开放消息服务（ONS）原理与实践

沈询
中间件

自我介绍

- 花名 沈询 @ 阿里巴巴中间件
- 新浪微博: 淘宝沈询_WhisperXD
- 阿里分布式数据库DRDS,TDDL负责人
- 阿里分布式消息服务ONS(Notify,MetaQ) 负责人
- 加群聊架构：326140964

Open Notification Service(ONS)

- ONS的应用场景
- ONS的设计思路
- ONS的关键概念
- 消息乱序问题
- 消息重复问题
- 分布式事务与ONS

ONS的应用场景

ONS的应用场景

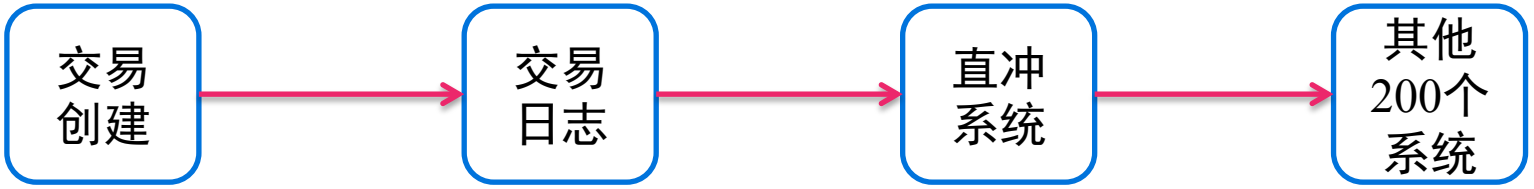
- 异步
- 解耦
- 最终一致
- 并行

ONS的应用场景

- 过年了，拜年发微信
  - 普通青年：编辑微信，群发给所有人
  - 文艺青年：编辑微信，交给美腻秘书发送，自己去~~~~~~
  - 二逼青年：编辑微信，发送；编辑微信，发送；编辑微信，发送；编辑微信，发送；编辑微信，发送；编辑微信，发送；编辑微信，发送；

