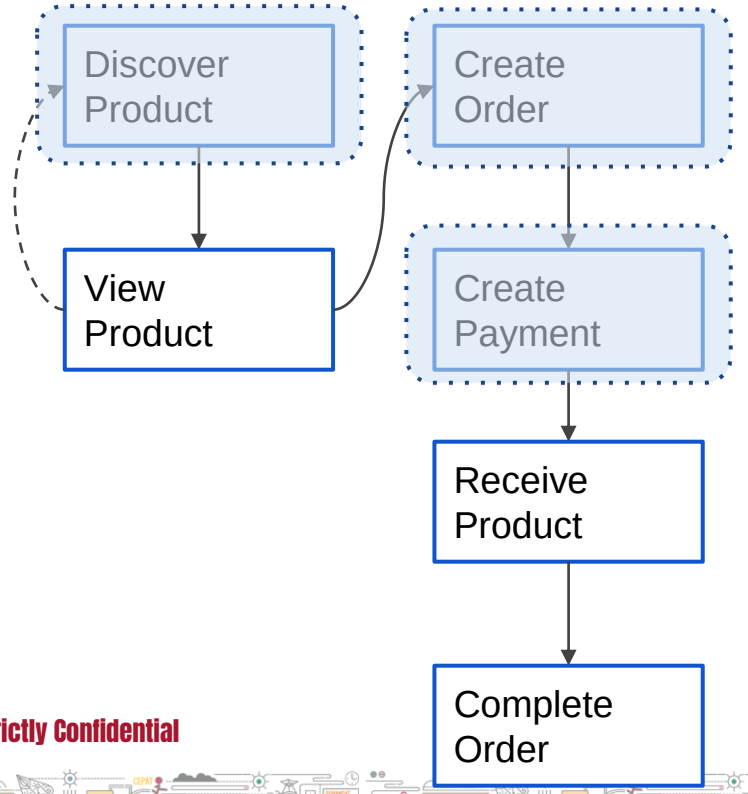


E-Commerce Overview

Common Process of an E-Commerce / Marketplace

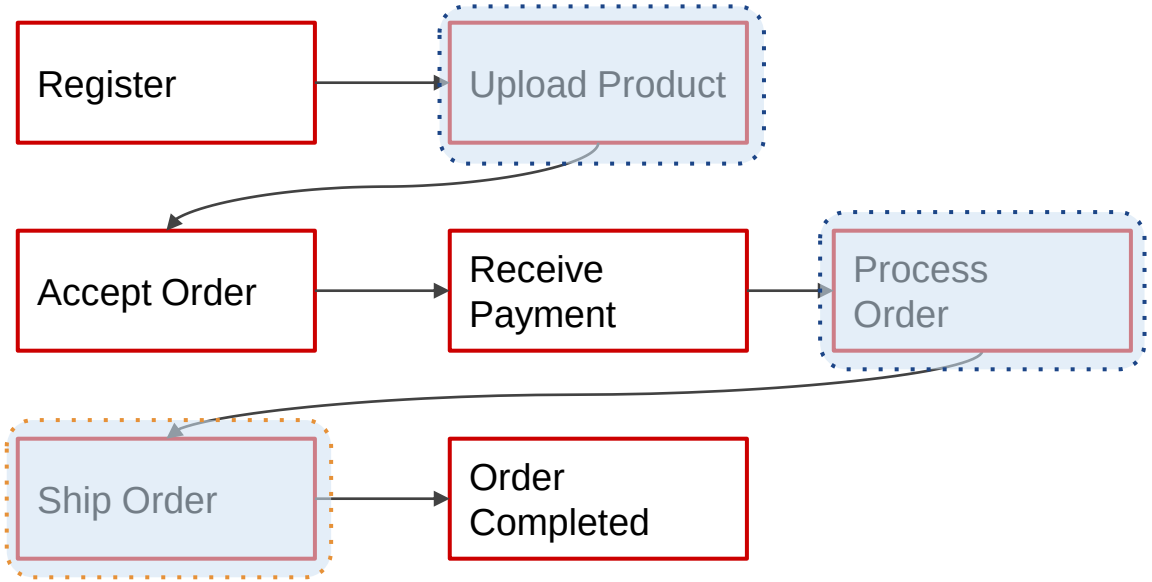
BUYERS

This actor use e-commerce as platform / media to help them find their daily needs



