Welcome to the World of Distributed Messaging Queue

Consumer

- The consumer pull data directly to the broker that is the **leader for the partition**.
- Is Consumer use zookeeper?
  - Yes (0.8.x)
  - No (0.9.x +)

Consumer

- General messaging system has two model:
  - Queuing
  - Publish-Subscribe
- Queuing:
  - A pool of consumers will read from a server and each message goes to one of them.
- Publish-Subscribe:
  - The message is broadcast to all consumers.

Consumer

1. Which model Kafka consumer support?

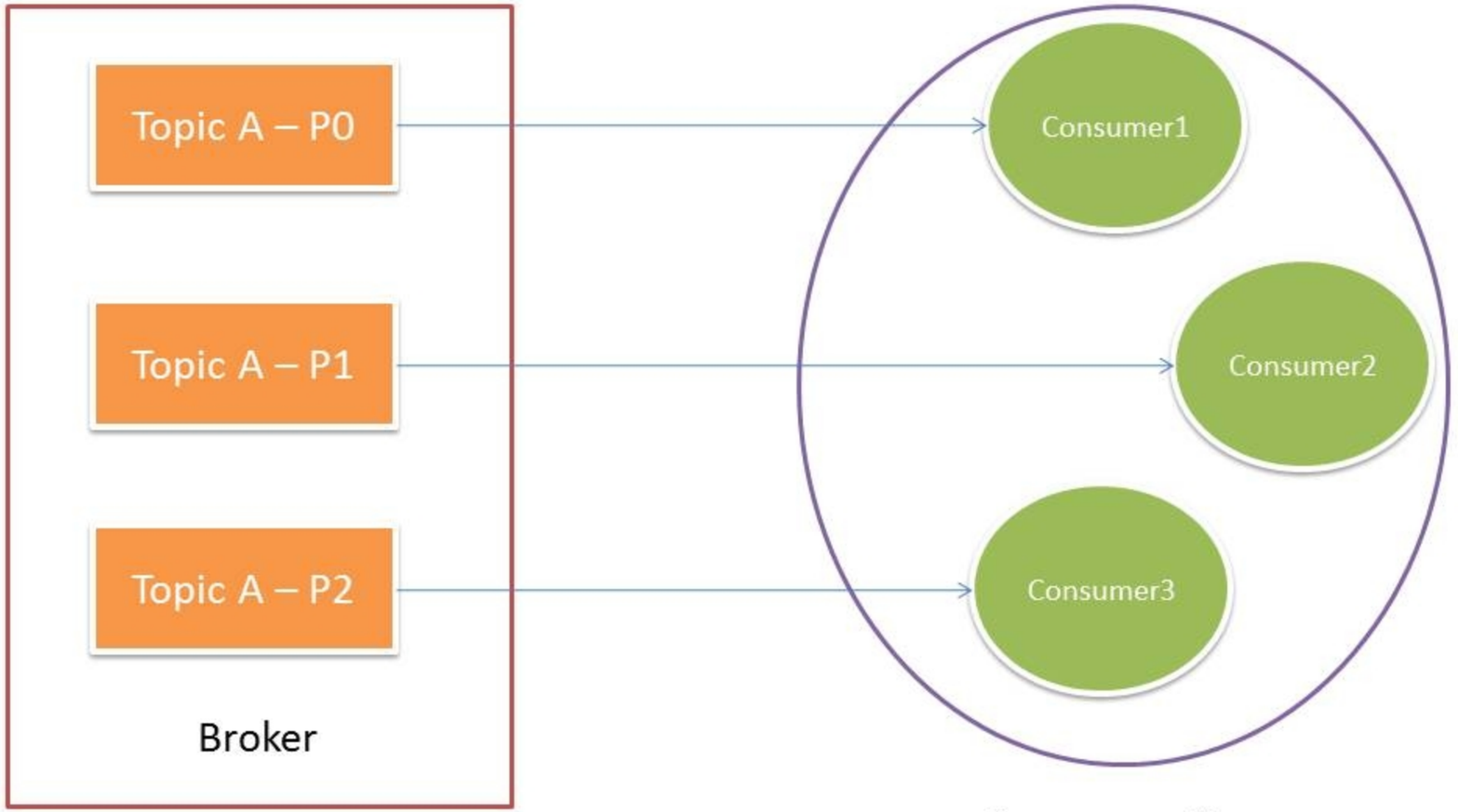
Consumer

- Which model Kafka consumer support?
  - Kafka Consumer supports both queuing and publish-subscribe mode.
- How?
  - Each Kafka Consumer belongs to a consumer group
  - A one consumer group can have multiple consumer instance
  - Each message published to a topic is delivered to one consumer instance within each subscribing consumer group.
  - Consumer instances can be in separate processes or on separate machines.

