Data Pipeline with Kafka

Peerapat Asoktummarungsri
AGODA

- Senior Software Engineer Agoda.com
- Contributor Thai Java User Group (THJUG.com)
- Contributor Agile66

AGENDA

- Big Data & Data Pipeline
- Kafka Introduction
- Quick Start
- Monitoring
- Data Pipeline for Search API
- Hadoop integration with Camus

Big Data

MapReduce

Hadoop

+

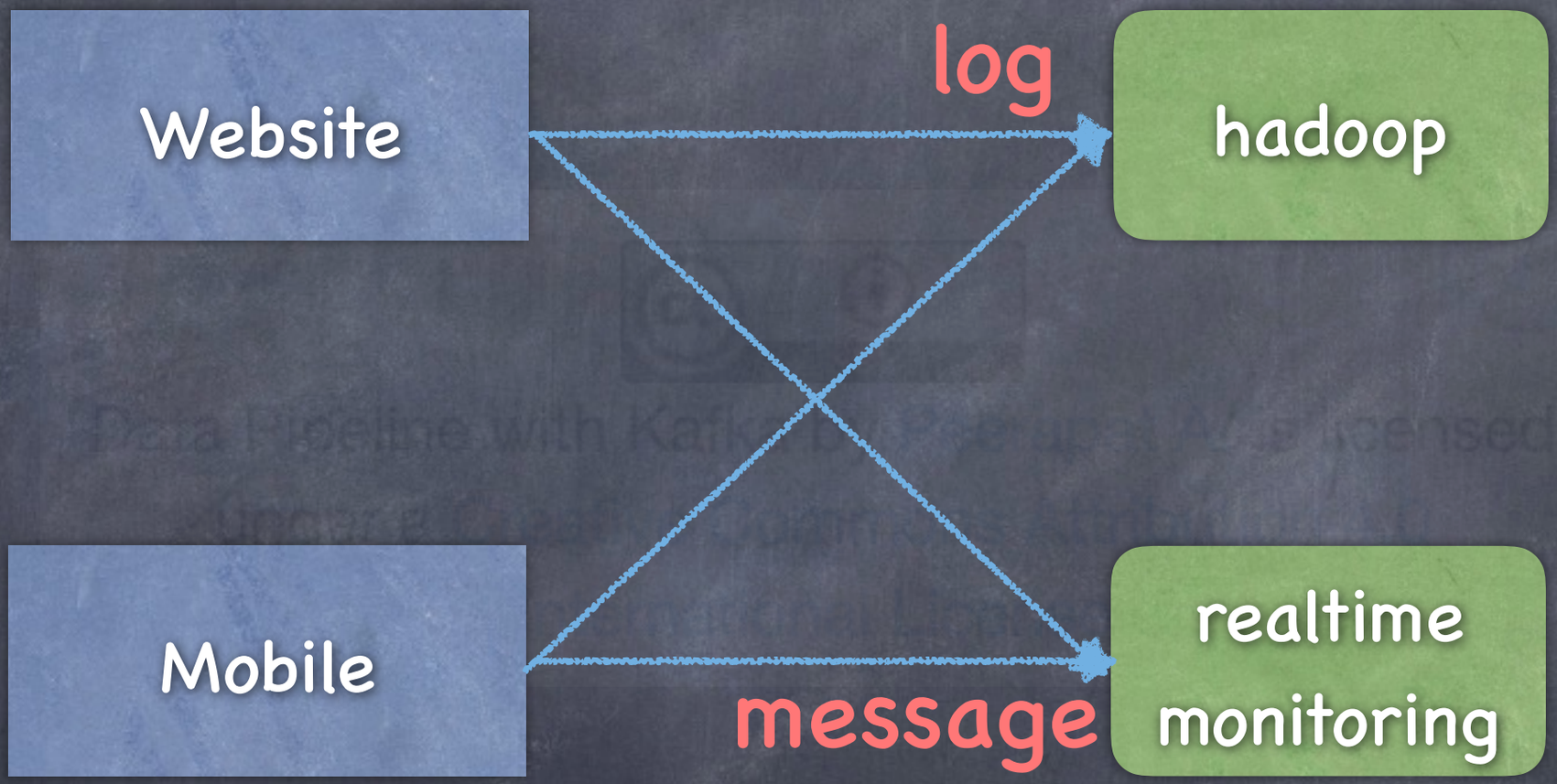
HDFS

Information

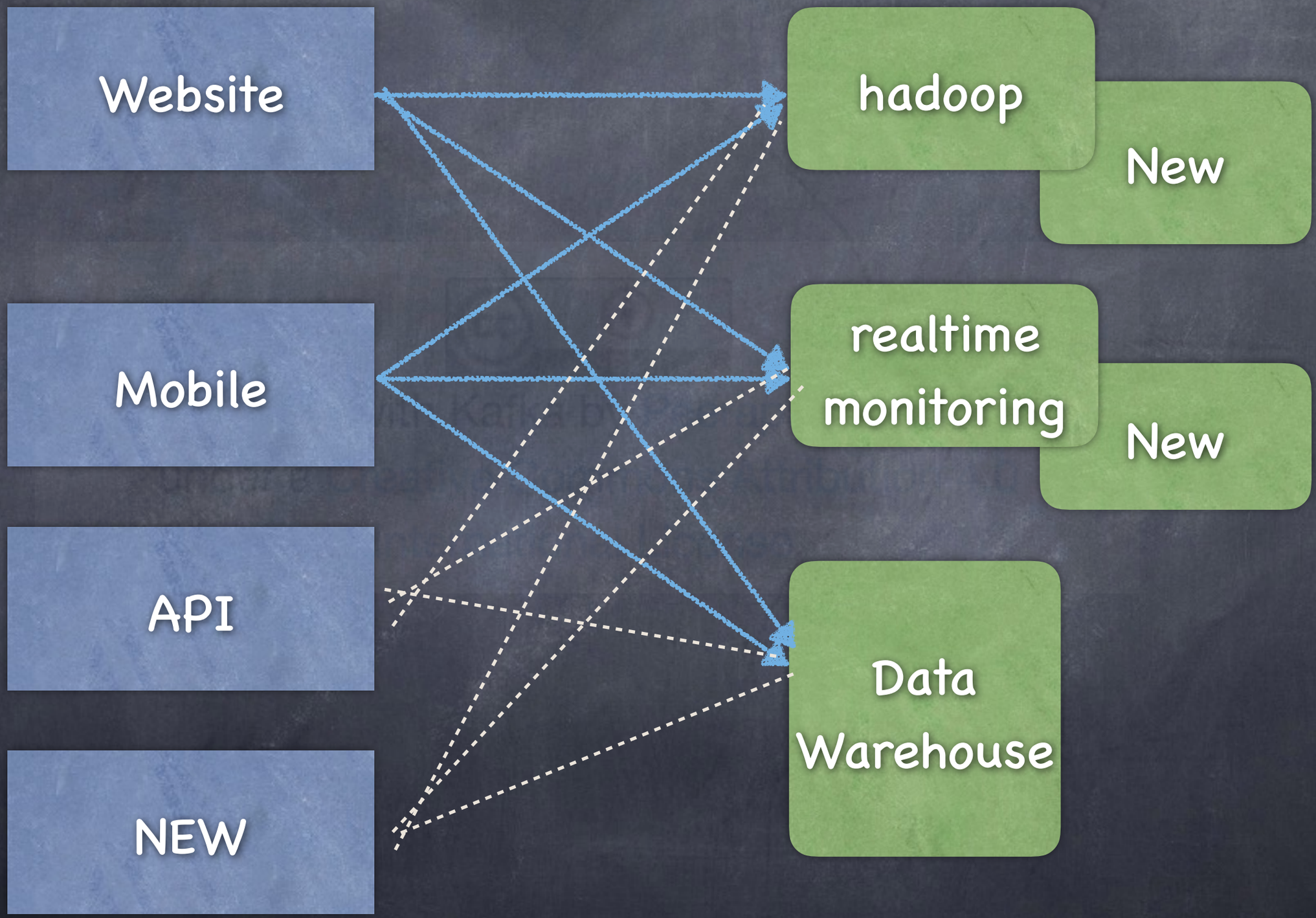
