卷皮
JUANPI.COM

Kakfa培训与实践

阿昌.何伟昌
2016.03.09

Kafka?

- http://kafka.apache.org
- 来源于linkedin, 2012年成为apache顶级项目
- 大部由scala编写, 一些java
- 9个核心开发者，社区比较活跃

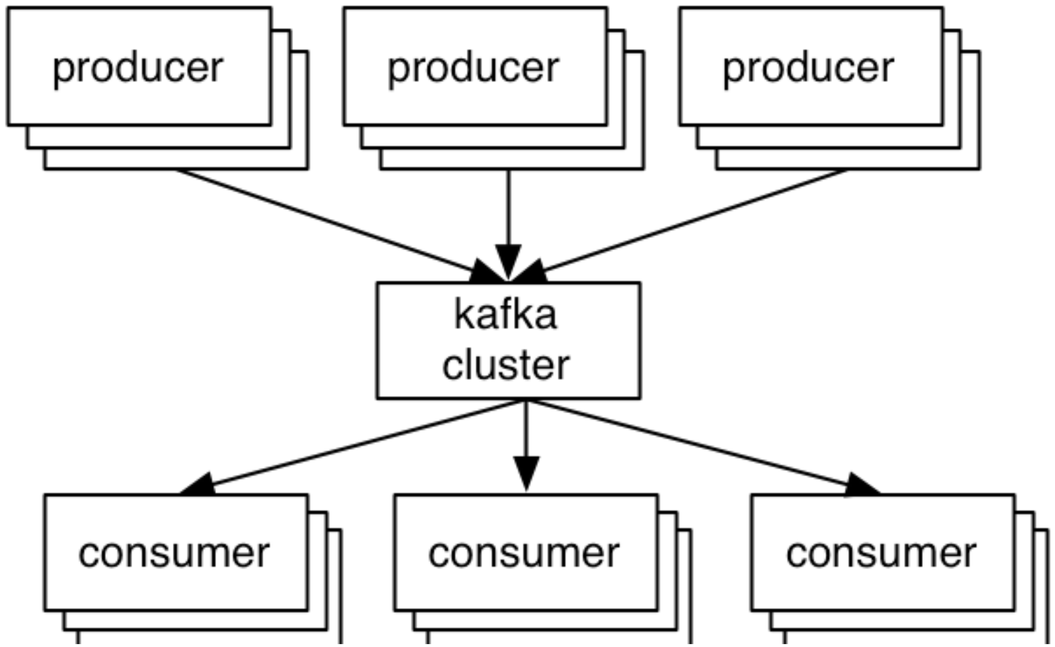
Kafka @ juanpi

- Proxy与Processor之间解耦，Processor负载均衡。
- 版本0.9.0.1，6个broker(32G内存，SAS盘)
- 40+ topics, 日均处理消息量6000W+