Consumer

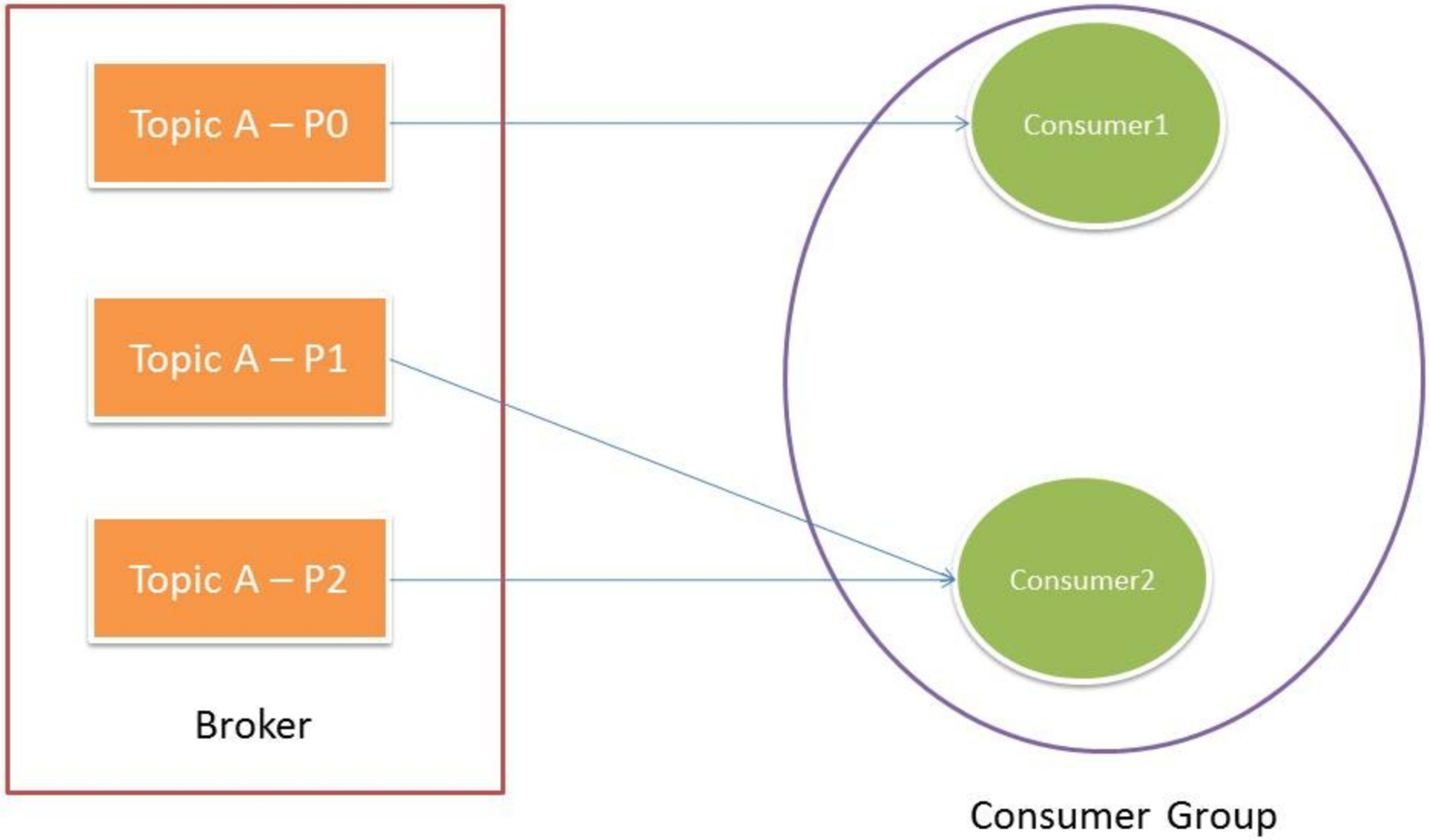
- If all the consumer instances have the same consumer group, then this works just like a traditional queue balancing load over the consumers.
- If all the consumer instances have different consumer groups, then this works like publish-subscribe and all messages are broadcast to all consumers.