SELLERS

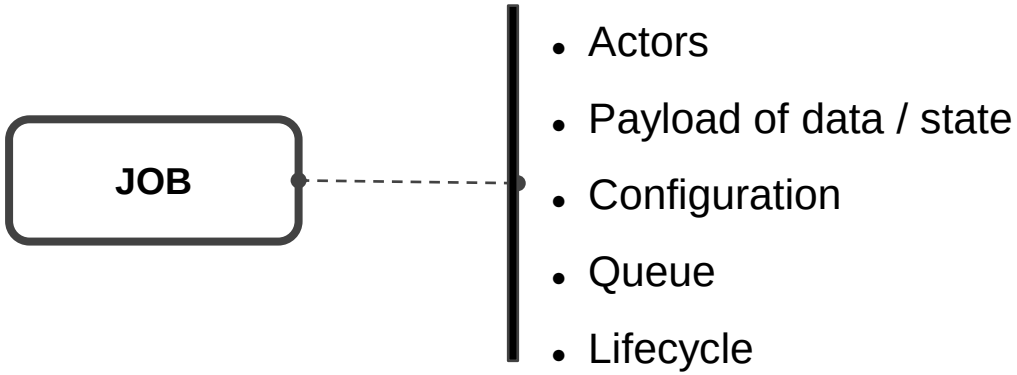
This actor use e-commerce as platform / media to help them sell their products via online for wider reach

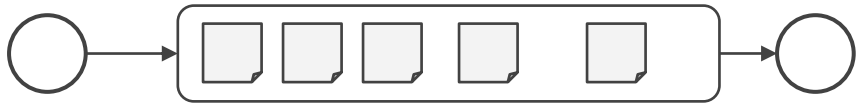


There are several process that might need to process data asynchronously as background job

Understanding of A Job (Background Job)

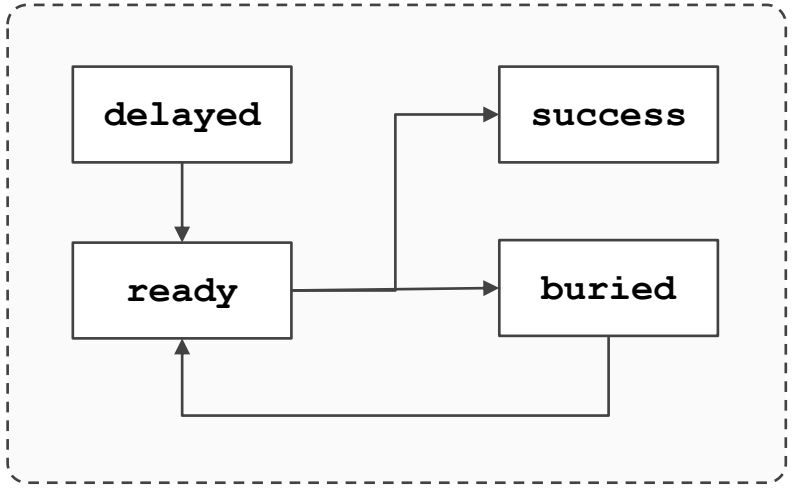
Overview of the job ecosystem





A job queue that store job definition submitted by the producer (actor) and to be executed by the consumer (actor)

HOW CAN WE BUILD SYSTEM WITH APACHE KAFKA AS JOB QUEUE THAT CAN ALSO SUPPORT JOB LIFECYCLE ?