Pipeline

Growth

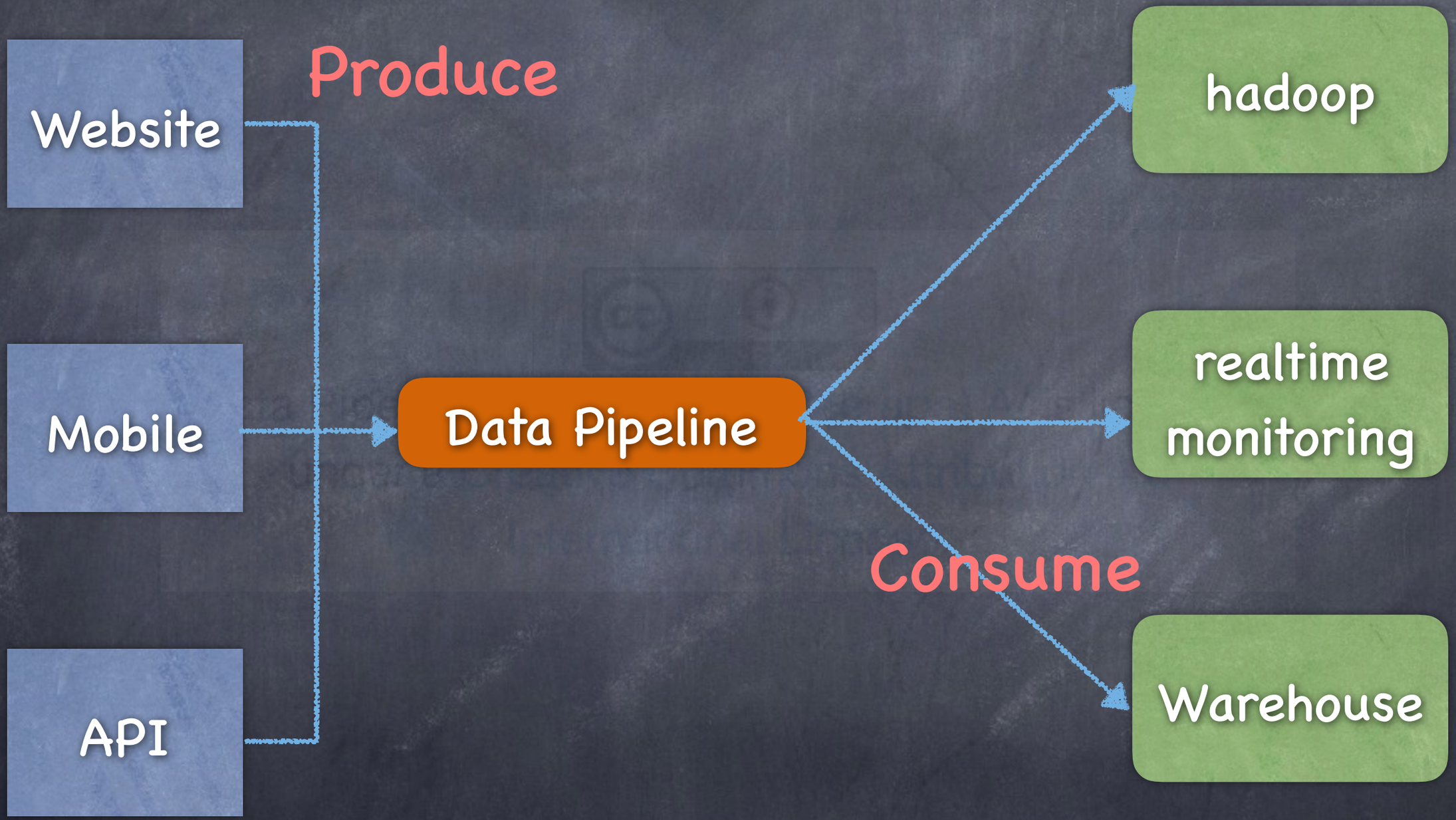
Complex

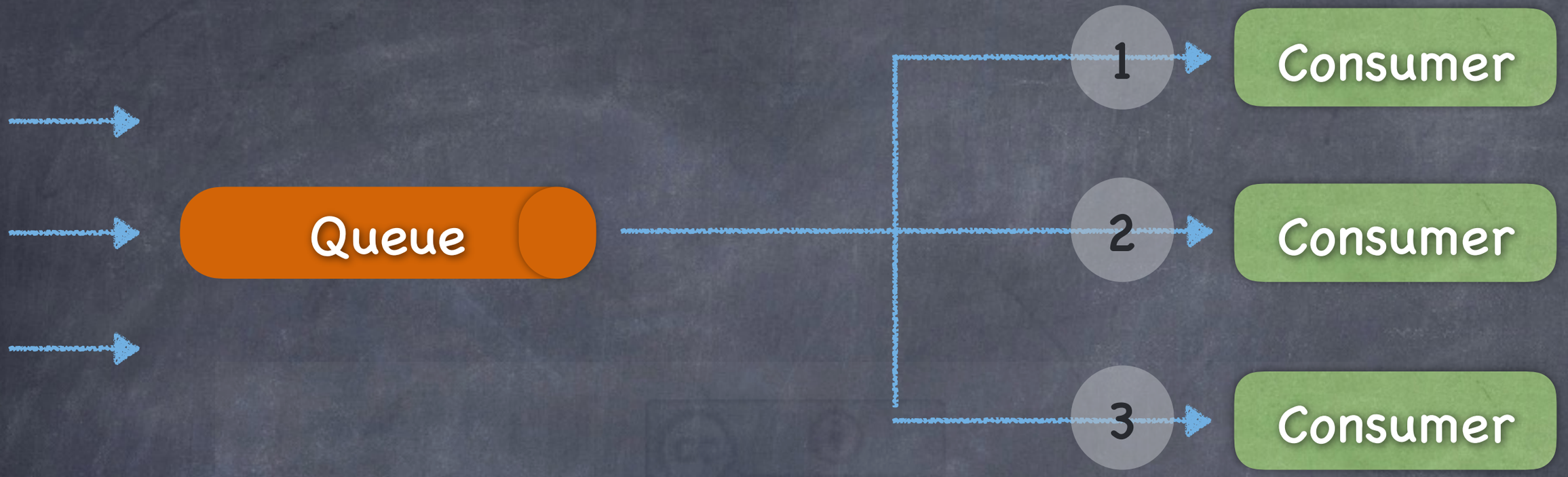


Features becomes the problem

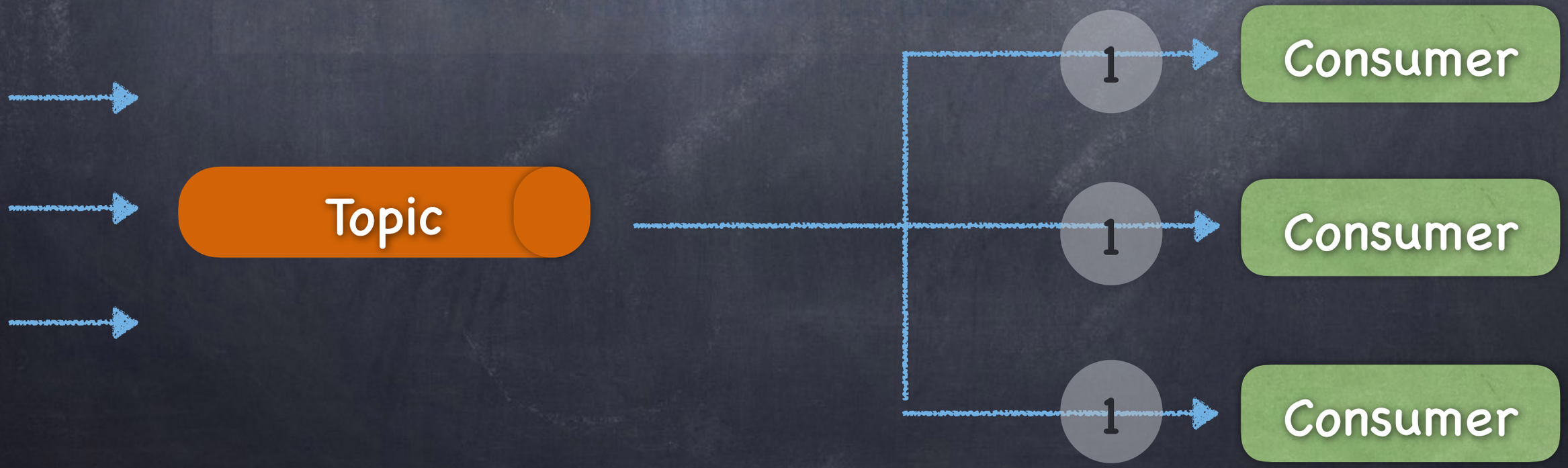


Data Pipeline