Consumer - Queuing

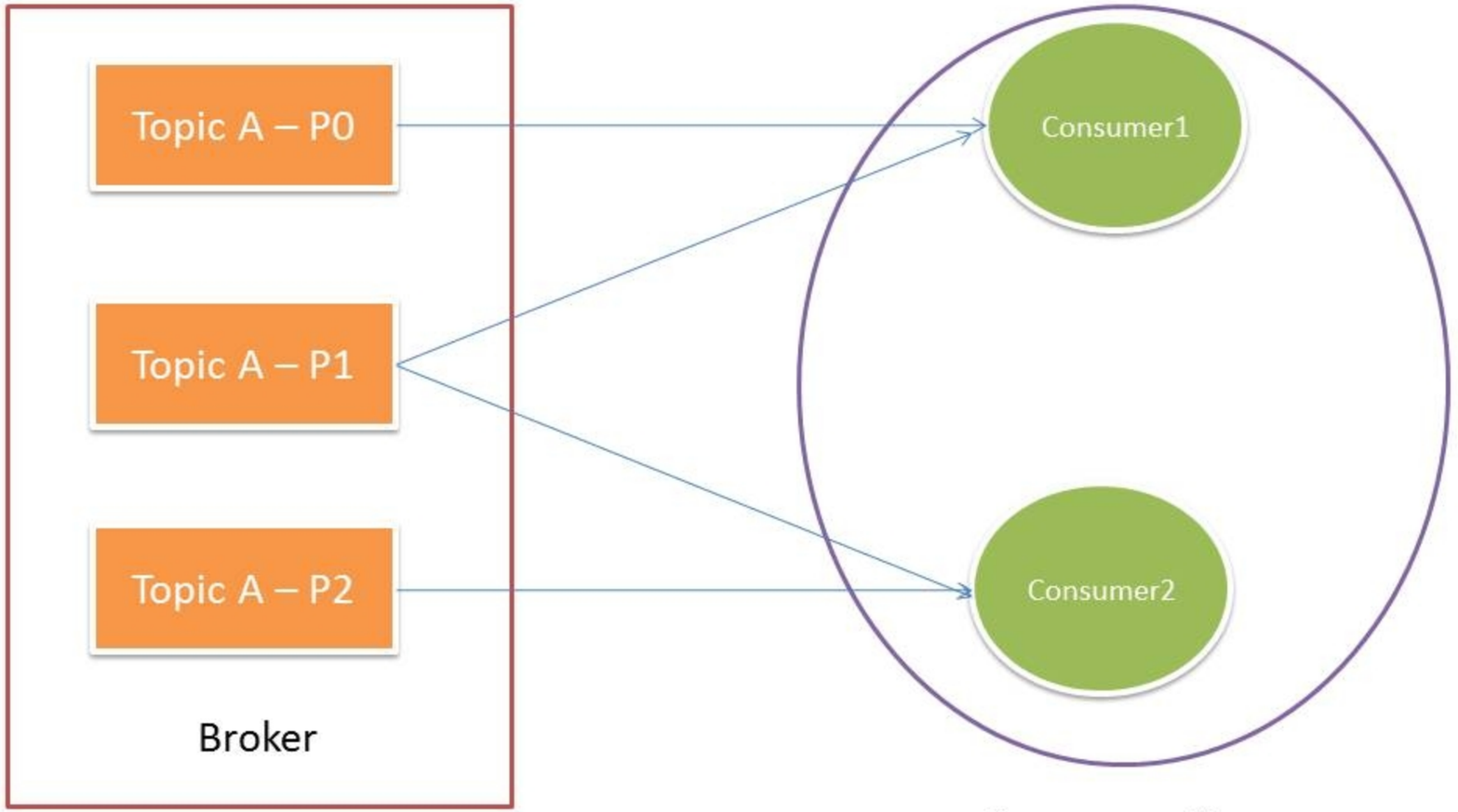


Ideally, each consumer instance will read data from one partition if number of consumer instances equals to number of partitions.