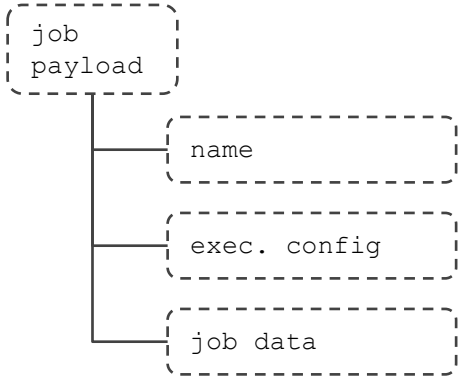


Simple job semantic illustrated as a lifecycle process

Modelling the Job Ecosystem into Workable System

Translating the job components into system components

PAYLOAD OF DATA



JSON over REST/HTTP

ACTORS

Proxy GW Service

Service that provide interface abstraction to the job queue and other actors

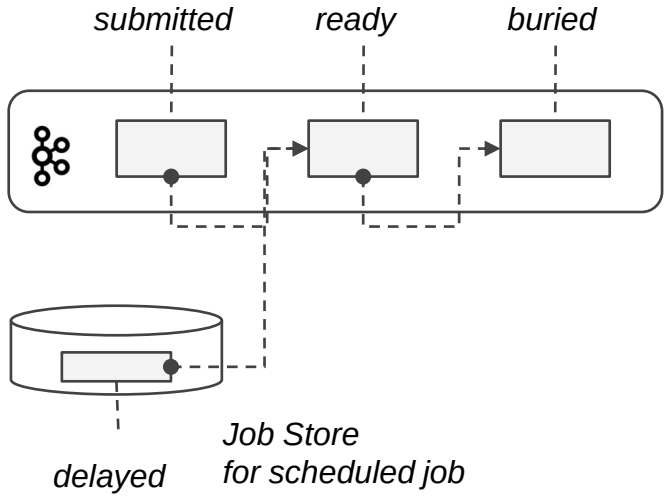
Scheduler Service

Service that holding role for scheduling the job that need delay on the execution

Executor Service

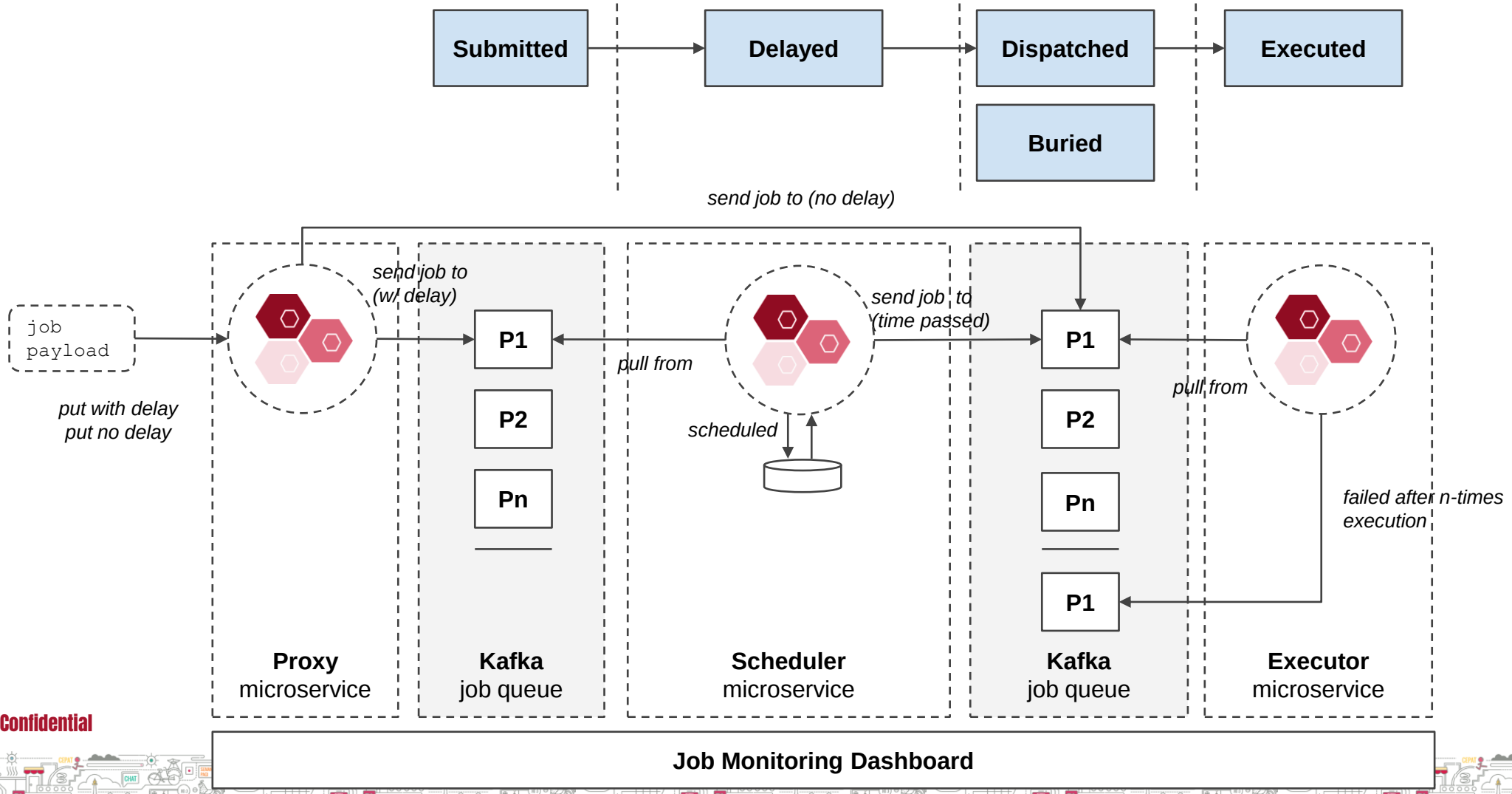
Service that perform job execution wrapped with executor library / framework

