



Cloud Computing

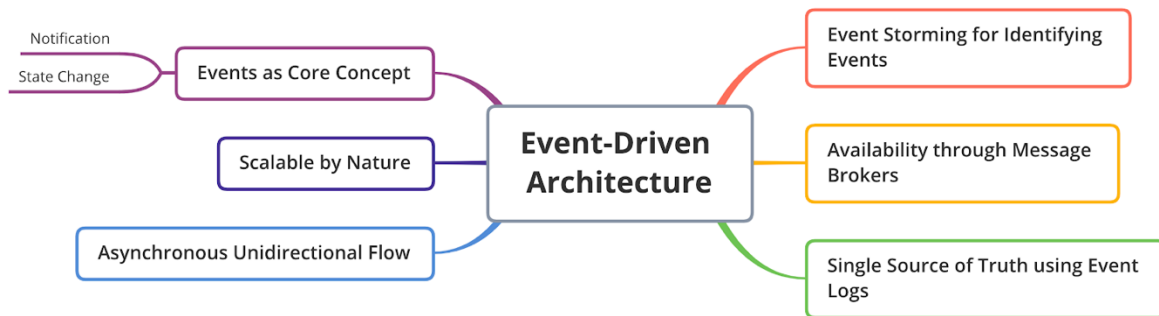
Project Milestone – Data Ingestion Software – Kafka Clusters

Jane Coralde | 100660214

Table of Contents

1. EVENT-DRIVEN ARCHITECTURE.....	3
2. MAIN CONCEPTS OF EDA AS CORE OF KAFKA.....	3
3. Q'	3
3.1. DEFINE EDA. ADVANTAGES AND DISADVANTAGES	3
3.2. IN KAFKA, WHAT'S MEANT BY CLUSTER, BROKER, TOPIC, REPLICATION, ZOOKEEPER, CONTROLLER, LEADER, CONSUMER, PRODUCER, AND CONSUMER GROUP	3
4. KAFKA IN LOCAL MACHINE	4
4.1. TASK.....	4
4.2. EXECUTION	4
4.3. PROBLEMS ENCOUNTERED	6
5. NODEJS SCRIPTS TO CREATE KAFKA TOPICS, CONSUMER, AND PRODUCERS.....	7
5.1. TASK.....	7
5.2. EXECUTION	8
5.3. PROBLEM(S) ENCOUNTERED.....	ERROR! BOOKMARK NOT DEFINED.
6. PYTHON SCRIPTS TO CREATE TOPIC, PRODUCER, AND CONSUMERS	9
6.1. TASK.....	9
6.2. EXECUTION	9
7. PREPARE A VIDEO	9
8. UPDATE YAML FOR PERSISTENT DATA	9
9. KAFKA IN CONFLUENT CLOUD.....	9

1. Event-Driven Architecture



2. Main Concepts of EDA as Core of Kafka

Event-driven architectures are ideal for distributed application architectures. Kafka is distributed by design, to provide a high-performance messaging system with a log-aggregation system.

Replication is the heart of Kafka's architecture, which guarantees users the availability of data and durability between message transactions. Replication is done by partitions that can handle storage and pub/sub independently to others. It is done so that in the case that one of the machines fails; the data DOES NOT get lost.

3. Q'

3.1. Define EDA. Advantages and Disadvantages

Event-driven architecture (EDA) is a software architecture paradigm promoting the production, detection, consumption of, and reaction to **events**.