Consumer - Queuing

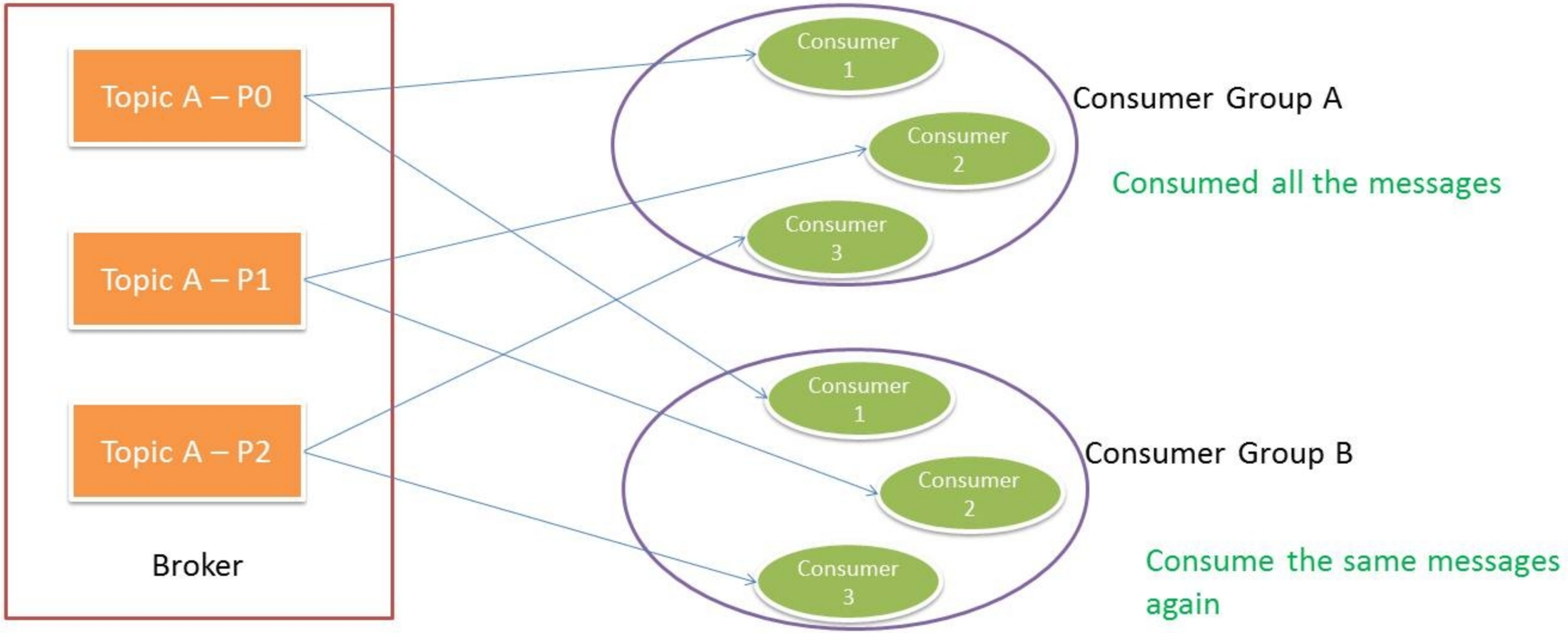


Consumer - Queuing

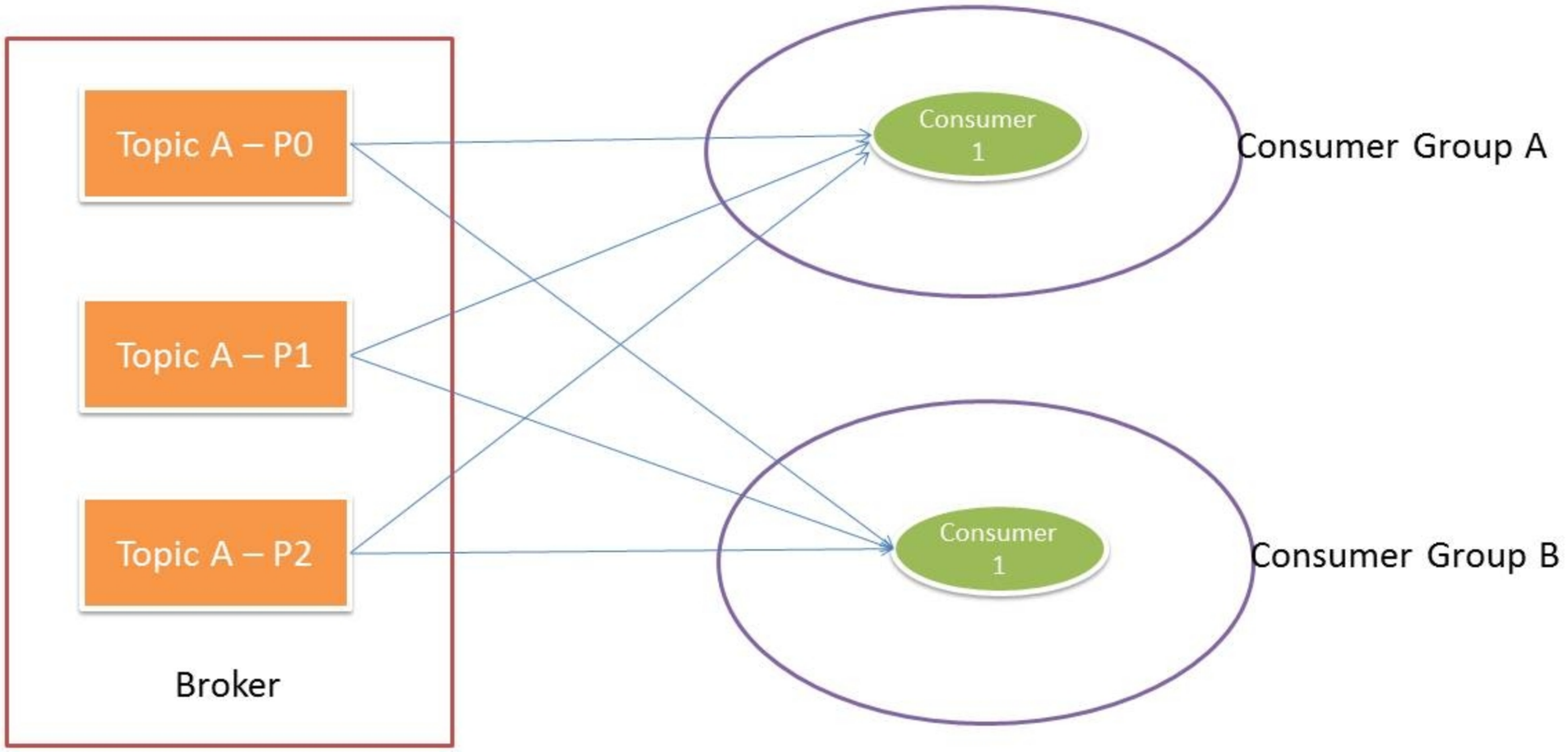


Each message published to a topic is delivered to one consumer instance within each subscribing consumer group.

Consumer Group



Consumer Broadcast



Kafka Console Producer, Consumer – Hands-On

- Push data into Kafka
  - bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test-replication-1
- Pull data from Kafka
  - bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test-replication-1 --from-beginning

Assignment

- Installation of kafka on multiple machines

Kafka Client

- For 0.8.x
  - Producer Daemon
  - Python
  - Go (AKA golang)
  - C
  - C++
  - .net
  - Clojure
  - Ruby
  - Node.js
  - Alternative Java Clients
  - Storm
  - Scala DSL
  - HTTP REST
  - JRuby
  - Perl
  - stdin/stdout
  - Erlang

Producer Java API

- Need to set the following properties:

  metadata.broker.list=broker1:9092,broker2:9092

  serializer.class=kafka.serializer.StringEncoder

  request.required.acks=1

- The property “**metadata.broker.list”** is used to determine the leader partition for each topic.
- The property “**serializer.class**” defines what Serializer to use when preparing the message for transmission to the Broker.
- The property "**request.required.acks**" tells Kafka that you want your Producer to require an acknowledgement from the Broker that the message was received.