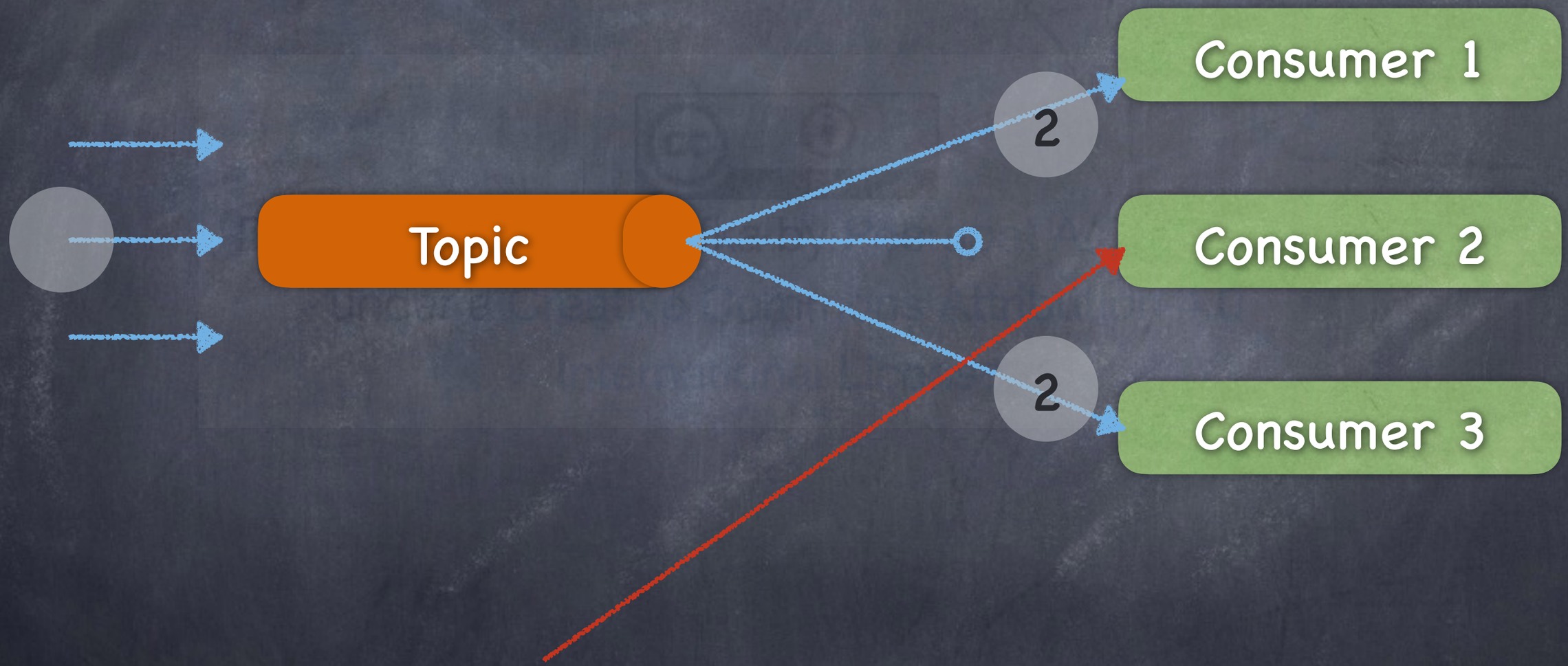




compare



General Topic Implement

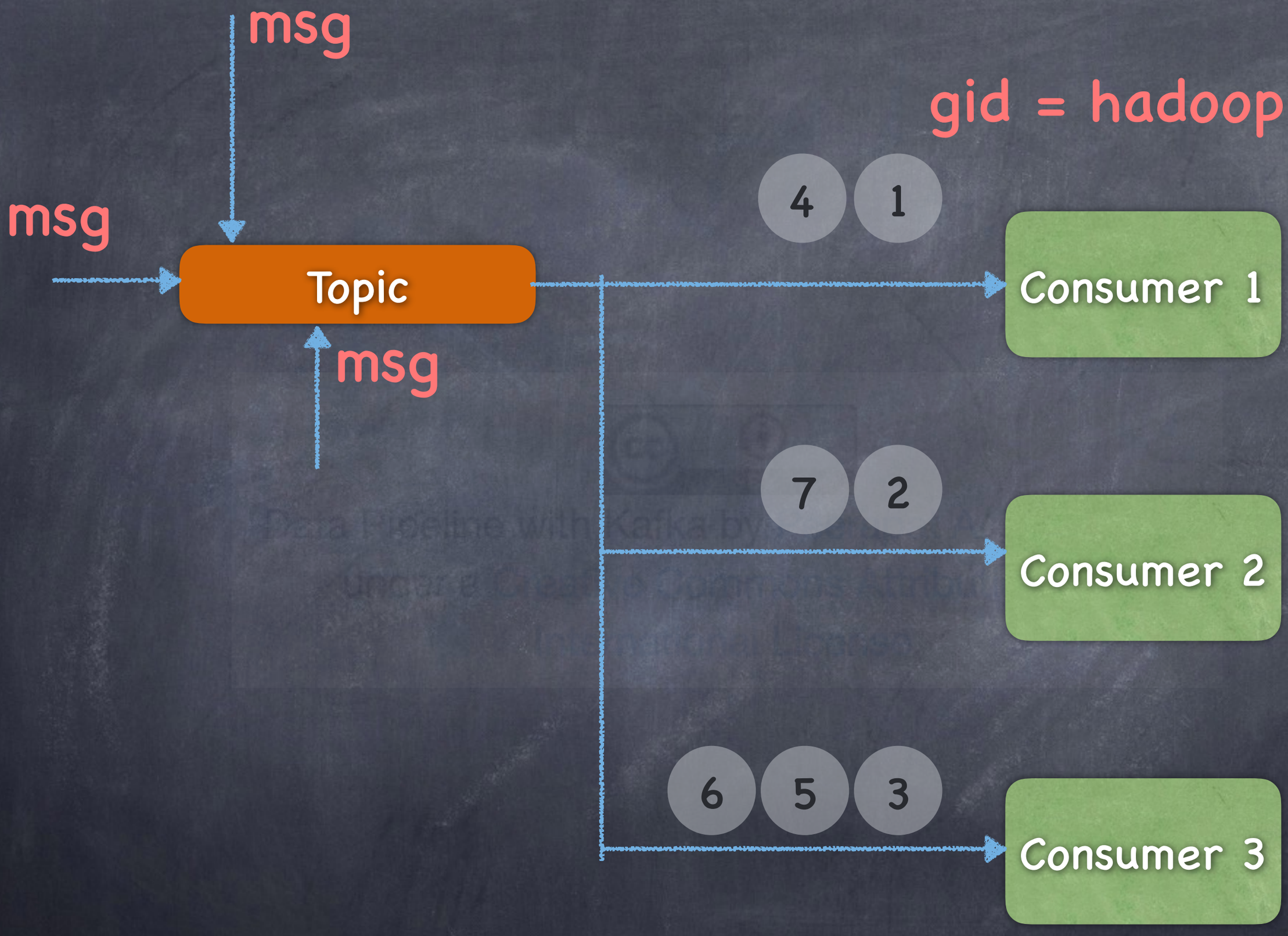


This consumer will lose a message.

Apache Kafka

A high-throughput distributed messaging system.

- **Distributed by Design**
- **Fast**
- **Scalable** - It can be elastically and transparently expanded without downtime.
- **Durable** - Messages are persisted on disk and replicated within the cluster to prevent data loss.



