Introduction to Apache Kafka

2020 INTRO. TO NETWORK PROGRAMMING

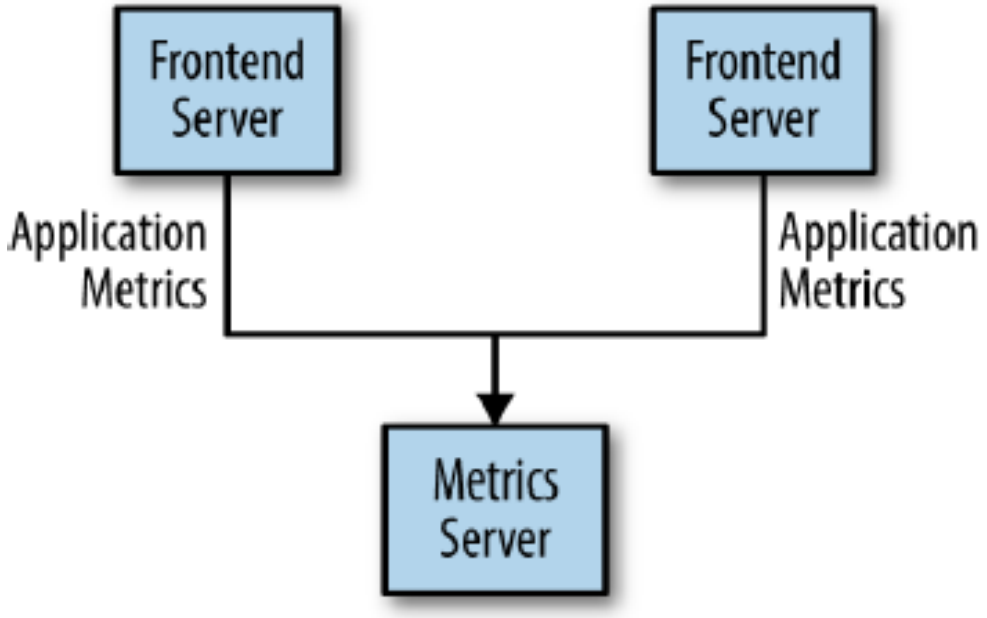
TA 城溥

Outline

- Why Kafka
- What is Kafka
- Terminologies
- Kafka Architecture
- Producer
- Consumer Subscribe vs Assign
- Example Architectures for HW4
- Hint