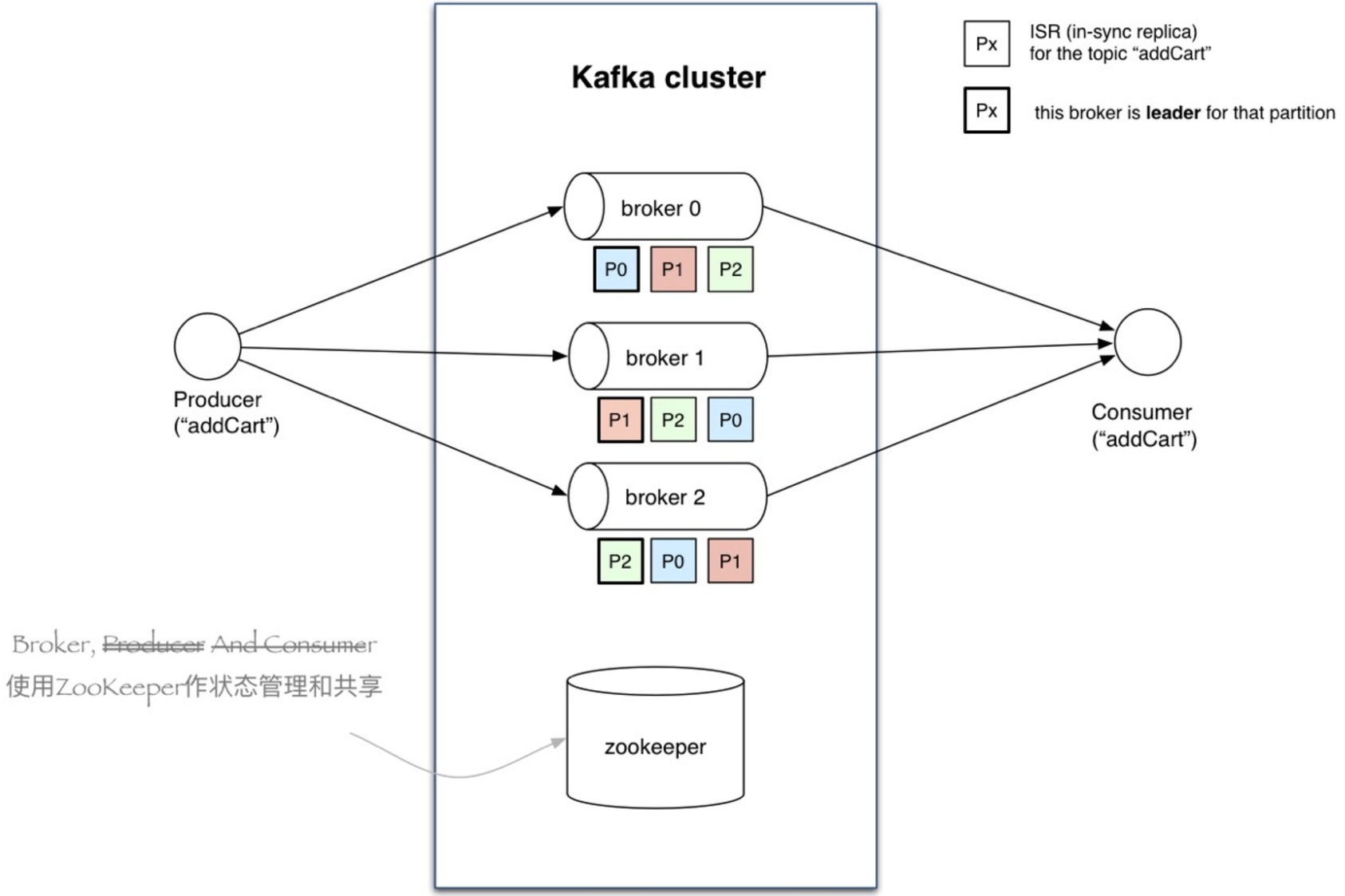
基础培训

术语

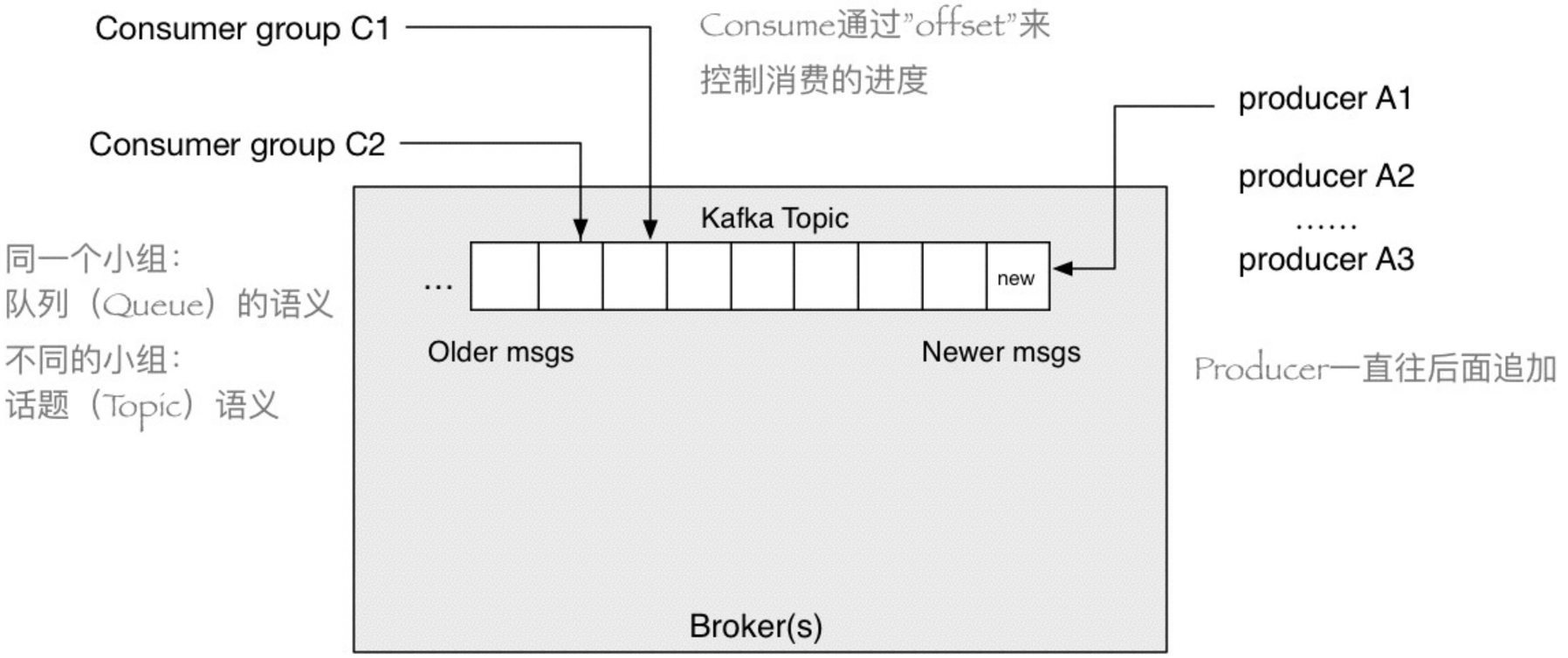
- 角色
  - **Producers** 写数据到 **Brokers**
  - **Consumers** 从 **Brokers** 读数据
  - 都是分布式的
- 数据
  - 数据储存在 **topics**
  - **topics** 分布在 互为**replicas**（副本）的 **partitions**