-Wikipedia

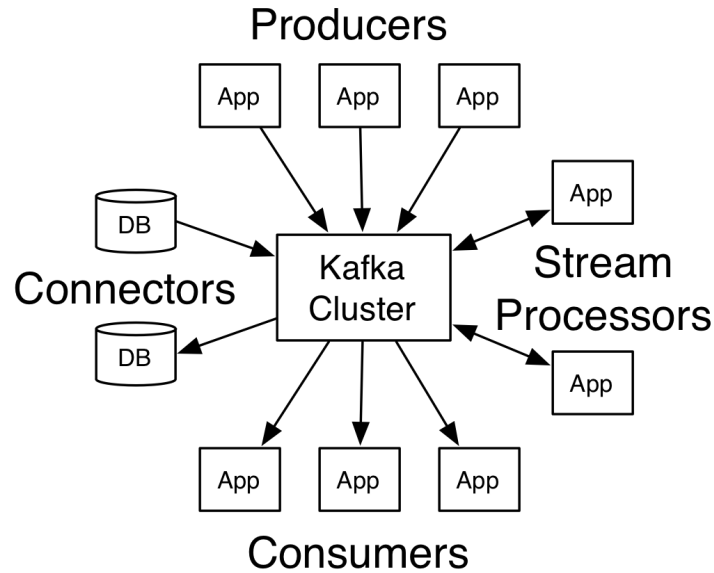
In a nutshell, EDA allows us to solve modern web demands. Advantages are as follows:

- Availability – machines are independent if one is down the others are not affected
- Scalability – multiple machines instead of one
- Asynchronous nature – unlike request/response, it does not need to wait for a block for code execution and therefore eliminating risk of timeouts

3.2. In Kafka, what's meant by cluster, broker, topic, replica, partition, zookeeper, controller, leader, consumer, producer, and consumer group

Kafka allows PRODUCERS to write data for CONSUMERS to retrieve MESSAGES between applications in distributed systems through the stream. MESSAGES are a key-value pair and grouped into TOPICS. The consumer gets the message (of one topic) from multiple

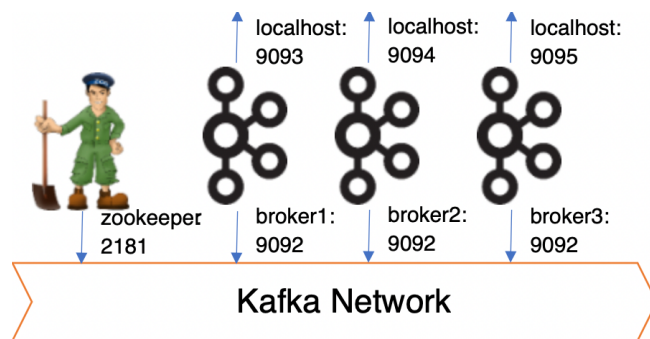
producers, and in the same way, the producer writes data to every subscriber of that topic. This is done through a KAFKA CLUSTER. ZOOKEEPER is introduced as to track status of (leader) nodes and maintain order in queue.



4. Kafka in Local Machine

4.1. Task

Follow the following [video](#) to install Kafka into your local machine and create topics, consumers, and producers using Kafka's built-in tools.



4.2. Execution

Below, I've listed all the commands used and attached screenshots I've taken.

```
docker-compose up -d
```

```
[+] Running 5/5
# Network kafka_Network Created 0.0s
# Container zookeeper Started 0.4s
# Container broker1 Started 1.2s
# Container broker3 Started 1.3s
# Container broker2 Started 1.1s
```

```
devorahjanecoralde@Devorahs-MacBook-Pro Lab2 % docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
dc34b09a55db	confluentinc/cp-kafka	"/etc/confluent/dock_..."	2 minutes ago	Up 2 minutes	9092/tcp, 0.0.0.0:9094->909
4/tcp	broker2				
4ded8546b7fb	confluentinc/cp-kafka	"/etc/confluent/dock_..."	2 minutes ago	Up 2 minutes	9092/tcp, 0.0.0.0:9095->909
5/tcp	broker3				
da12fe100aa9	confluentinc/cp-kafka	"/etc/confluent/dock_..."	2 minutes ago	Up 2 minutes	9092/tcp, 0.0.0.0:9093->909
3/tcp	broker1				
f20528caad4c	confluentinc/cp-zookeeper	"/etc/confluent/dock_..."	2 minutes ago	Up 2 minutes	2888/tcp, 0.0.0.0:2181->218
1/tcp, 3888/tcp	zookeeper				