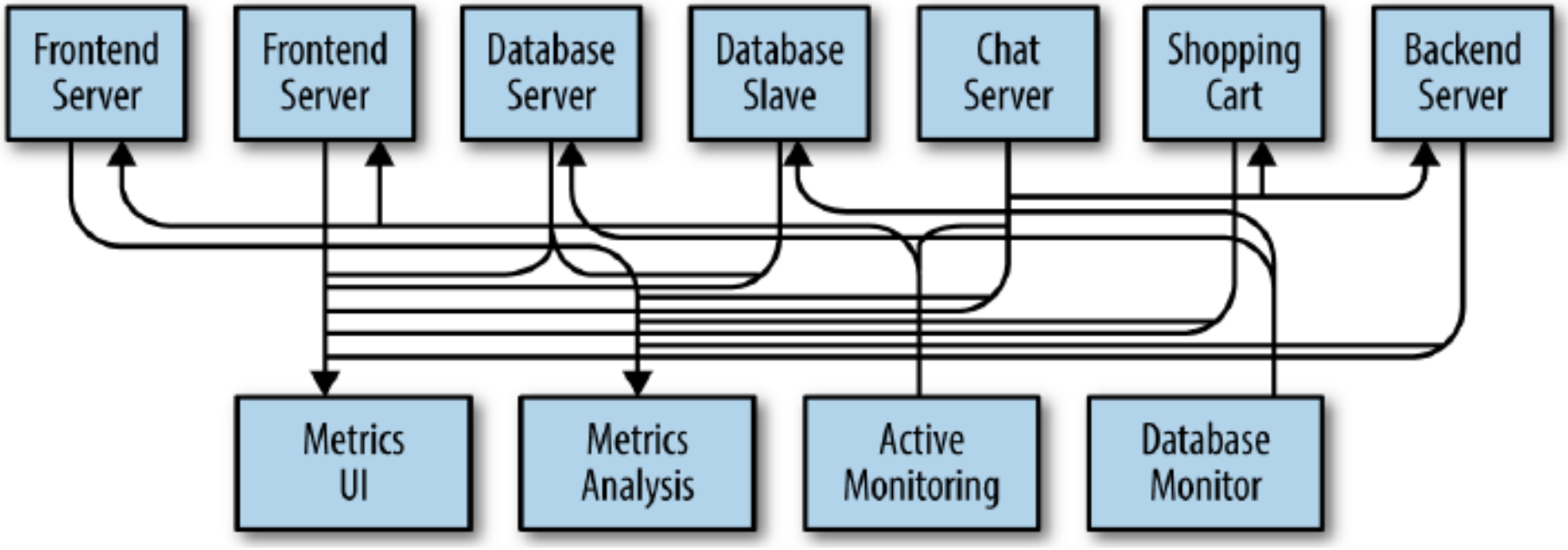
Why Kafka

- A single, direct metrics publisher



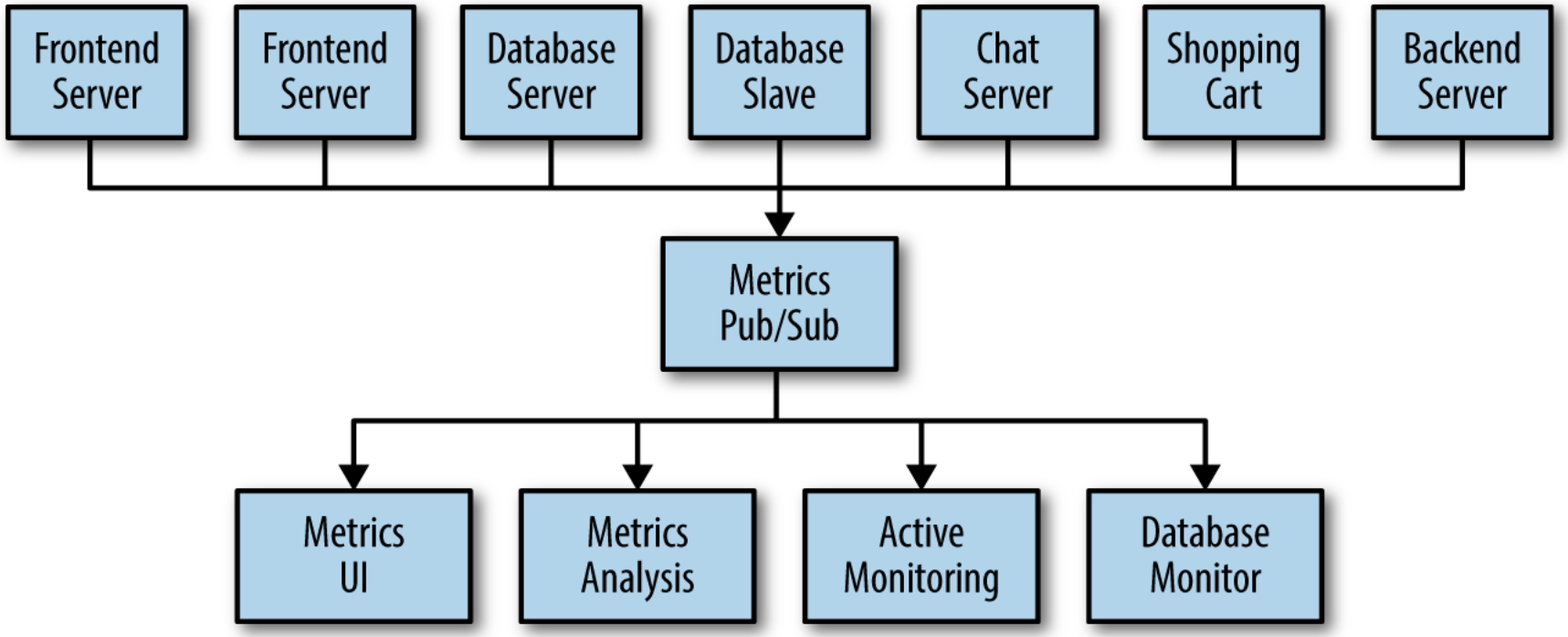
Why Kafka (cont.)