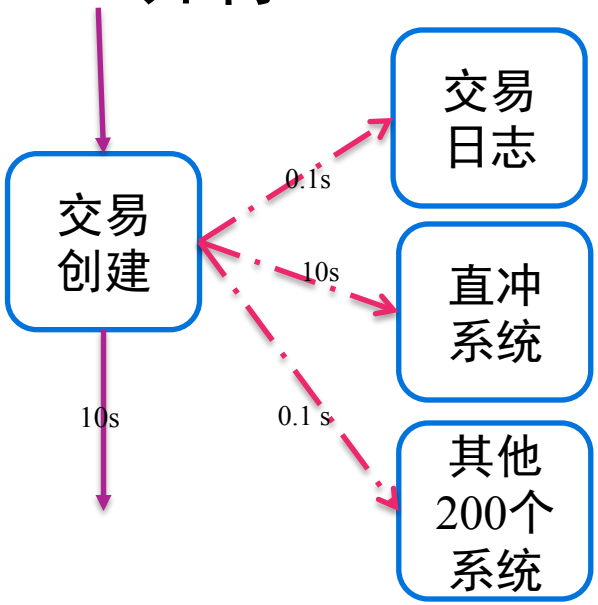
消息系统

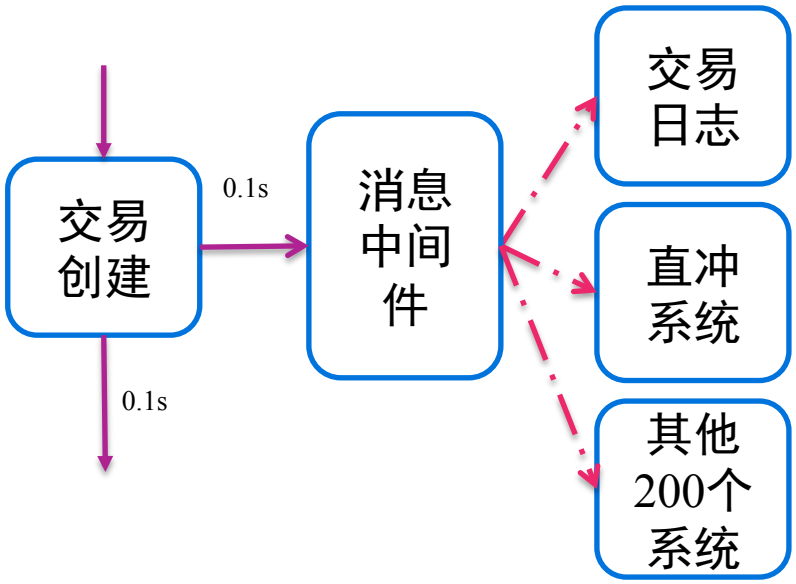
- 消息中间件
  - 解耦
  - 异步
  - 最终一致
  - 并行
- 举一个淘宝的简单的例子



消息系统

- 消息中间件
  - 解耦
  - 异步
  - 最终一致
  - 并行





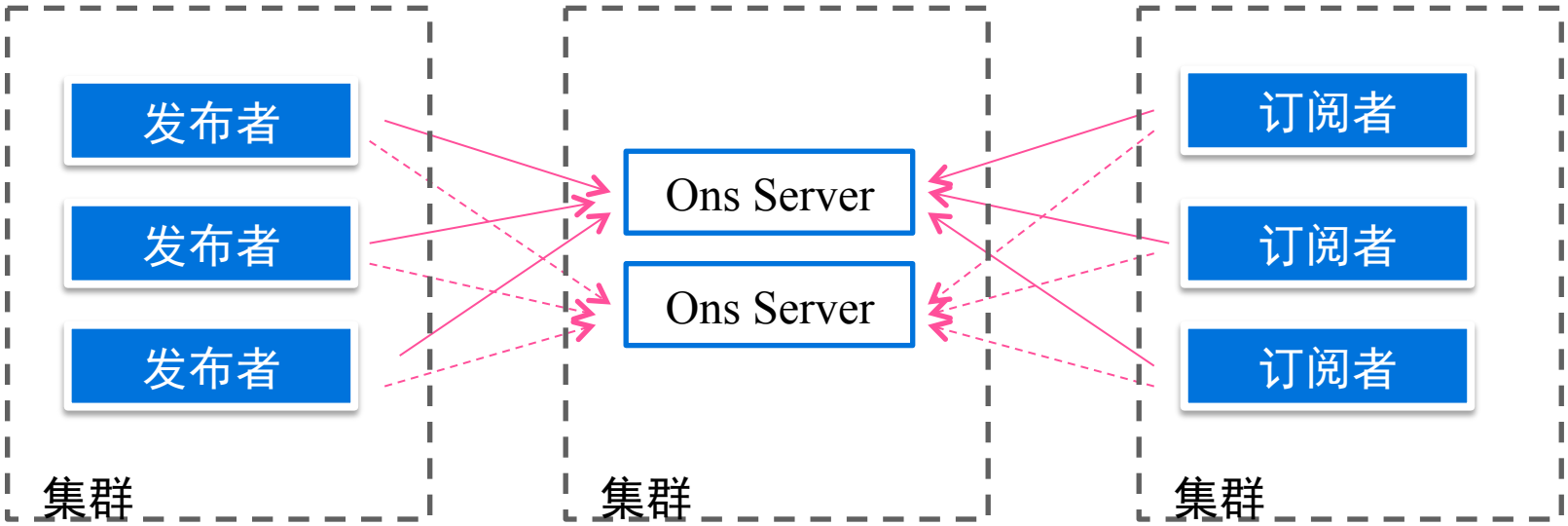
ONS的设计思路

ONS的设计思路和关键概念

- 设计假定：
  - 每台PC机器都可能down机不可服务
  - 任意集群都可能处理能力不足
  - 最坏情况一定会发生
  - 内网环境需要低延迟来提供最佳用户体验
- 关键设计
  - 分布式集群化
  - 强数据安全
  - 海量数据堆积
  - 毫秒级投递延迟