QUEUE & LIFECYCLE



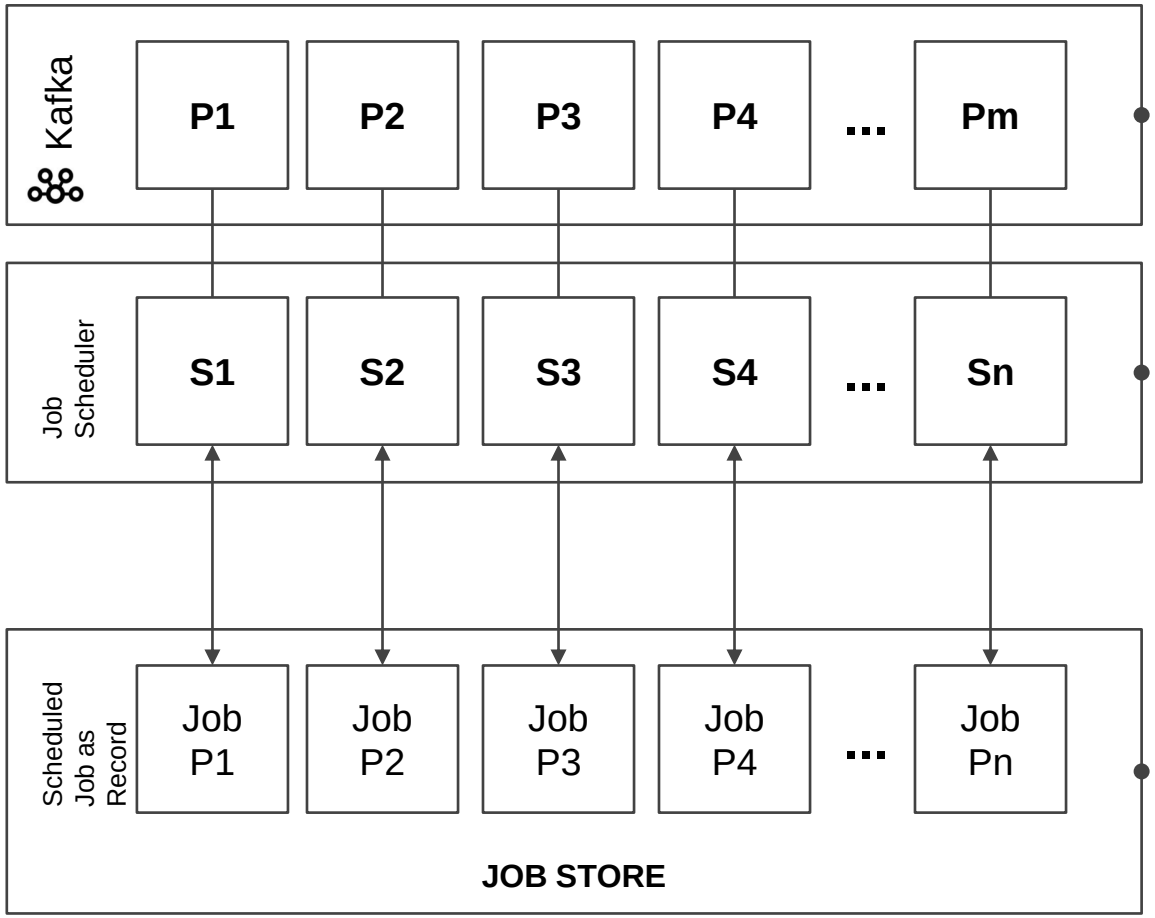
Modelling the Job Ecosystem into Workable System

Connecting it all together with process flow of a job with defined semantic



Addressing Scheduler Service concerns on concurrency

Each job will be stored along with the assigned partition in the Scheduler Service



Logical representation of queue for submitted job. Queue consists of configurable *m-partitions*