- Many metrics publishers, using direct connections



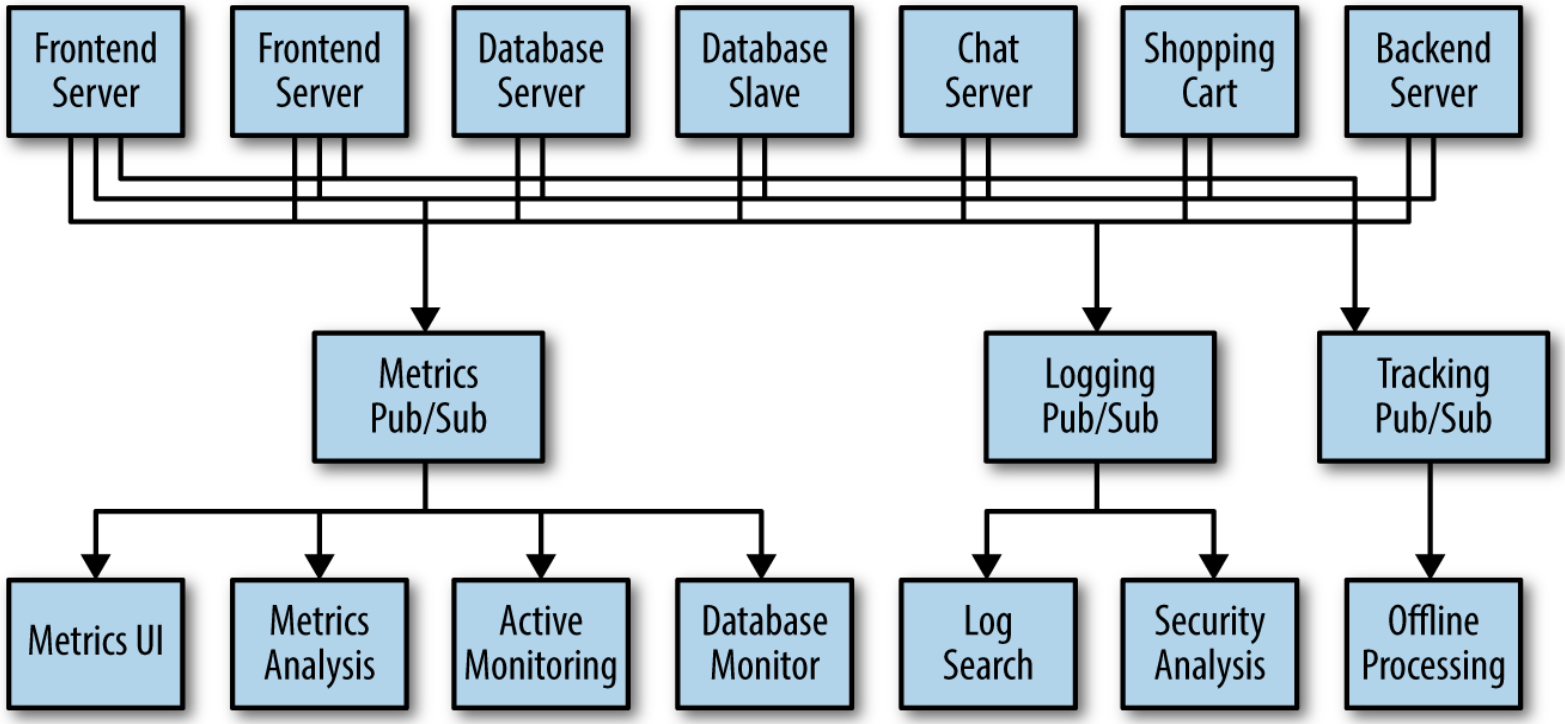
Why Kafka (cont.)

- A metrics publish/subscribe system



Why Kafka (cont.)

- Multiple publish/subscribe systems



Why Kafka (cont.)