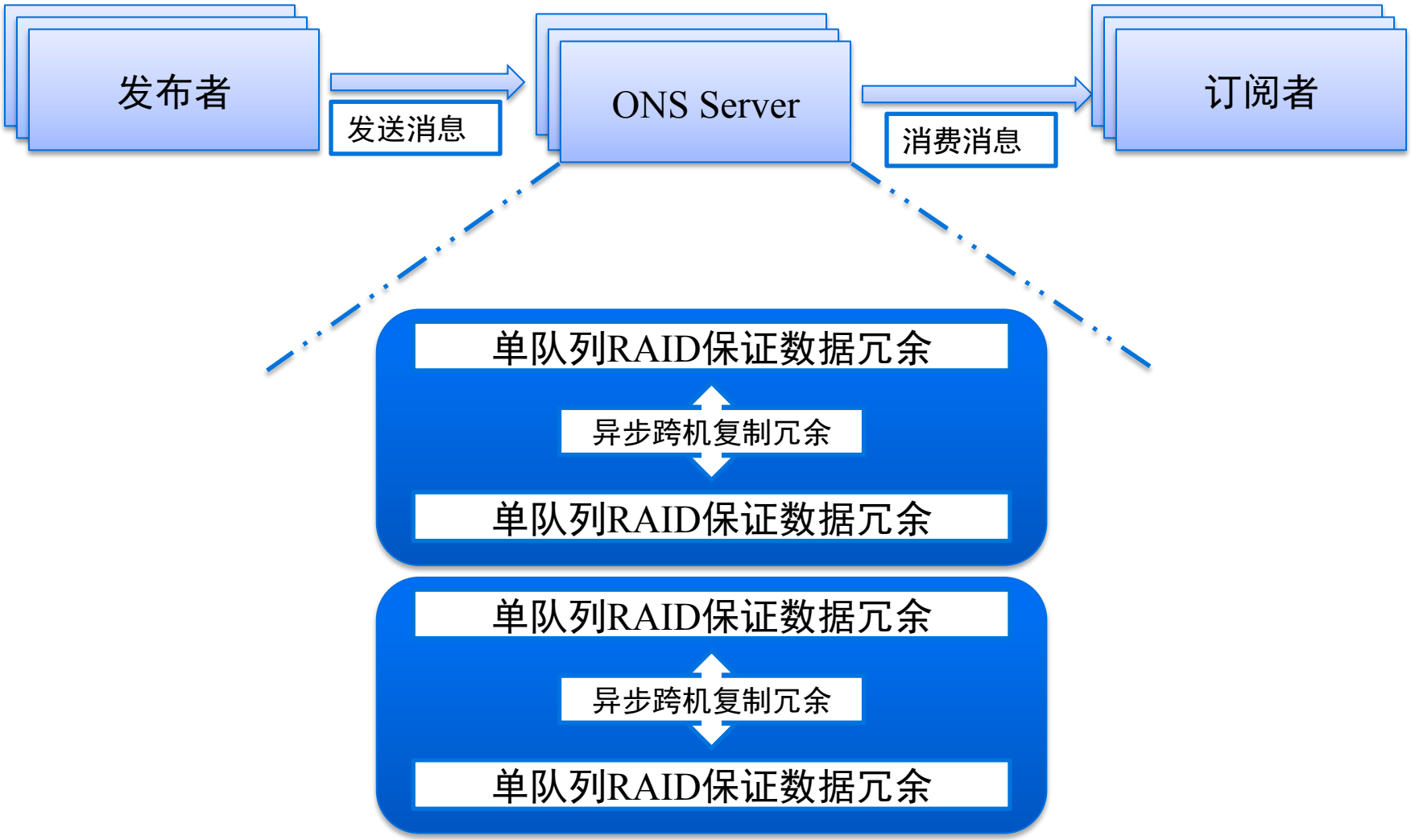
无单点集群化设计

- 理论上无限的处理能力
- 集群级别高可用



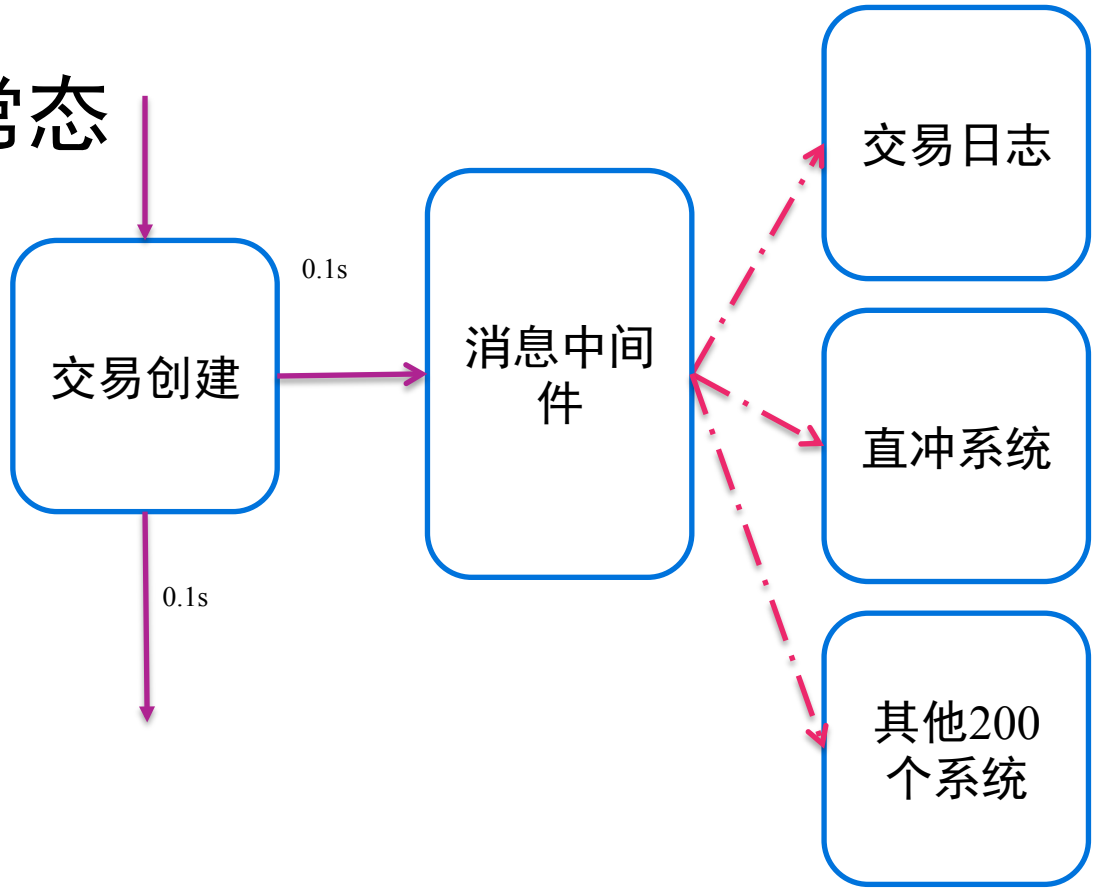
强数据安全和高可用


SALE


发布者
发送消息
ONS Server
消费消息
订阅者
单队列RAID保证数据冗余
异步跨机复制冗余
单队列RAID保证数据冗余
单队列RAID保证数据冗余
异步跨机复制冗余
单队列RAID保证数据冗余

海量数据堆积能力

- 任意集群都可能处理能力不足
- 消息堆积是常态



海量数据堆积能力

- 面向堆积设计
  - 大量堆积，系统稳定，延迟不增
    - 百亿级别的消息堆积能力
    - 双11多年考验
    - 单消息Server不可用数据不丢
- 默认就是落磁盘策略，并针对磁盘吞吐做了大量优化
- 集群可无限扩展，保证足够堆积能力