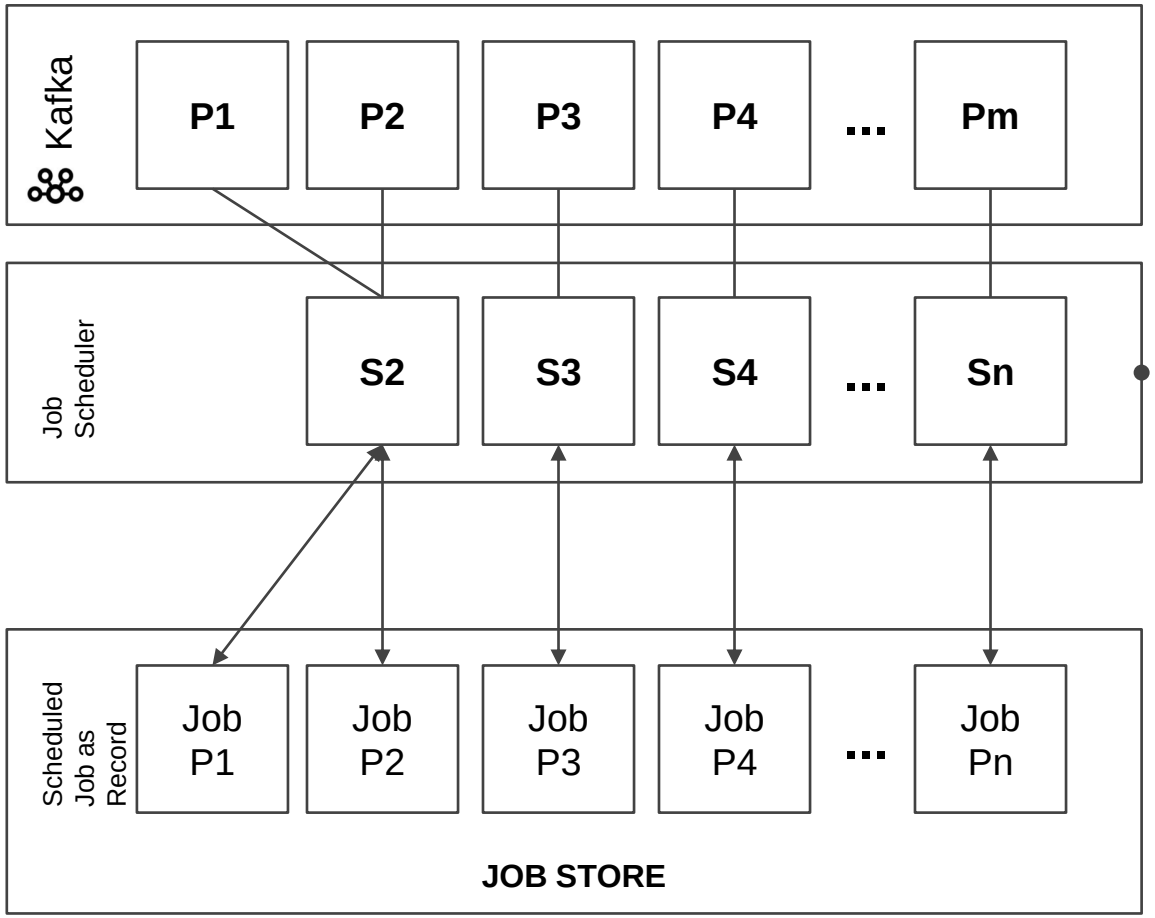
Let Job Scheduler run in *n-instances* in which # of n <= # of m. Each scheduler instance will be assigned with partition-x of the queue for submitted job

Each delayed job that received by instance-y of Job Scheduler, will be stored in Job Store with corresponding logical partition as Job Store record / document

This is to prevent Job Scheduler instance querying and executing same job record when it triggered concurrently

Addressing Scheduler Service concerns on concurrency

When 1 instance assigned more than 1 partition, it will store job to each assigned partition in round robin fashion



By the case when one instance of Job Scheduler unavailable, corresponding queue partition will be assigned to other Job Scheduler instance

Correspondingly, Job Scheduler will store the job to Job Store into logical partition based on all new partitions that has just assigned to it

Summary & Key Takeaways

- Kafka is powerful multi purpose messaging and streaming platform that can also be used as Job Queue. However in order to model full semantic of the job, we need to have external system to be plugged-in as complementary tools (e.g. scheduler, scheduled job)
- With the help of microservices architectural style, we can de-couple each of job's actors into separate isolated package service which then can be scaled and deployed autonomously
- Hexagonal pattern gives us flexibilities to plug-in / plug-out external system to our business logic without any change impact on the logic itself
- Concurrency is one the major key concerns that we must consider in the distributed computing world to minimize duplication of execution

software architecture

WE ARE HIRING ! :) - Check out careers.bukalapak.com

Thank You

BukaLapak