```
docker exec broker1 kafka-topics --create --topic topic --partitions 3 --replication-
factor 3 --if-not-exists --bootstrap-server broker1:9092
```

```
docker exec broker1 kafka-topics --create --topic topic2 --partitions 3 --replication-
factor 2 --if-not-exists --bootstrap-server broker1:9092,broker2:9092,broker3:9092
```

```
docker exec broker1 kafka-topics --describe --bootstrap-server broker1:9092
```

```
devorahjanecoralde@Devorahs-MacBook-Pro v1 % docker exec broker1 kafka-topics --describe --bootstrap-server broker1:9092
```

Topic	TopicId	PartitionCount	ReplicationFactor	Configs
Topic: topic	TopicId: zx_QgdMEQyG-0zZJgcXLYQ	PartitionCount: 3	ReplicationFactor: 3	Configs:
Topic: topic	Partition: 0	Leader: 1	Replicas: 1,2,3	Isr: 1,2,3
Topic: topic	Partition: 1	Leader: 2	Replicas: 2,3,1	Isr: 2,3,1
Topic: topic	Partition: 2	Leader: 3	Replicas: 3,1,2	Isr: 3,1,2
Topic: topic2	TopicId: LR5VAParT3mSrdtYpa-CGw	PartitionCount: 3	ReplicationFactor: 2	Configs:
Topic: topic2	Partition: 0	Leader: 2	Replicas: 2,3	Isr: 2,3
Topic: topic2	Partition: 1	Leader: 3	Replicas: 3,1	Isr: 3,1
Topic: topic2	Partition: 2	Leader: 1	Replicas: 1,2	Isr: 1,2

```
docker exec broker1 kafka-topics --list --bootstrap-server broker1:9092
```

```
docker exec broker2 bash -c "echo 'value1' | kafka-console-producer --request-
required-acks 1 --broker-list broker2:9092 --topic topic"
```

```
docker exec broker1 bash -c "seq 0 2.5 10 | kafka-console-producer --request-required-
acks 1 --broker-list broker3:9092,broker2:9092,broker1:9092 --topic topic"
```

```
docker exec broker1 bash -c "echo key,value | kafka-console-producer --broker-list
broker1:9092 --topic topic --property parse.key=true --property key.separator=,"
```

```
docker exec broker2 kafka-console-consumer --bootstrap-server broker1:9092 --topic
topic --from-beginning
```

```
devorahjanecoralde@Devorahs-MacBook-Pro ~ % docker exec broker2 kafka-console-
consumer --bootstrap-server broker1:9092 --topic topic --from-beginning
0.0
2.5
5.0
7.5
10.0
0.0
2.5
5.0
7.5
10.0
value1
value1
temp=20
```

*Note: I had problems where I entered the commands again due to one of the brokers exiting and causing some problems in the **Kafka-network**. Please see 4.3 for details.*

```
docker exec broker2 kafka-console-consumer --bootstrap-server broker1:9092 --topic
topic --from-beginning --max-messages 10
```

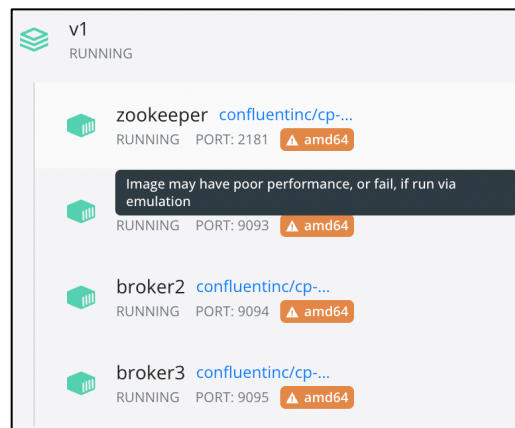
```
docker exec -it broker1 kafka-console-producer --broker-list
broker1:9092,broker2:9092,broker3:9092 --topic topic2
```

```
docker exec broker1 kafka-console-consumer --bootstrap-server
broker1:9092,broker2:9092,broker3:9092 --topic topic2 --from-beginning
```

```
docker-compose down
```

4.3. Problems Encountered

During running docker, images had some problem and would exit by themselves. This would create problems for the Kafka-network. See image below.



As such, terminal would sometimes give me these error messages.