- Twitter's Kafka adoption story
  - https://blog.twitter.com/engineering/en_us/topics/insights/2018/twitters-kafka-adoption-story.html

Apache Kafka vs RabbitMQ vs ActiveMQ

Stack Overflow Trends

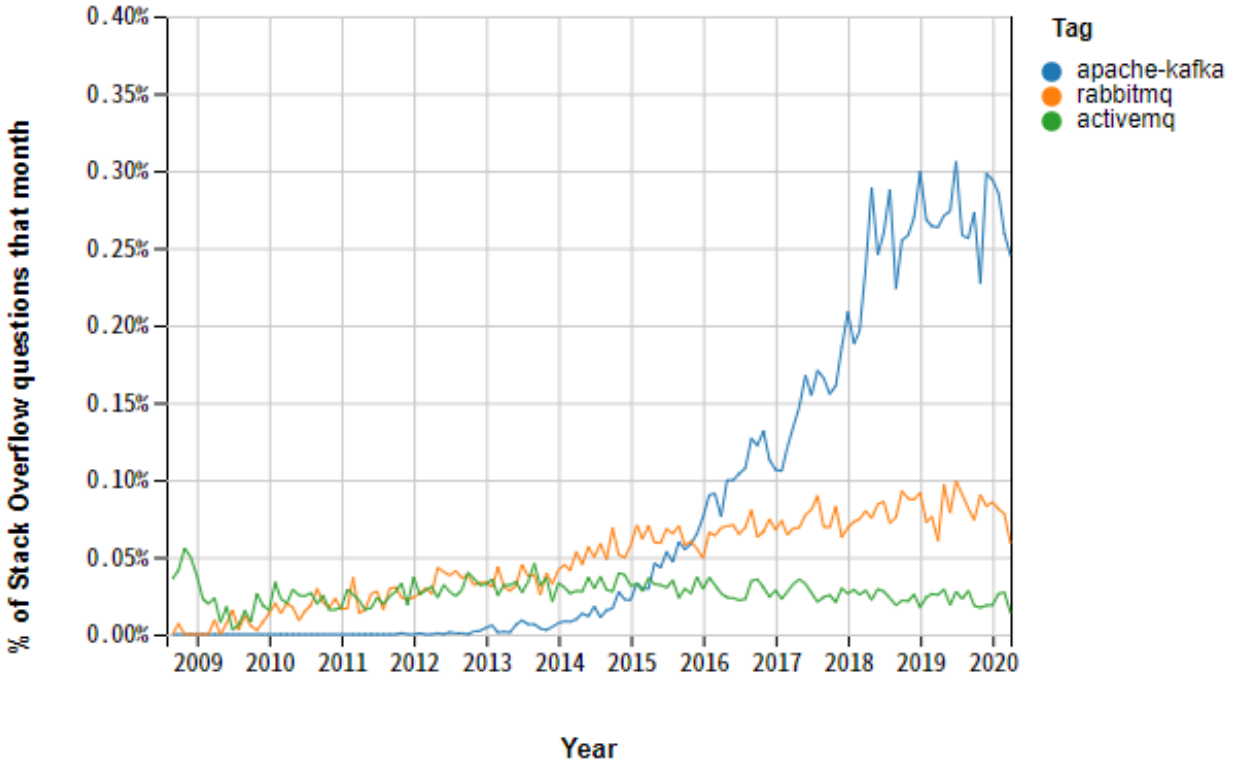
See how technologies have trended over time based on use of their tags since 2008, when Stack Overflow was founded. Enter up to 15 tags to compare growth and decline.

Tags:

apache-kafka × rabbitmq × activemq ×

Don't know what tags to look at? Try one of our presets:

- Most Popular Languages (TIOBE Index for May 2017)
- Operating Systems
- Mobile Operating Systems
- Javascript Frameworks
- Smaller Javascript Frameworks
- Closed-source Browser Plugins
- Data Science and Big Data
- Apache Open-source Projects



What is Apache Kafka

- Apache Kafka is a distributed streaming platform
  - Publish and subscribe to streams of records
  - Similar to a message queue or enterprise messaging system
  - Store streams of records in a fault-tolerant durable way
  - Process streams of records as they occur