毫秒级的投递延迟

- 采用长轮询/推送方式

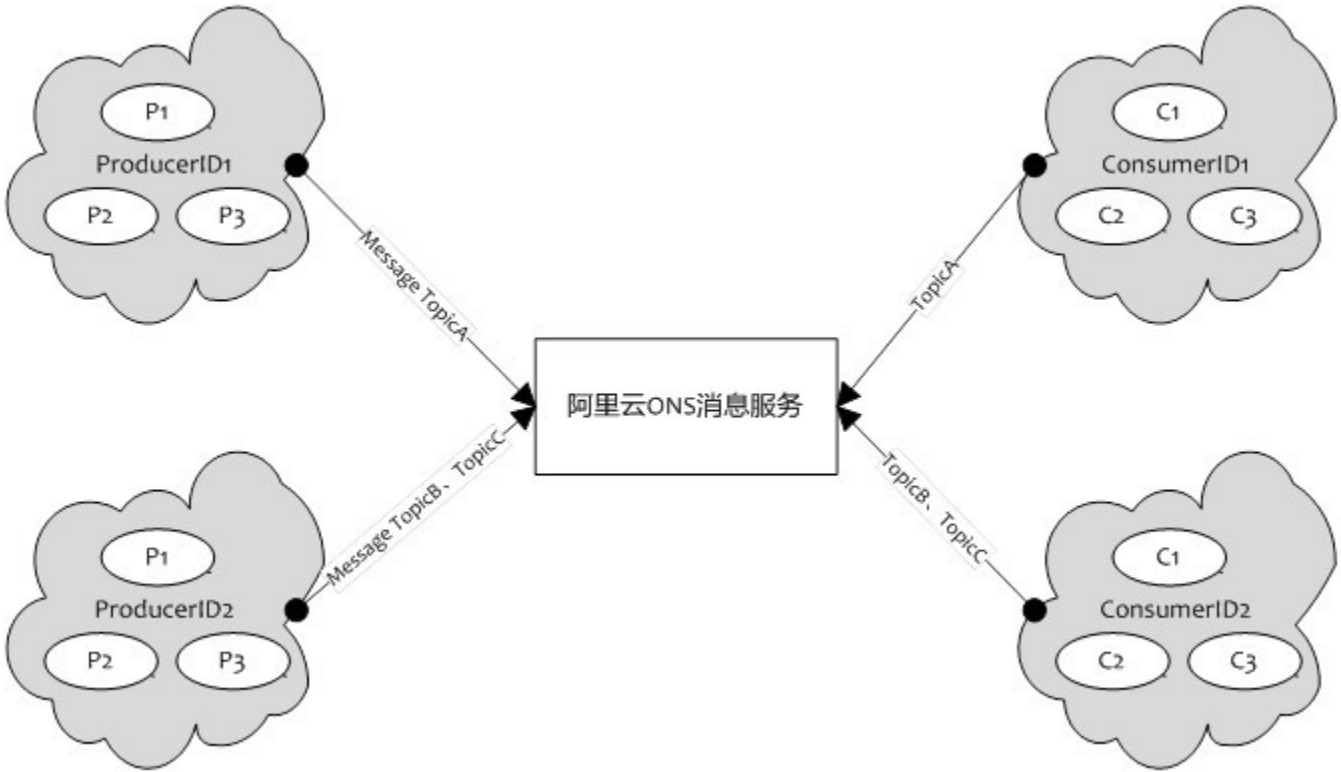
ONS Server

订阅者

ONS的关键概念

主题（Topic）

- 第一级消息类型
- 书的标题
- 交易消息

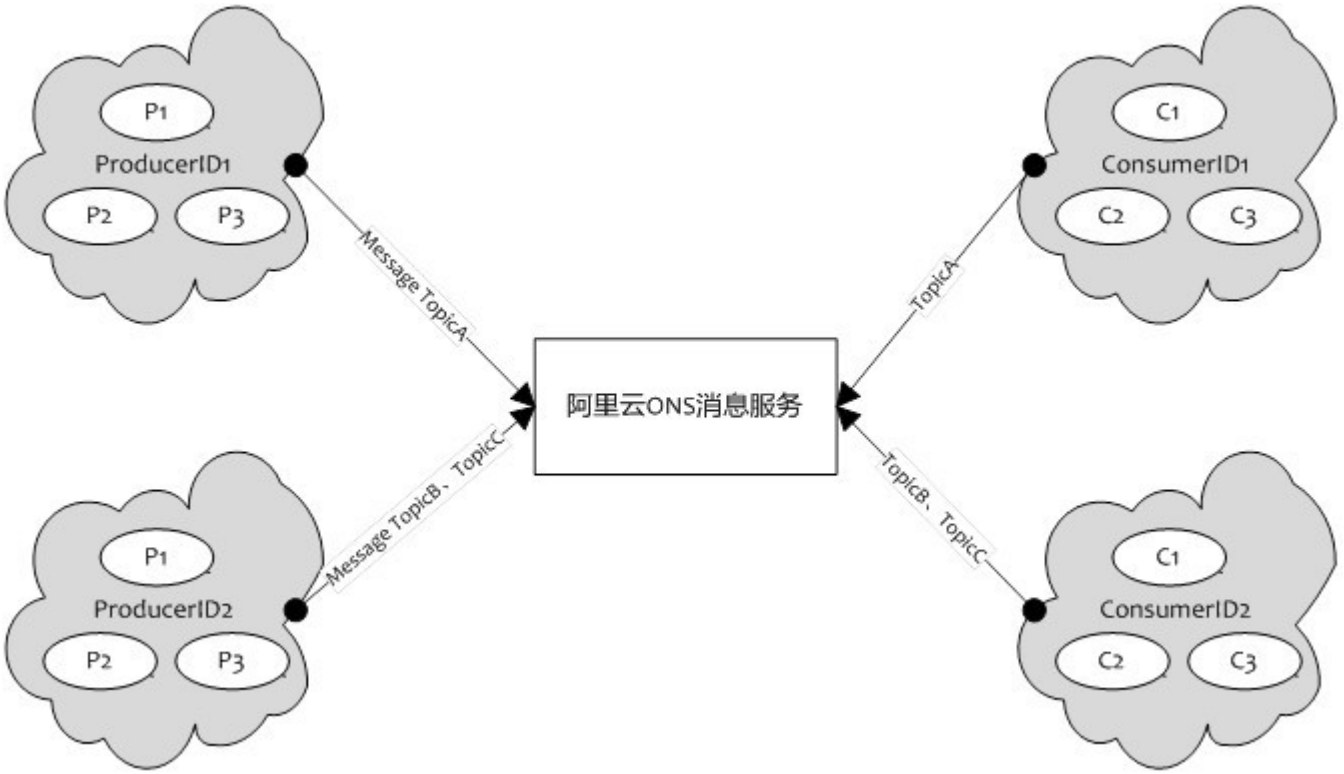


消息类型（Tag/MessageType）

- 第二级消息类型
- 书的目录
  - 方便检索使用
- 交易消息
  - 交易创建
  - 交易完成

发送/订阅组（ProducerID/ConsumerID）

- 发送/接受机器的集群



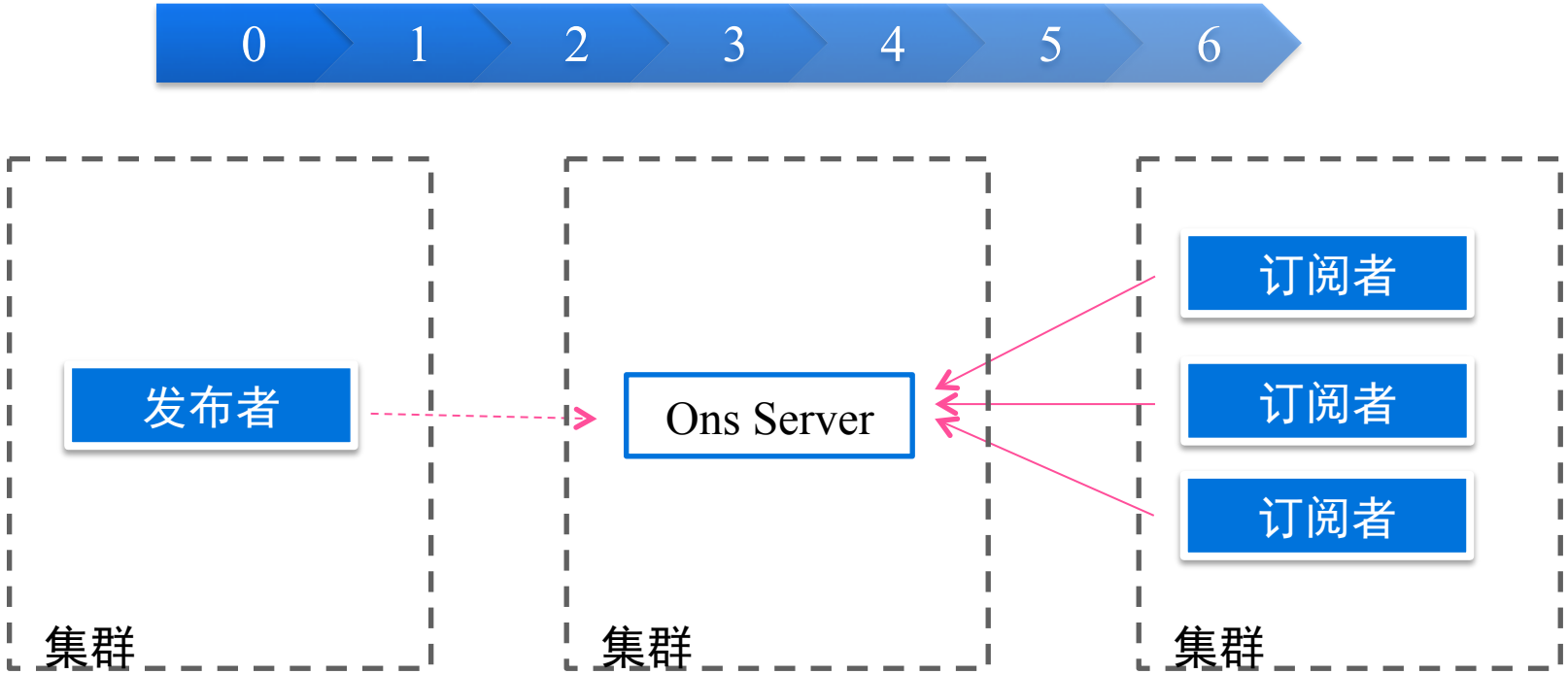
消息乱序问题

消息乱序问题

- 产生原因