第一印象

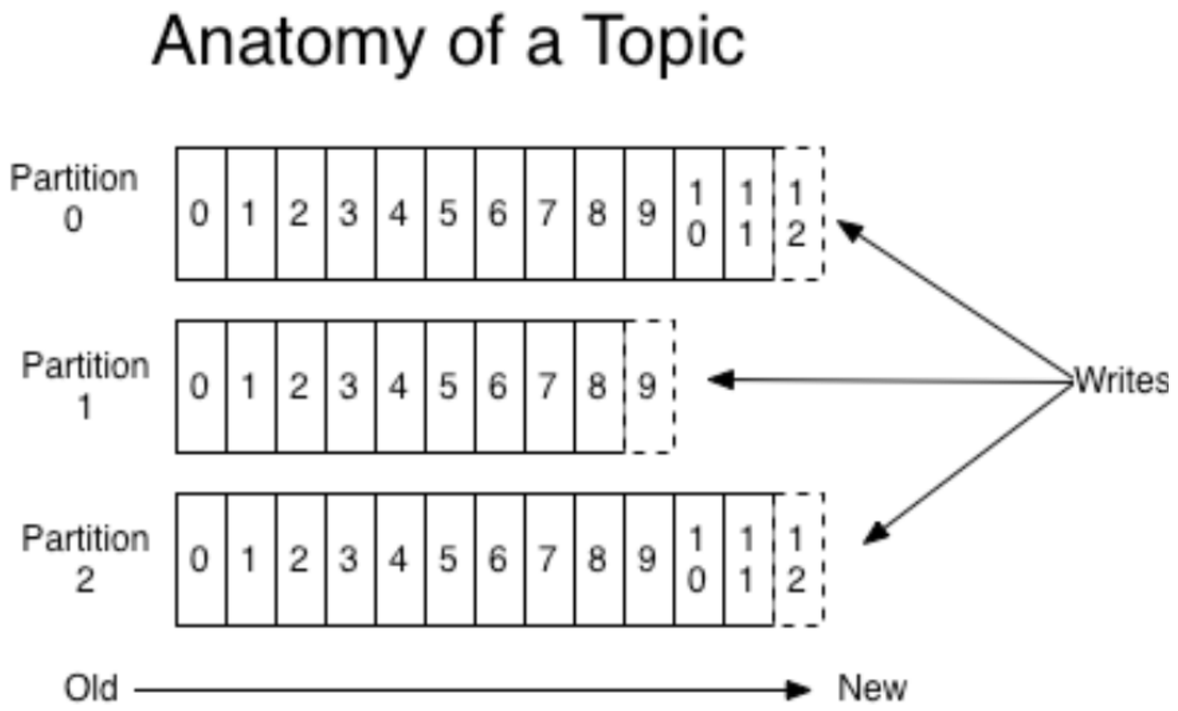

Kafka cluster
Px
ISR (in-sync replica)
for the topic “addCart”
Px
this broker is leader for that partition
broker 0
P0
P1
P2
broker 1
P1
P2
P0
broker 2
P2
P0
P1
Producer
(“addCart”)
Consumer
(“addCart”)
zookeeper
Broker, ~~Producer~~ ~~And Consumer~~
使用ZooKeeper作状态管理和共享

Topics


Consumer group C1