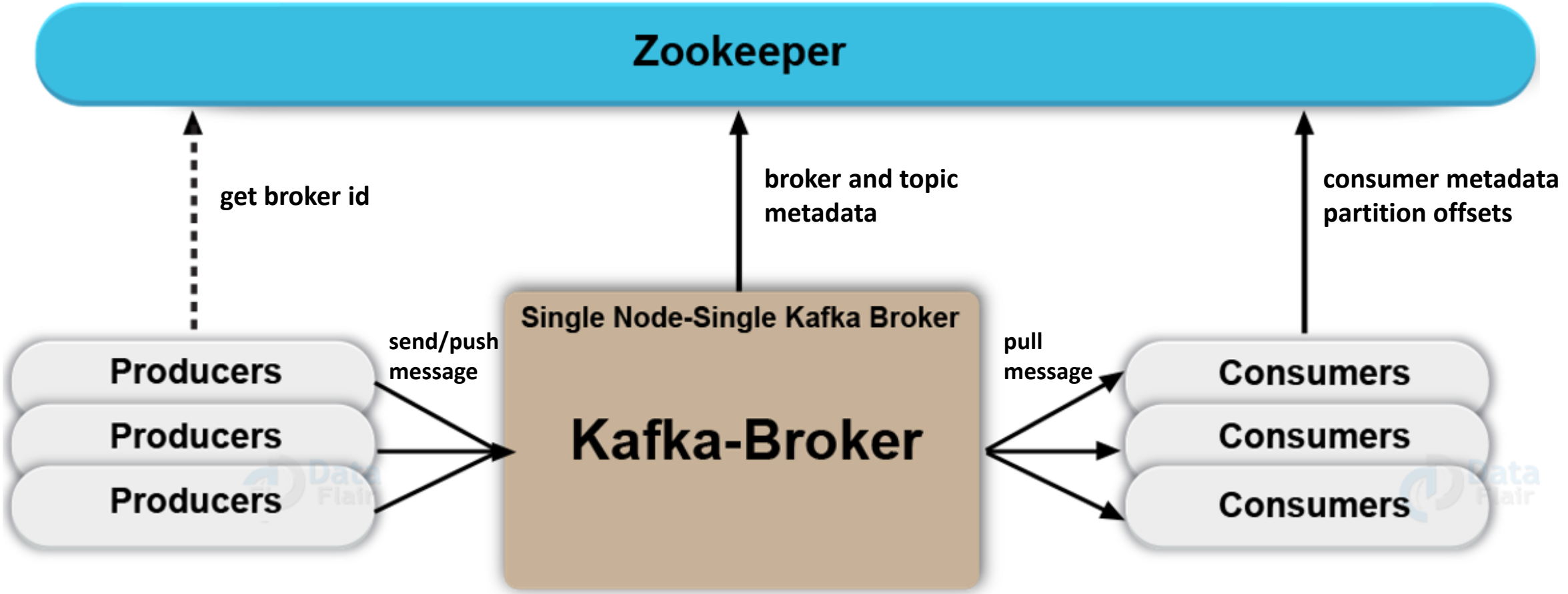
Terminologies

- **Zookeeper** – Centralized service for maintaining **metadata**
- **Kafka broker** – Responsible for receiving and storing the data
- **Topic** – A category or feed name to which records are published
  - Topics can be split into **partitions**, each partition is ordered
- **Record** – It is the data or message to be sent
  - Each record consists of a key, a value, and a timestamp
  - Key is used to **determine the partition**, and value is the **record contents**

Terminologies (cont.)