gid = Group ID

msg

msg

Topic

msg

gid = hadoop

3 2 1

hadoop

gid = rtmon

3 2 1

realtime monitoring

gid = warehouse

3 2 1

data warehouse

gid = Group ID

msg

msg

Topic

3

2

1

New Consumer

gid = newconsumer

9

gid = hadoop

hadoop

9

gid = rtmon

realtime monitoring

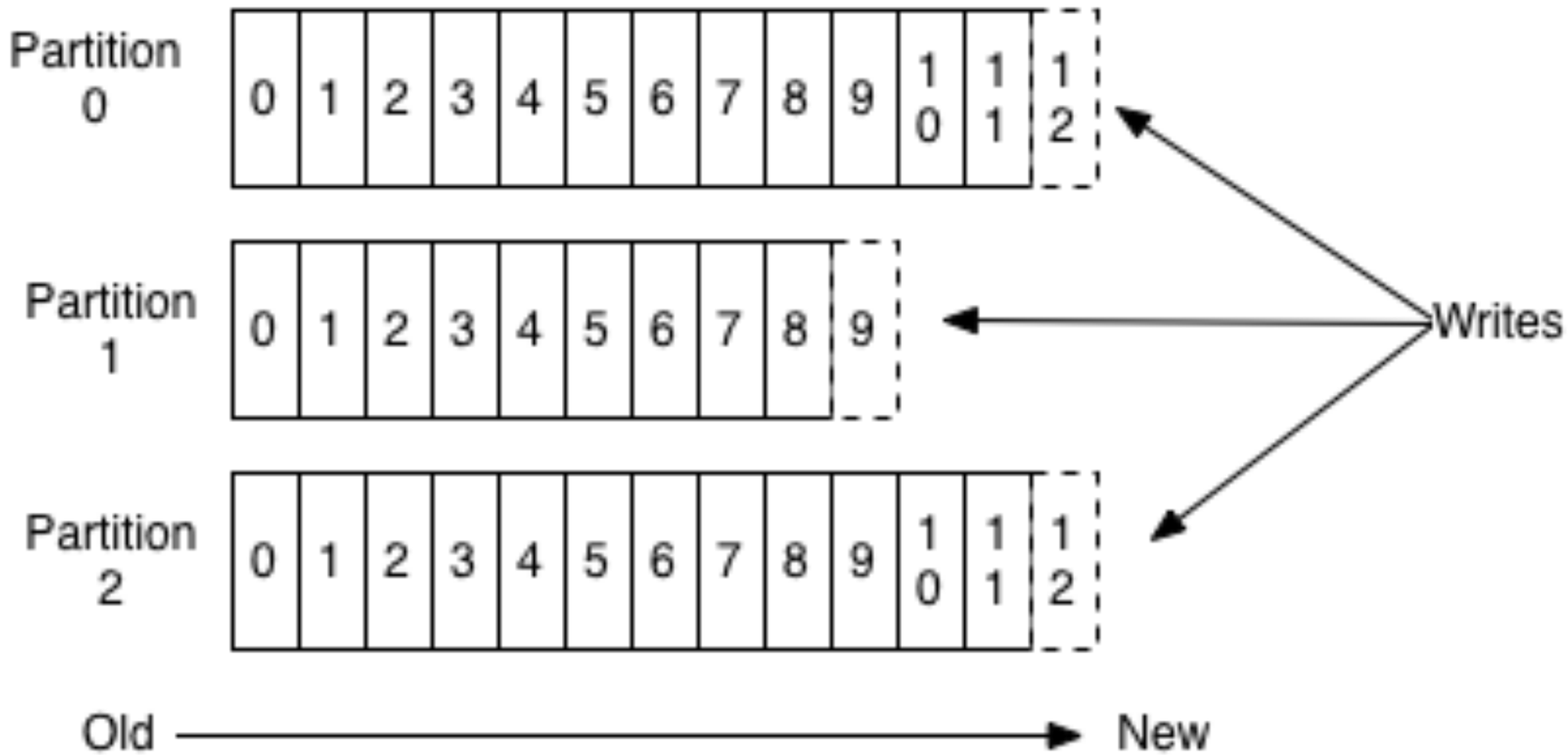
gid = warehouse

9

data warehouse

gid = Group ID

Anatomy of a Topic



Vagrant

- Install Vagrant
- Install Virtual Box
- Clone https://github.com/stealthly/scala-kafka
- vagrant up

once this is done

- Zookeeper will be running 192.168.86.5
- Broker 1 on 192.168.86.10
- All the tests in src/test/scala/* should pass

BREW

- brew update
- brew install zookeeper kafka -y

```
Last login: Tue Dec  8 00:02:24 on ttys000
~ ➜➜➜ zkServer start
JMX enabled by default
Using config: /usr/local/etc/zookeeper/zoo.cfg
Starting zookeeper ... STARTED
~ ➜➜➜ kafka-server-start.sh /usr/local/etc/kafka/server.properties
[2015-12-08 15:31:06,157] INFO Verifying properties (kafka.utils.Ver
[2015-12-08 15:31:06,189] INFO Property broker.id is overridden to 0
[2015-12-08 15:31:06,189] INFO Property log.cleaner.enable is overri
[2015-12-08 15:31:06,190] INFO Property log.dirs is overridden to /u
[2015-12-08 15:31:06,190] INFO Property log.retention.check.interval
[2015-12-08 15:31:06,190] INFO Property log.retention.hours is overr
[2015-12-08 15:31:06,190] INFO Property log.segment.bytes is overrid
[2015-12-08 15:31:06,190] INFO Property num.io.threads is overridden
[2015-12-08 15:31:06,191] INFO Property num.network.threads is overr
```

Some Kafka Config

```
# The id of the broker. This must be set to a unique integer for each broker.
broker.id=0

# The port the socket server listens on
port=9092

# Zookeeper connection string (see zookeeper docs for details).
zookeeper.connect=localhost:2181

# Timeout in ms for connecting to zookeeper
zookeeper.connection.timeout.ms=6000

# The minimum age of a log file to be eligible for deletion
log.retention.hours=168
```

Kafka @ Linkedin (2013)

- 10 billion message writes per day
- 55 billion messages delivered to real-time consumers
- 367 topics that cover both user activity topics and operational data
- the largest of which adds an average of 92GB per day of batch-compressed messages
- Messages are kept for 7 days, and these average at about 9.5 TB of compressed messages across all topics.

KafkaOffsetMonitor

1 Jar file, KafkaOffsetMonitor-assembly-0.2.1.jar

```
java -cp KafkaOffsetMonitor-assembly-0.2.1.jar \
     com.quantifind.kafka.offsetapp.OffsetGetterWeb \
--zk localhost \
--port 8080 \
--refresh 10.seconds \
--retain 2.days
```

Download KafkaOffsetMonitor from Github
https://github.com/quantifind/KafkaOffsetMonitor

Brokers

id	host	port
0	10.59.13.9	9092

Consumer Offsets

Topic	Partition	Offset	logSize	Lag	Owner	Created	Last Seen
test		2352	2452	100			
	0	365	375	10	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-0	8 days ago	a minute ago
	1	212	219	7	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-1	8 days ago	a minute ago
	2	219	237	18	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-2	8 days ago	a minute ago
	3	228	238	10	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-3	8 days ago	a minute ago
	4	236	245	9	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-4	8 days ago	a minute ago
	5	226	235	9	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-5	8 days ago	a minute ago
	6	217	225	8	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-6	8 days ago	a minute ago
	7	219	230	11	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-7	8 days ago	a minute ago
	8	224	232	8	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-8	8 days ago	a minute ago
	9	206	216	10	akka-kafka_C02NC1XCG3QT-1449560407280-b9520992-9	8 days ago	a minute ago

Groups List / akka-kafka / test

1.667 msg/s

over a minute.