```
devorahjanecoralde@Devorahs-MacBook-Pro ~ % docker exec broker2 kafka-console-consumer --bootstrap-server broker1:9092 --
topic topic --from-beginning --max-messages 10
Error response from daemon: Container fba428264097b79e15ffed3b3502aaa6ab4a5ab44bd59c49e334fdbbc4d392f8 is not running
```

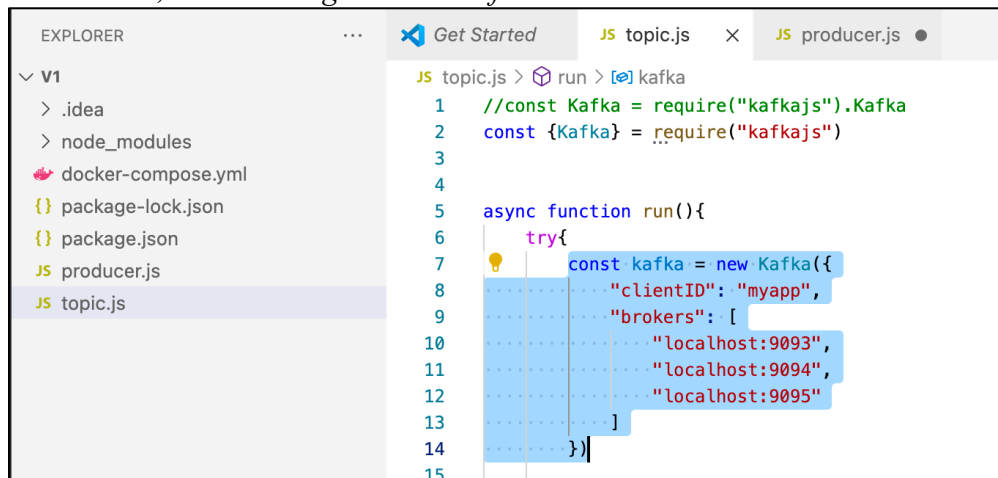
```
devorahjanecoralde — com.docker.cli - docker exec broker2 kafka-console-consumer --bootstrap-server broker1:9092...
devorahjanecoralde@Devorahs-MacBook-Pro ~ % docker exec broker2 kafka-console-consumer --bootstrap-server broker1:9092 --
topic topic --from-beginning
[2022-02-15 13:40:13,969] WARN Couldn't resolve server broker1:9092 from bootstrap.servers as DNS resolution failed for b
roker1 (org.apache.kafka.clients.ClientUtils)
[2022-02-15 13:40:14,007] ERROR Unknown error when running consumer: (kafka.tools.ConsoleConsumer$)
org.apache.kafka.common.KafkaException: Failed to construct kafka consumer
    at org.apache.kafka.clients.consumer.KafkaConsumer.<init>(KafkaConsumer.java:823)
    at org.apache.kafka.clients.consumer.KafkaConsumer.<init>(KafkaConsumer.java:664)
    at org.apache.kafka.clients.consumer.KafkaConsumer.<init>(KafkaConsumer.java:645)
    at kafka.tools.ConsoleConsumer$.run(ConsoleConsumer.scala:65)
    at kafka.tools.ConsoleConsumer$.main(ConsoleConsumer.scala:52)
    at kafka.tools.ConsoleConsumer.main(ConsoleConsumer.scala)
Caused by: org.apache.kafka.common.config.ConfigException: No resolvable bootstrap urls given in bootstrap.servers
    at org.apache.kafka.clients.ClientUtils.parseAndValidateAddresses(ClientUtils.java:89)
    at org.apache.kafka.clients.ClientUtils.parseAndValidateAddresses(ClientUtils.java:48)
    at org.apache.kafka.clients.consumer.KafkaConsumer.<init>(KafkaConsumer.java:730)
    ... 5 more
devorahjanecoralde@Devorahs-MacBook-Pro ~ % docker exec broker2 kafka-console-consumer --bootstrap-server broker1:9092 --
topic topic --from-beginning
[2022-02-15 13:40:30,692] WARN [Consumer clientId=consumer-console-consumer-18355-1, groupId=console-consumer-18355] Conn
ection to node -1 (broker1/172.20.0.4:9092) could not be established. Broker may not be available. (org.apache.kafka.clie
nts.NetworkClient)
[2022-02-15 13:40:30,696] WARN [Consumer clientId=consumer-console-consumer-18355-1, groupId=console-consumer-18355] Boot
strap broker broker1:9092 (id: -1 rack: null) disconnected (org.apache.kafka.clients.NetworkClient)
```

5. NodeJS Scripts to Create Kafka Topics, Consumer, and Producers.

5.1. Task

Follow the following [video](#) to generate NodeJS scripts for creating topics, consumers, and producers

- Use the docker application from the repository not that shown in the video.
- To use the docker application from the repository, Kafka brokers' addresses will be localhost:9093,9094, and 9095.
 - *Below, I have changed the code for the brokers*