Sample Producer Example – Hands on

- API to create topic into Kafka.
  - Sample Example
- Produce data into Kafka
- How data distribute between partition

Consumer Java API

- Need to set the following properties:

  zookeeper.connect=IP1:port,IP2:port

  group.id=consumer_group_1

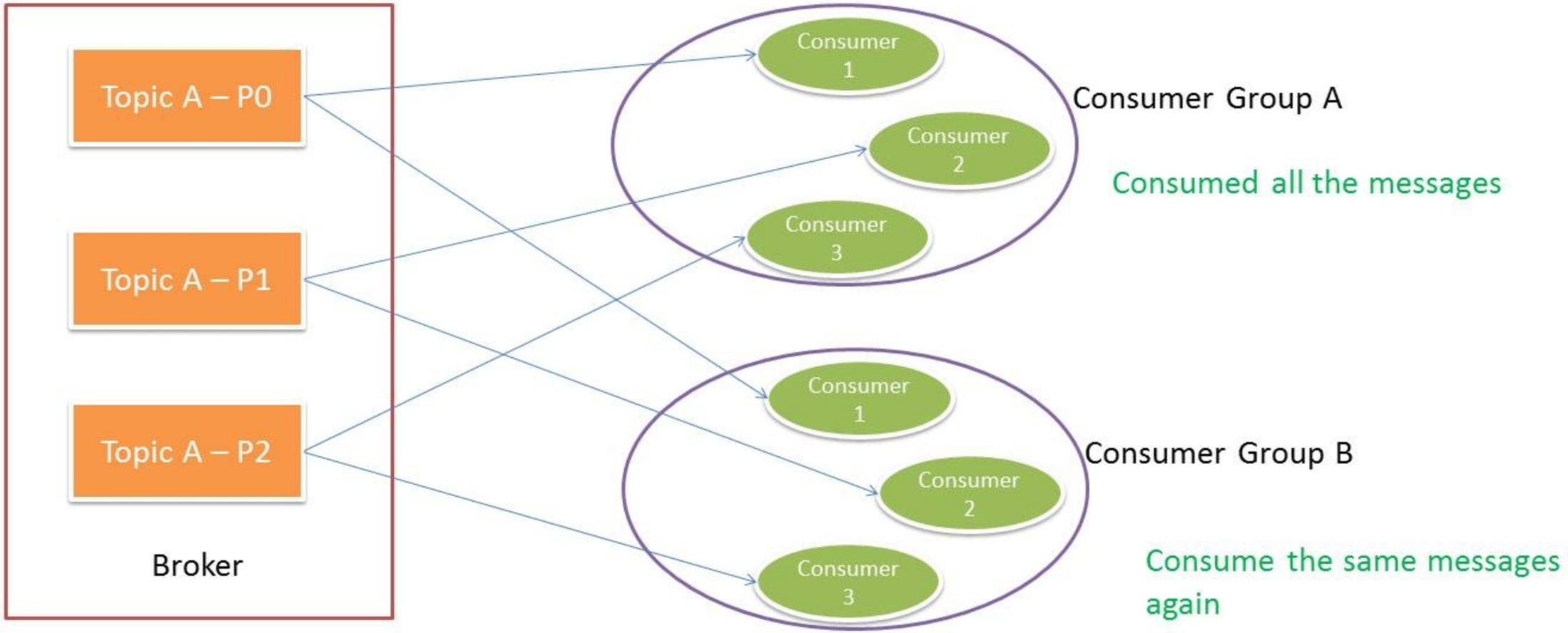
  auto.commit.interval.ms=1000

- The property "**zookeeper.connect**" is used to identifies the zookeeper cluster. Kafka consumer uses ZooKeeper to store offsets of messages consumed for a specific topic and partition.