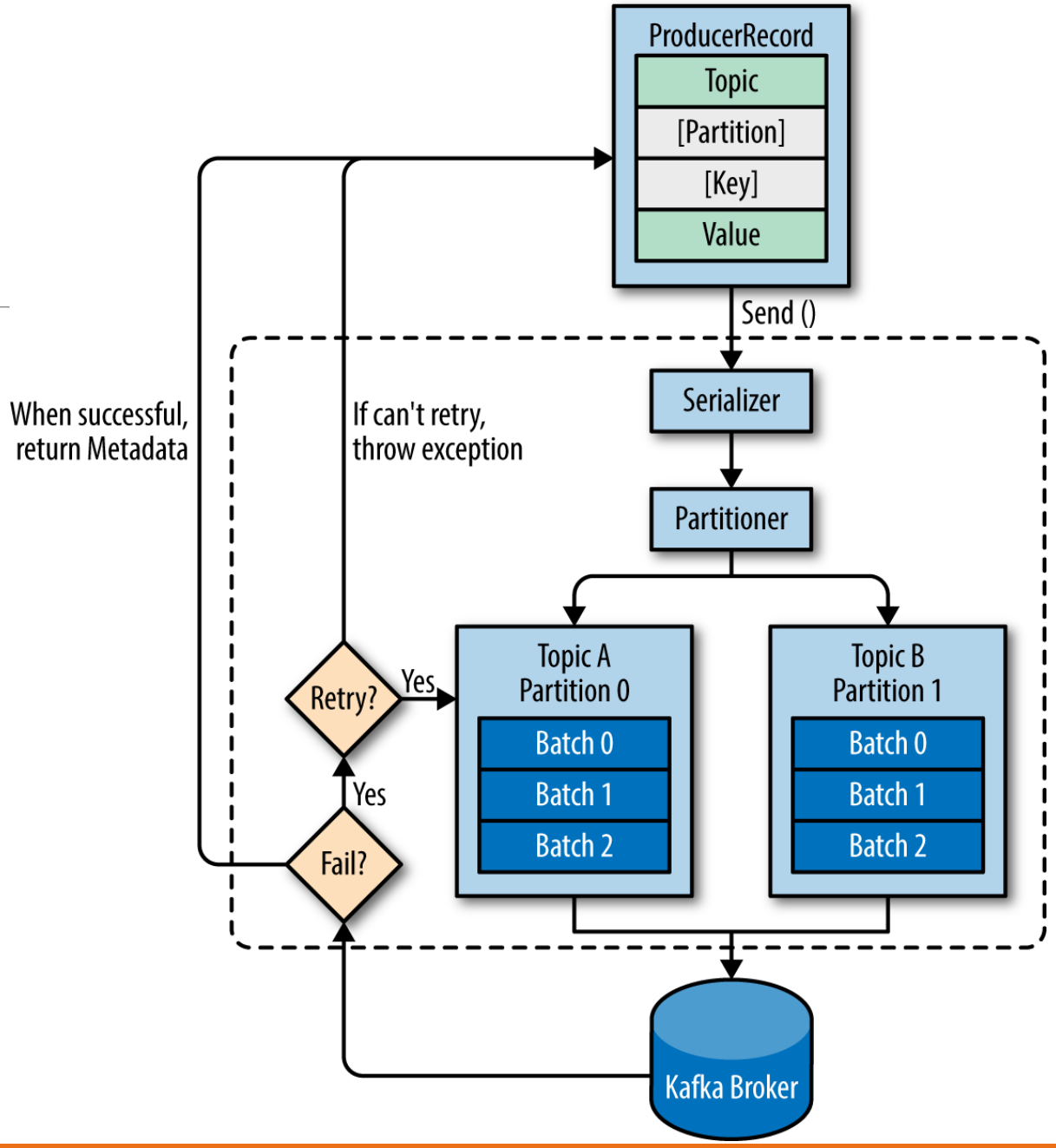
- **Producer** – Responsible for choosing which record to assign within the specified topic
- **Consumer** – The processes that subscribe to topics and process as well as read the feed of published messages
  - Consumers can label themselves with a **consumer group** name, and each record published to a topic is **delivered to one consumer** within each subscribing consumer group

Kafka Architecture



Producer

- High-level overview of Kafka producer components
- Producer will decide target partition depend on the key
- Batch records together for efficiency