Consume通过"offset"来
控制消费的进度
Consumer group C2
producer A1
producer A2
……
producer A3
Kafka Topic
new
同一个小组：
队列（Queue）的语义
不同的小组：
话题（Topic）语义
Older msgs
Newer msgs
Producer一直往后面追加
Broker(s)

Partitions

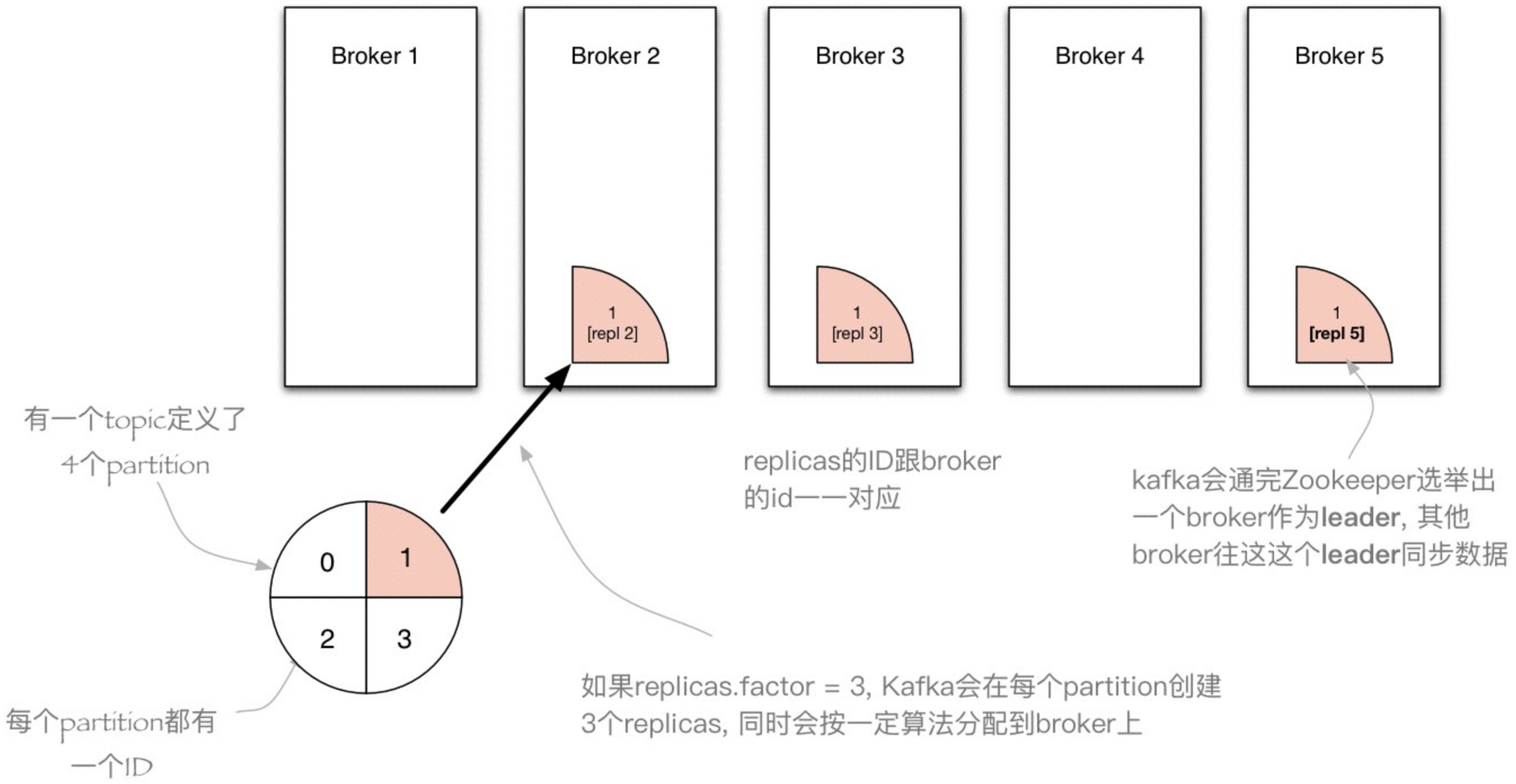
- 一个Topic由多个Partitions组成
- Partition: **有序** + **不可变** 的消息不断地往后追加
- 每一个Parition相当一个『文件夹』