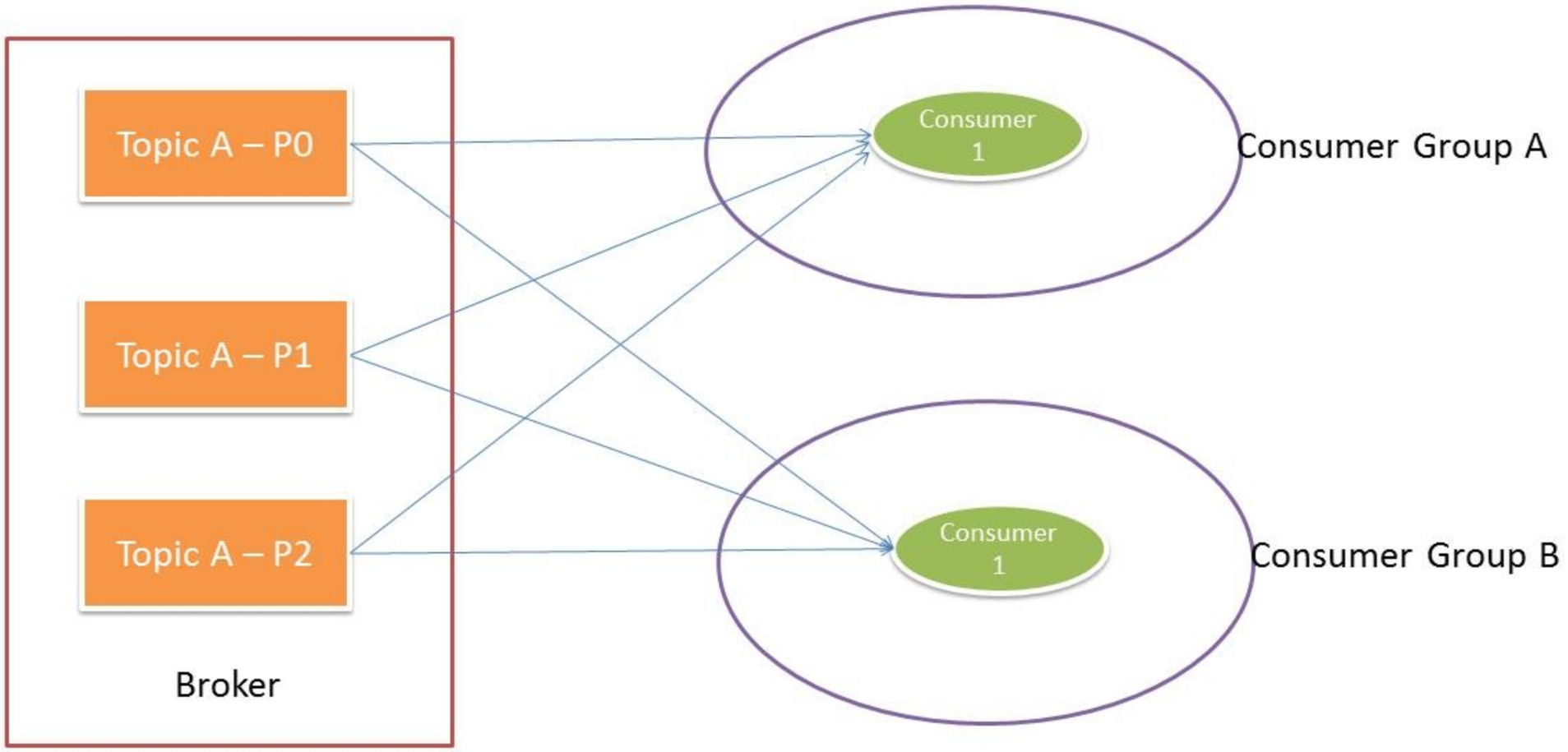
Consumer Java API

- The property "**group.id**" defines the Consumer group.
- The property "**auto.commit.interval.ms**" defines after how many milliseconds the consumed offsets are written to ZooKeeper.