Consumer – Subscribe vs Assign

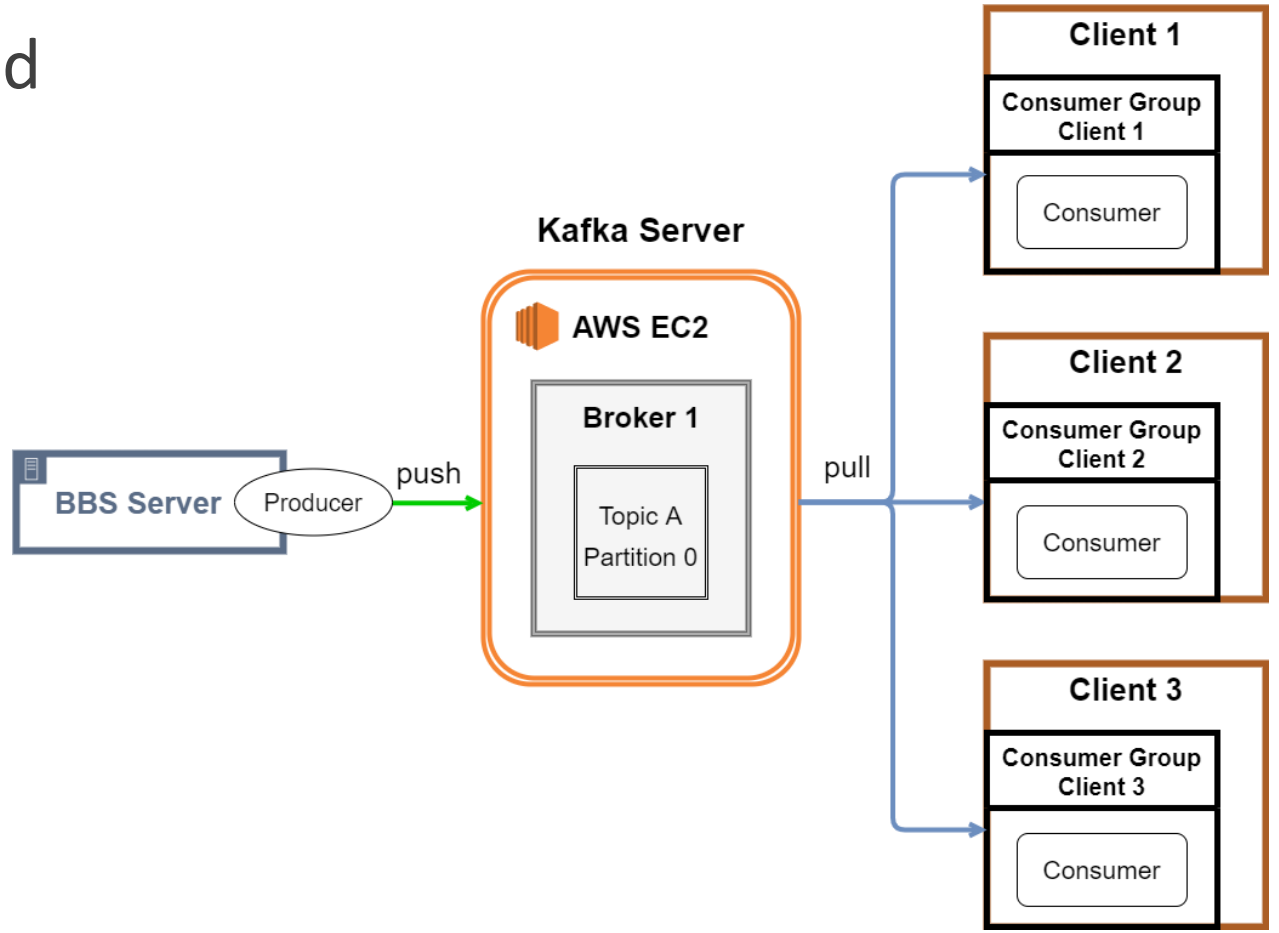
- Subscribe – This method will subscribe to the given list of topics to get **dynamically assigned** partitions