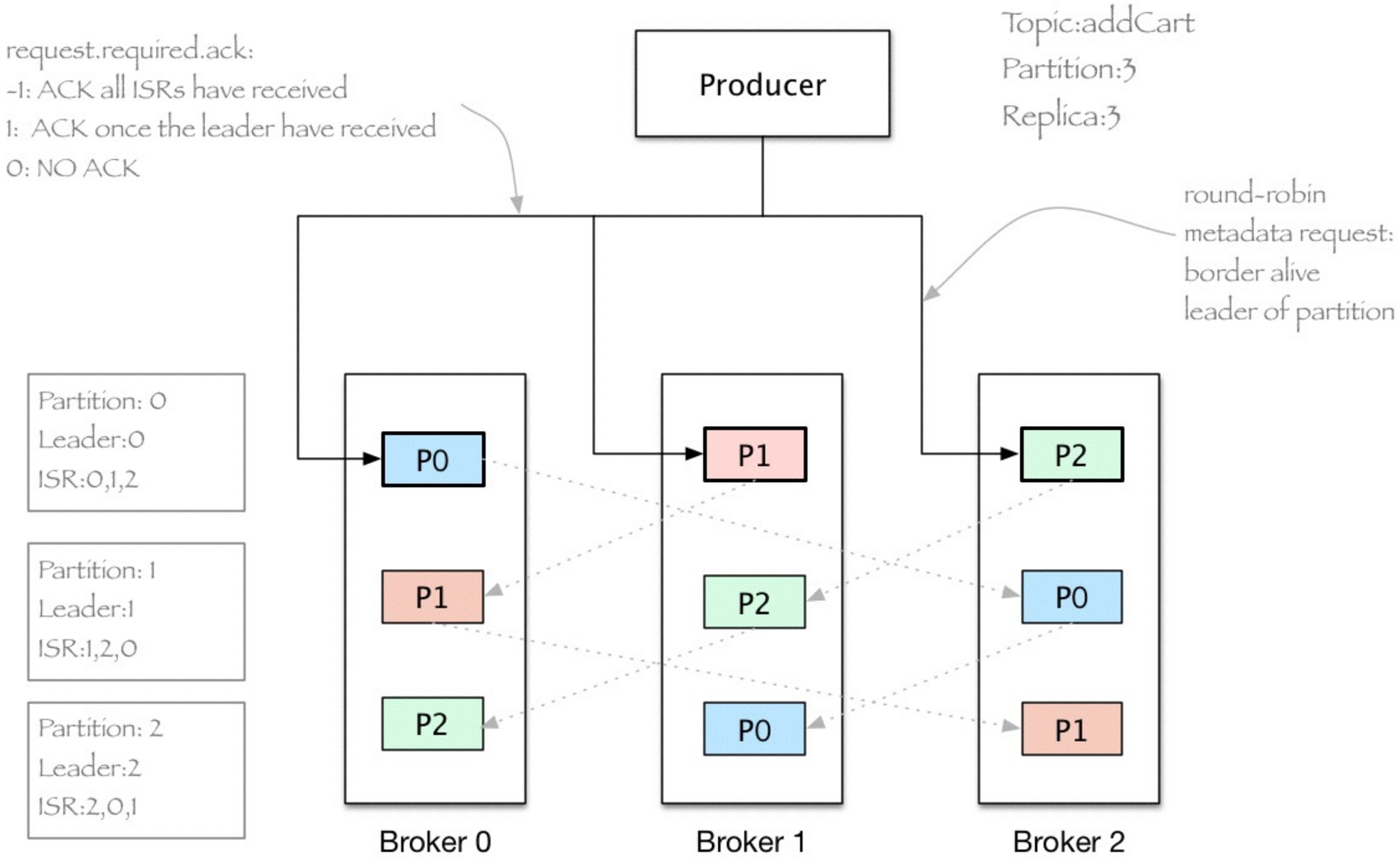
Replicas

- Replicas: Partition的『备份』，或者叫副本
- 目的：防止数据丢失(硬盘故障) & 高可用
- 失败容忍性：*replication-factor - 1*，如replication-factor=2，可容忍1个broker 宕机

