Kafka : Introduction

Event streaming, distribution et scaling infini... ou presque...

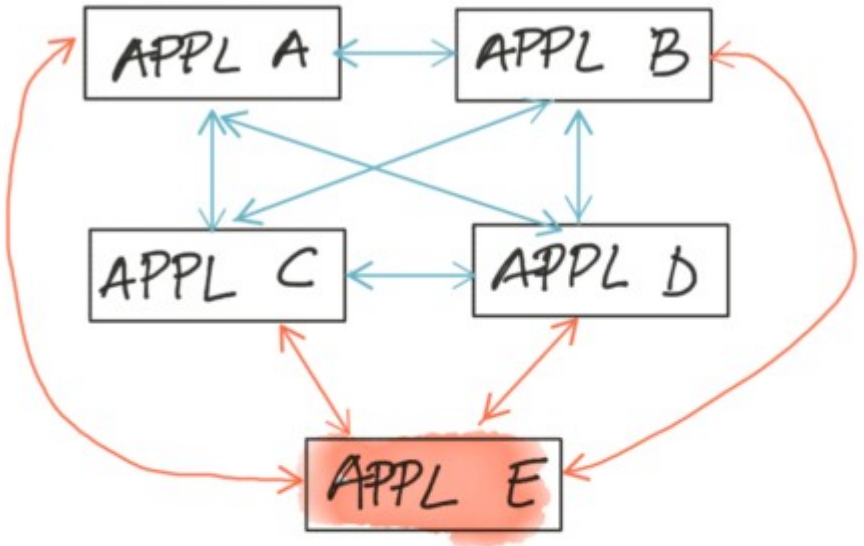
Lucas Pauzies

PAYLEAD

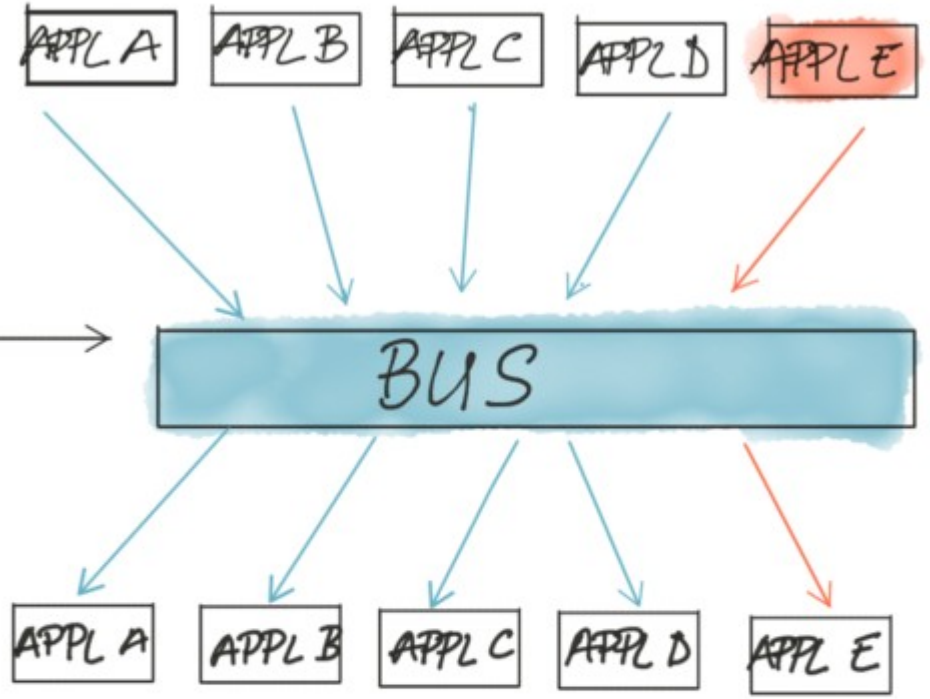
Historique

- Créé en 2011
- Créé par Linkedin
- Maintenu par Apache et Linkedin

Event/Message Driven Architecture

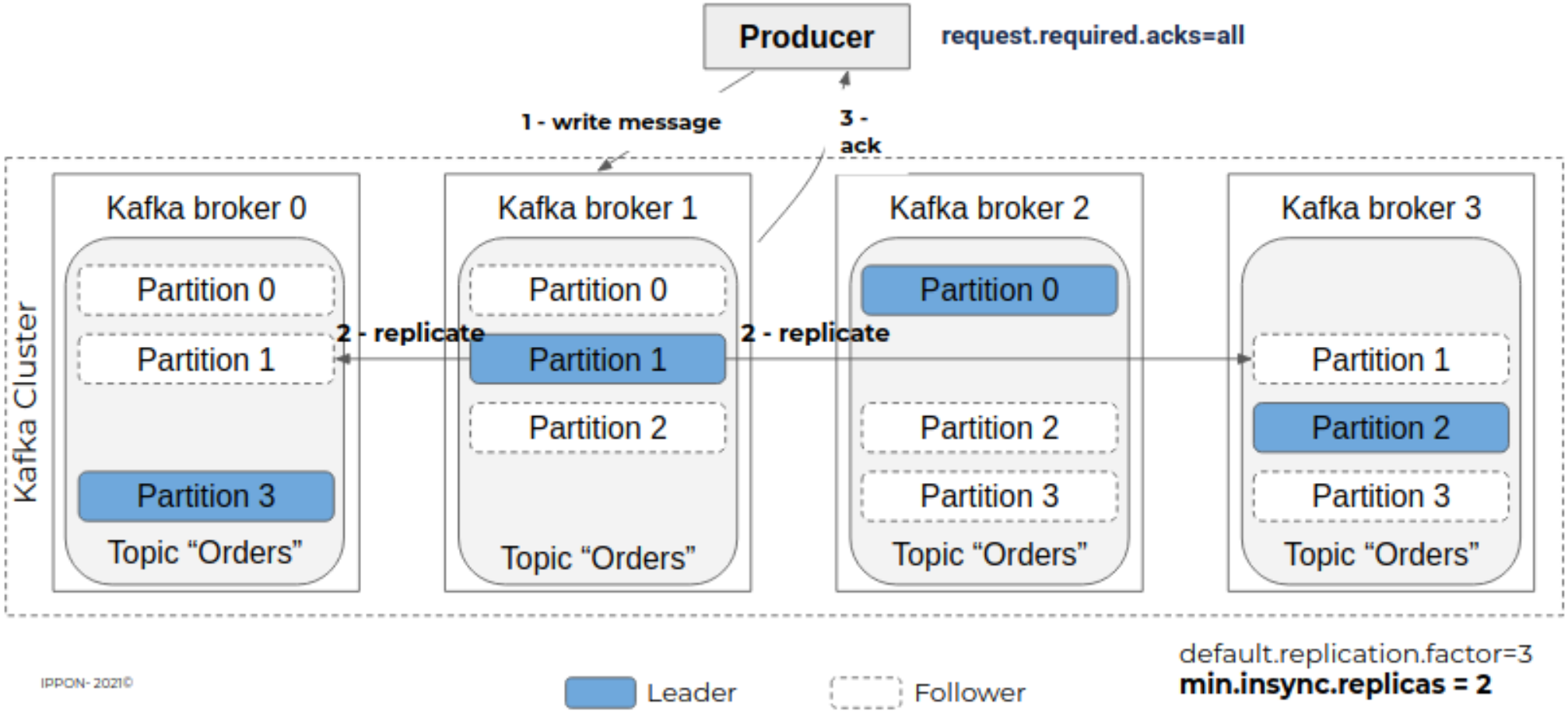


Integration point à point



Architecture en Bus (Pub-Sub)

Wtf is this?


Producer
request.required.acks=all
1 - write message
3 - ack
Kafka broker 0
Kafka broker 1
Kafka broker 2
Kafka broker 3
Kafka Cluster
Partition 0
Partition 1
Partition 2
Partition 3
2 - replicate
2 - replicate
Topic "Orders"
IPPON- 2021©
Leader
Follower
default.replication.factor=3
min.insync.replicas = 2

Broker

- Est une instance de Apache Kafka
- Plusieurs brokers créent un cluster

Topic

- Est une façon de regrouper des messages au même endroit
- En général, un topic = un type de message dans l’application
- Un broker Kafka peut contenir une infinité de topics

Partition

- Un topic peut être découpé en plusieurs partitions
- Séquence immuable
- Séquence ordonnée
- Se comporte comme une file de message (FIFO)
- Unité de parallélisation permettant de gérer les montées en charge
- Peut être répliqué sur plusieurs brokers pour être tolérant à la panne

Producer

- Tout système qui publie dans un topic Kafka
- Ne peut pas supprimer un message d’un topic Kafka
- Ecrit dans la partition « Leader », Kafka se charge de la réplication

Quid avec kafka-python ?

```
>>> from kafka import KafkaProducer
>>> producer = KafkaProducer(bootstrap_servers='localhost:1234')
>>> for _ in range(100):
...     producer.send('foobar', b'some_message_bytes')
```

Consumer

- Tout système qui lit dans un topic Kafka
- Un message lu ne disparaît pas du topic
- Plusieurs consumers peuvent lire le même topic
- Les consumer peuvent être groupés en consumer-group

Quid avec kafka-python ?

```
>>> from kafka import KafkaConsumer
>>> consumer = KafkaConsumer('my_favorite_topic')
>>> for msg in consumer:
...     print (msg)
```

Message ?

- Une chaîne de caractère
- Doit être le plus petit et concis possible
- Doit assurer un contrat d’interface entre le Producer et le Consumer