- Assign – It needs **manually assign** a list of partitions to the consumer

Example architecture for HW4

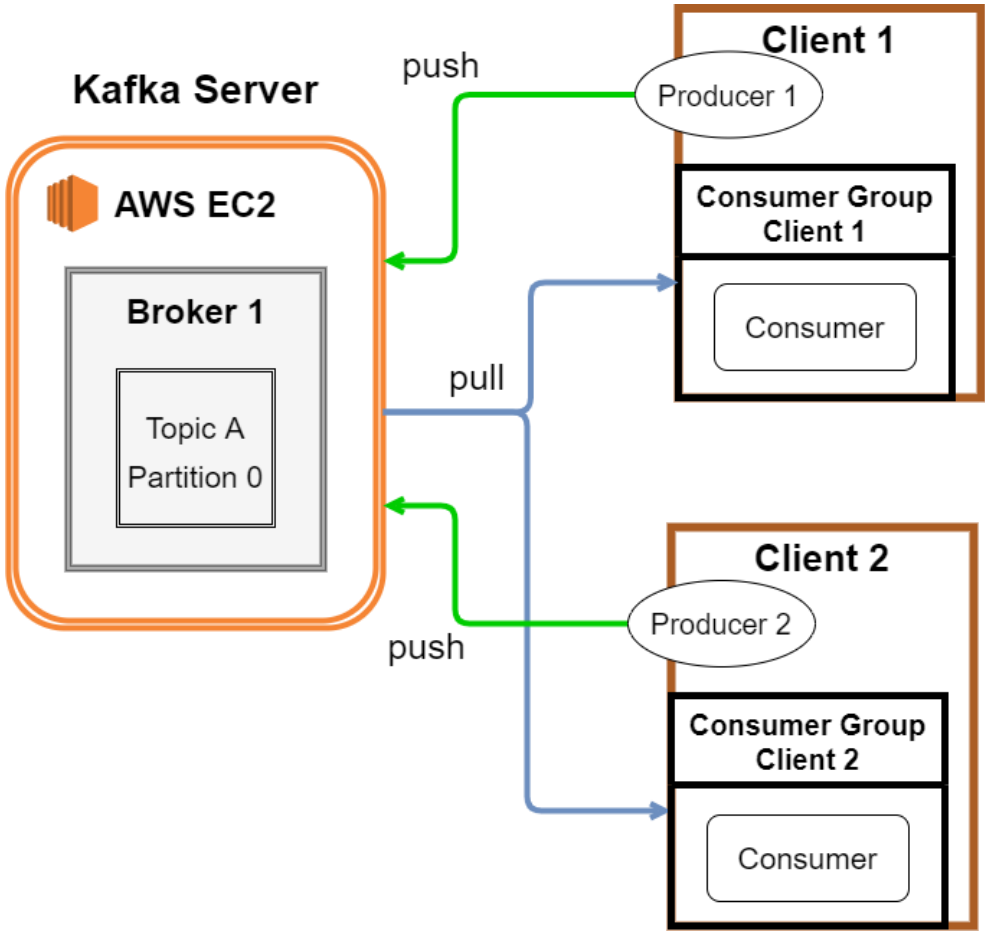
- BBS server is a producer, and

  each client is a consumer



Example architecture for HW4

- Each client act as both a producer and a consumer



Kafka clients

- C/C++
  - https://github.com/edenhill/librdkafka
- Python
  - https://github.com/dpkp/kafka-python
  - https://github.com/confluentinc/confluent-kafka-python
- Node.js
  - https://github.com/Blizzard/node-rdkafka
  - https://github.com/tulios/kafkajs
  - https://github.com/SOHU-Co/kafka-node

Kafka client – C/C++ (edenhill/librdkafka)

- Installing prebuilt packages
  - https://pkgs.org/search/?q=librdkafka-dev
  - https://github.com/edenhill/librdkafka/tree/0.11.x/examples

- Build from source
  - https://github.com/edenhill/librdkafka/tree/master/examples
  - https://github.com/edenhill/librdkafka/issues/466

Kafka client – Python (dpkp/kafka-python)

- How to install
  - https://github.com/dpkp/kafka-python/blob/master/docs/install.rst
- Example
  - https://github.com/dpkp/kafka-python/blob/master/example.py

Kafka client – Python (confluent-kafka-python)

- How to install
  - https://github.com/confluentinc/confluent-kafka-python#install
- Example
  - https://github.com/confluentinc/confluent-kafka-python#usage

Hint – Poll Loop

- At the heart of the consumer API is a **simple loop for polling the server**

- Consumers are usually long-running applications that **continuously poll Kafka** for more data

Hint – Consumer Thread safety

- **One consumer per thread is the rule**
- Can’t have multiple consumers that belong to the **same group in one thread**
- Can’t have multiple threads safely **use the same consumer**

References

- https://kafka.apache.org/intro
- https://www.confluent.io/wp-content/uploads/confluent-kafka-definitive-guide-complete.pdf
- https://data-flair.training/blogs/kafka-terminologies/
- http://www.diegocalvo.es/wp-content/uploads/2018/06/kafka-architecture.png