```
EXPLORER
...
Get Started
JS topic.js
JS producer.js

V1
> .idea
> node_modules
docker-compose.yml
{} package-lock.json
{} package.json
JS producer.js
JS topic.js

JS topic.js > run > kafka
1 //const Kafka = require("kafkajs").Kafka
2 const {Kafka} = require("kafkajs")
3
4
5 async function run(){
6   try{
7     const kafka = new Kafka({
8       "clientId": "myapp",
9       "brokers": [
10        "localhost:9093",
11        "localhost:9094",
12        "localhost:9095"
13      ]
14    })
15  }
```

- Follow only the NodeJS scripts not the docker commands.
- When the partition is not specified, Kafka will decide the partition randomly or by hashing the key.

5.2. Execution

Running topic.js

```
devorahjanecoralde@Devorahs-MacBook-Pro v1 % node topic.js
Connecting....
Connected....
Created successfully!
```

Running producer.js test

```
devorahjanecoralde@Devorahs-MacBook-Pro v1 % node producer.js test
Connecting.....
Connected!
Send Successfully! [{"topicName":"Users","partition":1,"errorCode":0,"baseOffset":"3","logAppendTime":"-1","logStartOffset":"0"}]
```

```
devorahjanecoralde@Devorahs-MacBook-Pro v1 % node producer.js Jane
Connecting.....
Connected!
Send Successfully! [{"topicName":"Users","partition":0,"errorCode":0,"baseOffset":"0","logAppendTime":"-1","logStartOffset":"0"}]
```

Running consumer.js