Consumer Group



Consumer Broadcast



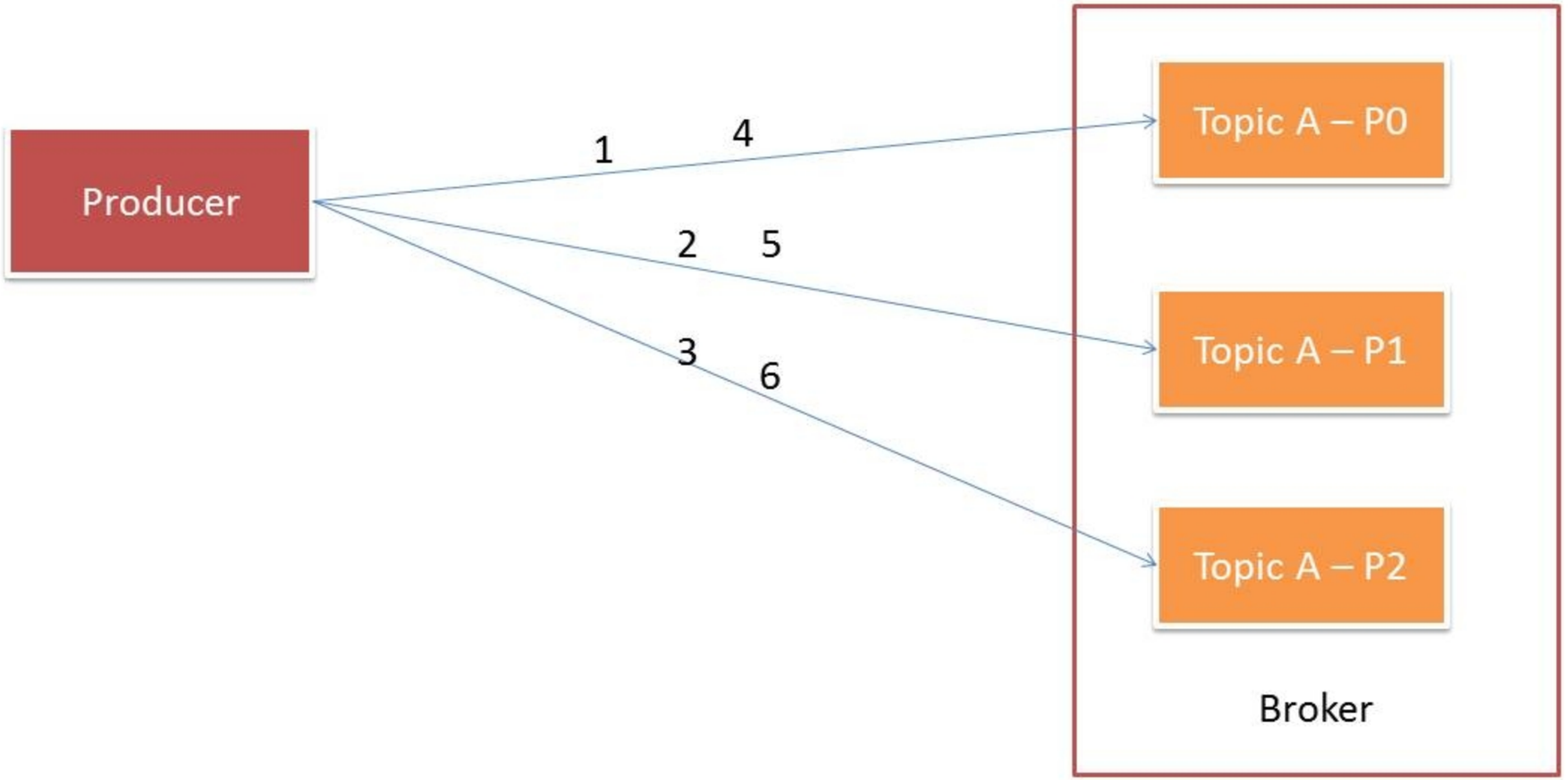
Sample Consumer Example – Hands on

- Java API to read data from Kafka
- How consumer offset is maintain in zookeeper
- How we can read same data again from kafka

Producer Custom Partition

- Need to set the following configuration:
  - partitioner.class=com.dataflair.CustomPartitioner
- The property "**partitioner.class**" defines what class to use to determine which Partition in the Topic the message is to be sent to.
- We can define the partition class by implement the **partition**(Object key , int paritionCount) method of **kafka.producer.Partitioner** interface

Producer – RoundRobinPartition



Distribute data uniformly between all partitions

Sample Custom Partition Example – Hands on

- How we can set partition class in kafka.

DataFlair Web Services Pvt Ltd

+91-8451097879
info@data-flair.com
http://data-flair.com