Topics vs. Partition vs. Replicas

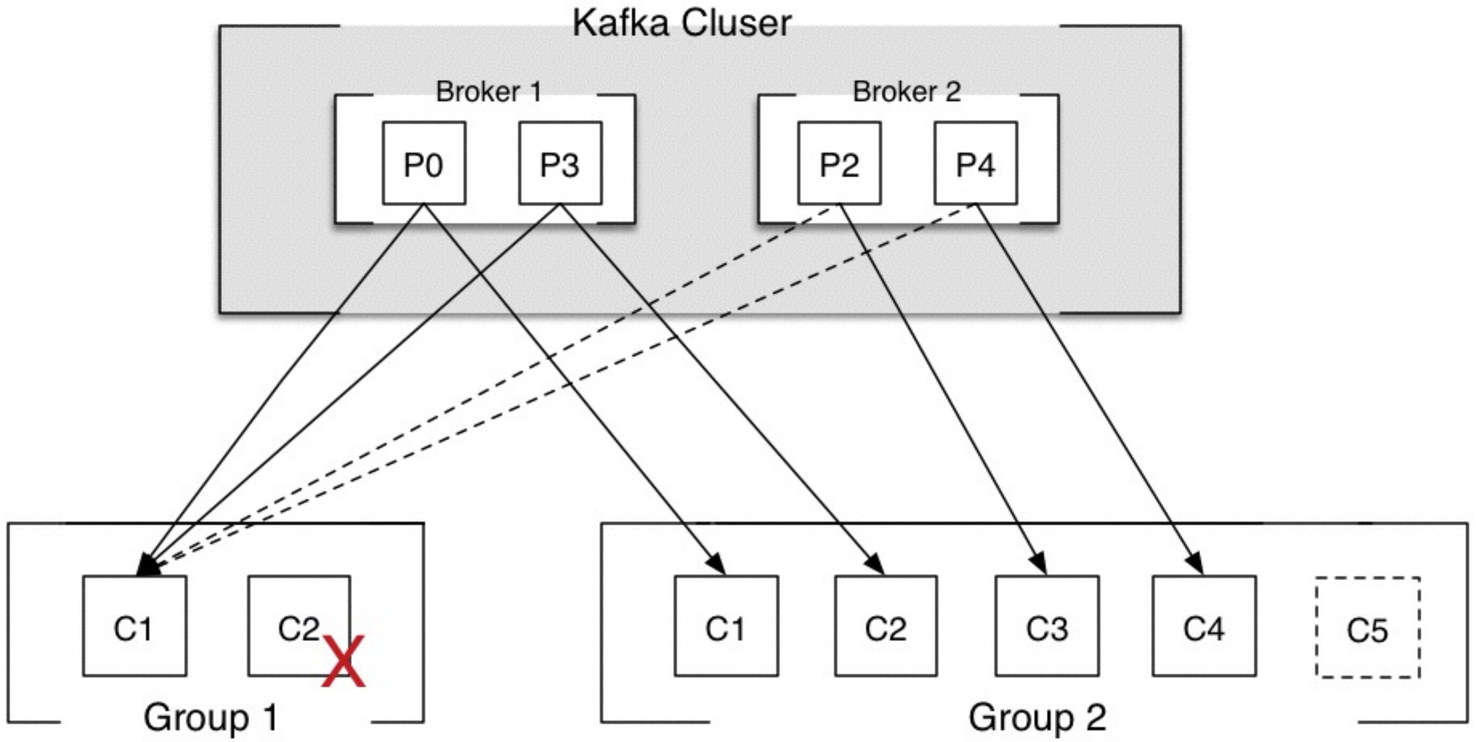


Producer


request.required.ack:
-1: ACK all ISRs have received
1: ACK once the leader have received
0: NO ACK
Producer
Topic:addCart
Partition:3
Replica:3
round-robin
metadata request:
border alive
leader of partition
Partition: 0
Leader:0
ISR:0,1,2
Partition: 1
Leader:1
ISR:1,2,0
Partition: 2
Leader:2
ISR:2,0,1
P0
P1
P2
P1
P2
P0
P2
P0
P1
Broker 0
Broker 1
Broker 2

Consumer

- Topic的Parttions都是可配的，只能增加不能减小。
- Partions决定Consumer的并行度。



消息传递语义

- 最多一次。消息不会重复，故障情况下会导致消息丢失。
- 最少一次。消息不会丢失，故障情况下会重复。（Kafka默认）
- 仅仅一次。消息不重复也不会丢失。（Kafka 0.10.0.0开始支持，2016年Q2发布）

一些实践

『最少一次』的副作用

- 消费者处理过慢可能会导致重复消费，原因为处理时间大于session timeout时间，offset没有提交就rebalancing。
- 0.9.x 暂时可通过增大session timeout时间，或者减小分区拉取值（max.partition.fetch.bytes默认为1M），但会影响吞量，而且以bytes为单位也无法评估消息的数量。0.10.x将会增加max.poll.records参数。
- Kafka社区的讨论 https://cwiki.apache.org/confluence/pages/viewpage.action?pageId=61333789
- 结论：不应该阻塞kafka client的线程。

消息好像阻塞了？

- 通过kafka-consumer-group.sh来查看消息消费情况。
- hight-level consumer可能阻塞的情况：1）消息大于fetch.size（默认为1M）2）应用阻塞代码（如，异常没有捕获）3）consumer rebalancing 失败，会看到ConsumerRebalanceFailedException。
- 总结：不应该阻塞kafka client的线程，时刻关注rebalancing的情况。

Kafka 会丢消息吗?

- Broker(持久)：0.8.x后引入了副本机制。
- Producer： 三种ACK级别
- Conumser: 最少一次 和 仅仅一次 （未来）消息传递语义
- 总结：在卷皮，replication-factor = 2，ack = 1, conumser 使用high-level api。

消息能有序吗?

- 单分区有序

- 具体做法：1）producer send指定分区 。2）consumer 指定分区消费，需要自己实现fail-over 3）replication-factor设置为N（N为broker数量），保证broker n-1 宕机容忍性。

- Apache Samza = Kafka + Yarn + SamzaJob

Topic的Partitions权衡

- partitions决定comsuer并行度。
- partitions 只可以增加，不能减小。
- 每个partition都在ZooKeeper上注册。
- 越多partition越久Leader fail-over时间。
- 总结：根据应用场景制定partitions，关注rebalancing情况，关注ZooKepper情况。

Kafka 在未来轩辕?

- Kafka：把日志作为服务
- 日志：**有序 + 不变性**
- 事件：某个时间点的状态，事件按时间先后顺序产生（有序），只能新增不能删除或修改（不变性），从这个角度看日志和事件并没有什么区别。

后续培训计划

- Kafka数据存储
- Broker的HA
- 0.9后的NEW Consumer的设计
- Kafka高吞吐的秘密
- 其他

作业

- 实现一个消息有序的例子。
- 创建一个topic，partition为3。
- 假设现有a, b , c 三个用户，各每发十条消息。让这些消息有序分布到3个consumer。

2016.03.09