1.667 msg/s

over a minute.

```scala
import java.util.Properties

import com.typesafe.scalalogging.Logger
import kafka.producer.{KeyedMessage, Producer, ProducerConfig}
import org.slf4j.LoggerFactory

/**
  * @author Peerapat A
  */
class KafkaProducer[T](servers:String, topic: String, serializer: Option[String], requireAck: Boolean) {
  import KafkaProducer._

  private lazy val props = new Properties()
  props.put("metadata.broker.list", servers)
  props.put("serializer.class", serializeClass)
  props.put("request.required.acks", if (requireAck) "1" else "0")

  private lazy val config = new ProducerConfig(props)
  val producer = new Producer[String, T](config)

  log.info(s"Created connection to $servers")

  def send(msg: T, partition: String = "0"): Unit = {
    val data = new KeyedMessage[String, T](topic, partition, msg)
    producer.send(data)
  }

  private def serializeClass: String = serializer.getOrElse("kafka.serializer.StringEncoder")

}
```

```scala
import org.scalatest.FunSuite

import scala.util.Random

class KafkaProducerSpec extends FunSuite {

  private val rand = new Random()
  private val kafka = new KafkaProducer[String]("localhost:9092", "test", None, true)

  test("testSend with no Exception.") {
    for (a <- 1 to 100) {
      val p = rand.nextInt(10)
      kafka.send(s"P$p - ${System.currentTimeMillis}", p.toString)
    }
  }

}
```

```scala
import akka.actor.Actor
import com.sclasen.akka.kafka.StreamFSM
import com.typesafe.scalalogging.Logger
import org.slf4j.LoggerFactory

import scala.util.Random

/**
  * @author Peerapat A
  */
class EchoActor extends Actor {
  import EchoActor._

  override def receive = {
    case msg: String => {
      log.info(s"$msg, Finish")

      sender ! StreamFSM.Processed
    }
  }

}
```

```scala
import akka.actor.{ActorRef, ActorSystem, Props}
import akka.routing.RoundRobinPool
import com.sclasen.akka.kafka.{AkkaConsumer, AkkaConsumerProps}
import thjug.actor.{EchoActor, Distributer}
import thjug.decoder.StringDecoder

/**
  * @author Peerapat A
  */
object KafkaConsumer extends App {

  val system = ActorSystem("cusco")

  val distributer = system.actorOf(Props(new Distributer(system)), name = "distributer")

  val actor: ActorRef =
    system.actorOf(RoundRobinPool(10).props(Props[EchoActor]), "echoactor")

  val consumerProps = AkkaConsumerProps.forSystem(
    system = system,
    zkConnect = "localhost:2181",
    topic = "test",
    group = "akka-kafka",
    streams = 10,
    keyDecoder = StringDecoder(),
    msgDecoder = StringDecoder(),
    receiver = actor
  )

  val consumer = new AkkaConsumer(consumerProps)
  consumer.start

}
```