```
devorahjanecoralde@Devorahs-MacBook-Pro v1 % node consumer.js
Connecting.....
Connected!
{"level":"INFO","timestamp":"2022-02-15T14:51:15.052Z","logger":"kafkajs","message":"[Consumer] Starting","groupId":"test"}
{"level":"ERROR","timestamp":"2022-02-15T14:51:15.538Z","logger":"kafkajs","message":"[Connection] Response GroupCoordinator(key: 10, version: 2)","broker":"localhost:9095","clientId":"kafkajs","error":"The group coordinator is not available","correlationId":0,"size":55}
{"level":"ERROR","timestamp":"2022-02-15T14:51:15.953Z","logger":"kafkajs","message":"[Connection] Response GroupCoordinator(key: 10, version: 2)","broker":"localhost:9095","clientId":"kafkajs","error":"The group coordinator is not available","correlationId":1,"size":55}
{"level":"ERROR","timestamp":"2022-02-15T14:51:16.727Z","logger":"kafkajs","message":"[Connection] Response GroupCoordinator(key: 10, version: 2)","broker":"localhost:9095","clientId":"kafkajs","error":"The group coordinator is not available","correlationId":2,"size":55}
{"level":"ERROR","timestamp":"2022-02-15T14:51:18.528Z","logger":"kafkajs","message":"[Connection] Response GroupCoordinator(key: 10, version: 2)","broker":"localhost:9095","clientId":"kafkajs","error":"The group coordinator is not available","correlationId":3,"size":55}
{"level":"INFO","timestamp":"2022-02-15T14:51:25.911Z","logger":"kafkajs","message":"[ConsumerGroup] Consumer has joined the group","groupId":"test","memberId":"kafkajs-51bc8e4b-1e1a-490c-8303-7c02cf978fb9","leaderId":"kafkajs-51bc8e4b-1e1a-490c-8303-7c02cf978fb9","isLeader":true,"memberAssignment":{"Users":[0,1]},"groupProtocol":"RoundRobinAssigner","duration":10858}
RVD Msg test on partition 1
RVD Msg test on partition 1
RVD Msg test on partition 1
RVD Msg test on partition 1
RVD Msg Jane on partition 0
```

Producers and consumer(s) side-by-side.

<pre>devorahjanecoralde@Devorahs-MacBook-Pr o v1 % node producer.js test Connecting..... Connected! Send Successfully! [{"topicName":"User s","partition":1,"errorCode":0,"baseOf fset":"4","logAppendTime":"-1","logSta rtOffset":"0"}] devorahjanecoralde@Devorahs-MacBook-Pr o v1 % node producer.js test Jane Connecting..... Connected! Send Successfully! [{"topicName":"User s","partition":1,"errorCode":0,"baseOf fset":"5","logAppendTime":"-1","logSta rtOffset":"0"}] devorahjanecoralde@Devorahs-MacBook-Pr o v1 % node producer.js test Meo Connecting..... Connected! Send Successfully! [{"topicName":"User s","partition":1,"errorCode":0,"baseOf fset":"6","logAppendTime":"-1","logSta rtOffset":"0"}] devorahjanecoralde@Devorahs-MacBook-Pr o v1 %</pre>	<pre>devorahjanecoralde@Devorahs-MacBook-Pr o v1 % node consumer.js Connecting..... Connected! {"level":"INFO","timestamp":"2022-02-1 5T14:54:16.720Z","logger":"kafkajs","m essage":"[Consumer] Starting","groupId ":"test"} {"level":"INFO","timestamp":"2022-02-1 5T14:54:32.667Z","logger":"kafkajs","m essage":"[ConsumerGroup] Consumer has joined the group","groupId":"test","me mberId":"kafkajs-79fe1a62-b97e-44c2-90 67-9f7d48558d8b","leaderId":"kafkajs-5 1bc8e4b-1e1a-490c-8303-7c02cf978fb9", "isLeader":false,"memberAssignment":{"U sers":[1],"groupProtocol":"RoundRobin Assigner","duration":15947} RVD Msg test on partition 1 RVD Msg test on partition 1 █</pre>	<div>node</div> <div>zsh</div> <div>node</div>
---	--	--

6. Python Scripts to Create Topic, Producer, and Consumers

6.1. Task

Using the python library Kafka-python, write three python scripts that

- Create a topic
- Produce messages on a topic. The message's key and the partition should be optional. Also, an error message should be displayed in the case of failure.
- Repeatedly consume messages from a topic and display the consumed messages whenever available. The group id should be optional.

6.2. Execution

Had difficulty with this section. After following a few tutorials, looked at my groupmates' files to see how to run the code – python in Kafka.

7. Prepare a Video

Video uploaded only contained Task 5.

8. (N/A) Update YAML for Persistent Data NOT DONE.

9. (N/A) Kafka in Confluent Cloud NOT DONE.