Produce Change

API

CHANGE

Kafka

Price & Inventory Consumer

Calculate Price

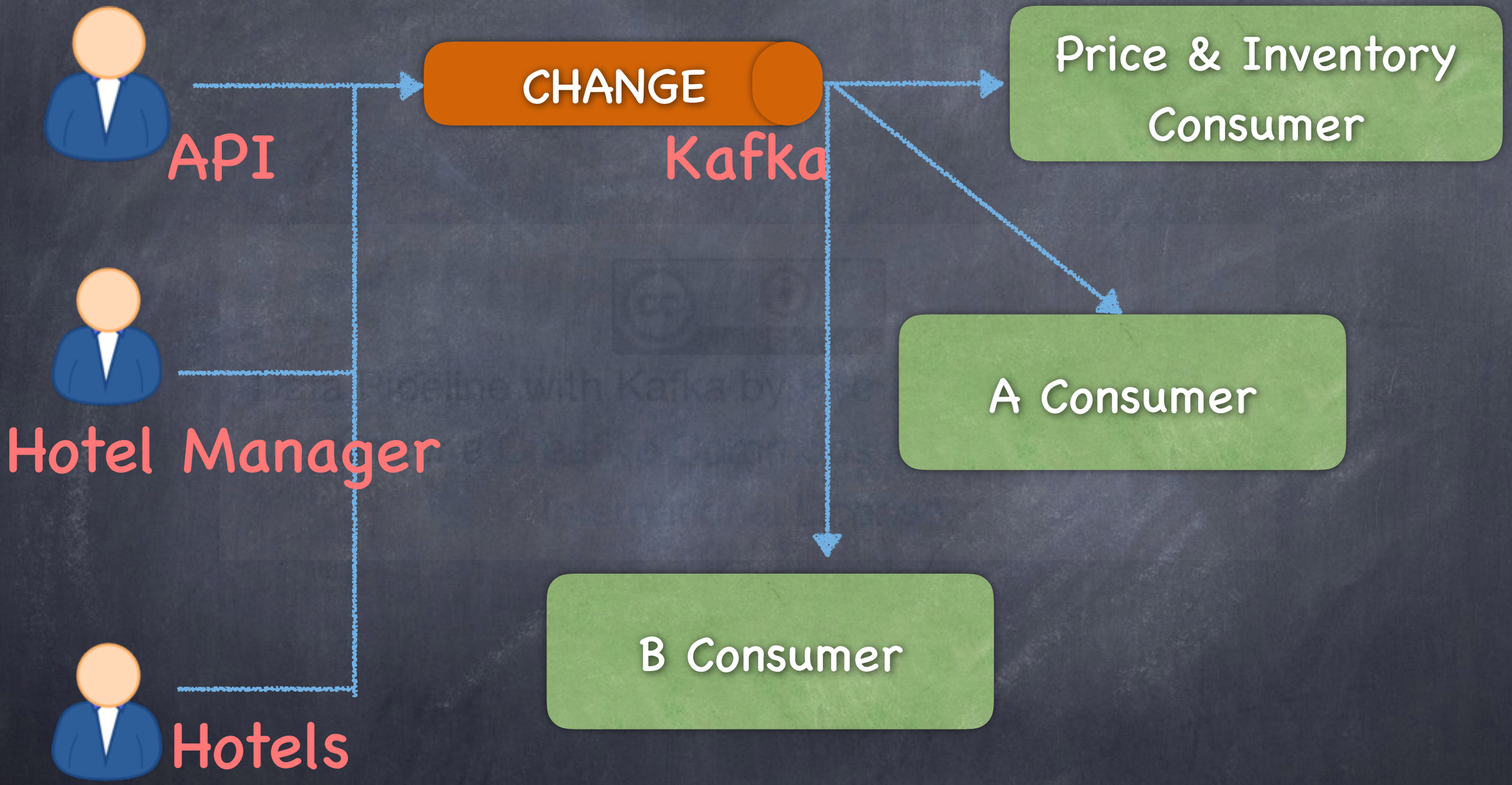
Hotel Manager

Hotels

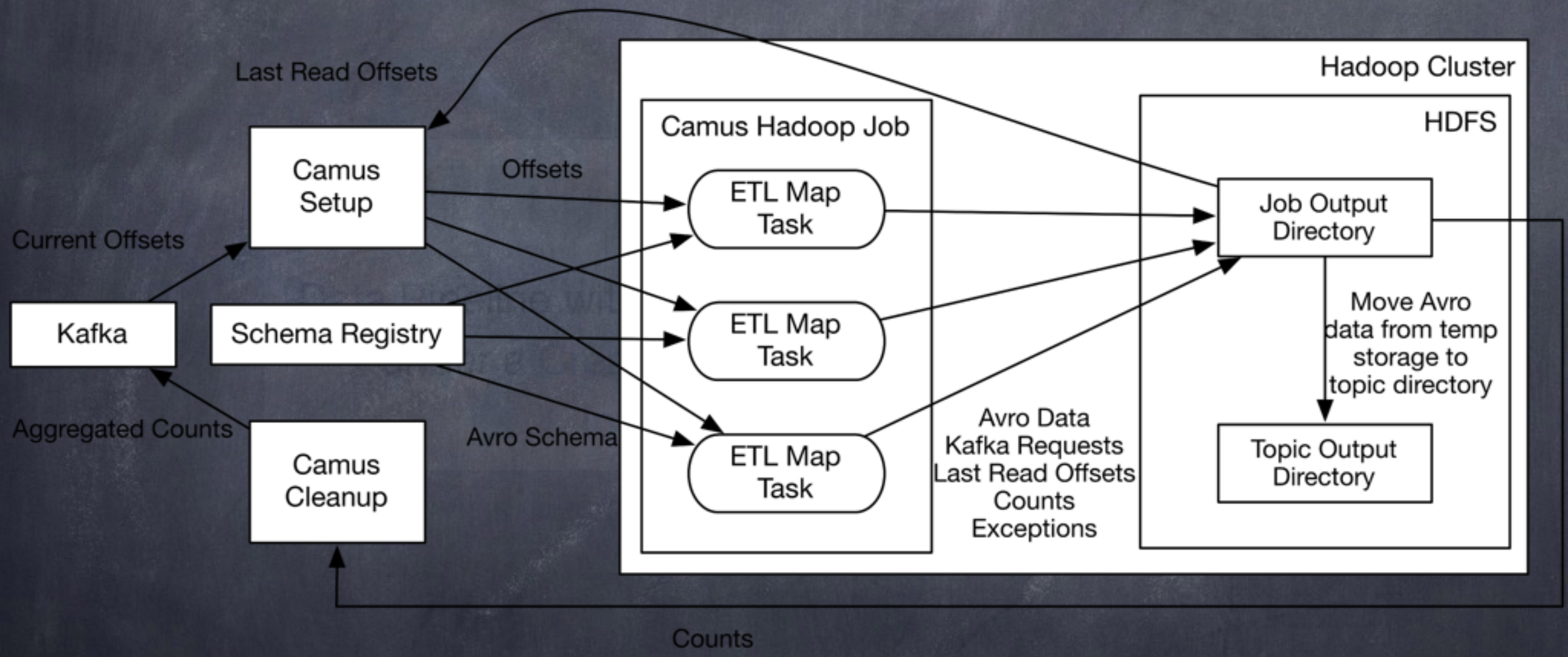
Cassandra

Search API

HTTP



Camus



Slide available here

http://www.slideshare.net/nuboat

Sourcecode available here

https://github.com/nuboat/akkakafkaexam

REFERENCES

- http://www.slideshare.net/charmalloc/developingwithapachekafka-29910685
- http://www.infoq.com/articles/apache-kafka
- http://kafka.apache.org/
- https://github.com/stealthly/scala-kafka
- https://github.com/quantifind/KafkaOffsetMonitor

Q & A