- 有序队列优劣分析
- 阿里中间件经验谈

产生原因

- 吞吐+容错 vs. 方便 容易理解



有序队列优劣分析

- 优势：
  - 容易理解
  - 处理问题容易
- 劣势：
  - 并行度瓶颈
  - 异常处理
- BUT
  - 我们需要集群的容错性和高吞吐！

阿里中间件经验谈

- 在世界上解决一个计算机问题最简单的方法：
- “恰好”不需要解决它！

阿里中间件经验谈

- 一笔订单有三个状态(创建，付款，发货)
  - 订单之间没有先后顺序，所以乱序无所谓
  - 某应用只关注付款 ：）

Bob给Smith转账

事务单元

操作指令	耗时
锁定Bob账户	0.001ms
查看Bob是否有100元	1ms
从Bob账号中减少100元	2ms
解锁Bob账户	0.001ms

异步并行消息

异步事务单元

操作指令	耗时
锁定Smith账户	0.001ms
给Smith账户中增加100元	2ms
解锁Smith账户	0.001ms

事务时间序

多人通过消息转账情况

异步并行消息1

异步事务单元2

操作指令	耗时
锁定Smith账户	0.001ms
给Smith账户中增加100元	2ms
解锁Smith账户	0.001ms

异步并行消息2

异步事务单元1

操作指令	耗时
锁定Smith账户	0.001ms
给Smith账户中增加100元	2ms
解锁Smith账户	0.001ms

事务时间序

Smith账户

阿里中间件经验谈

- 不关注乱序的应用是大量存在的
- 队列无序并不意味着消息无序
  - TCP协议
  - 可以通过发送端编号和接收端恢复的方式恢复顺序

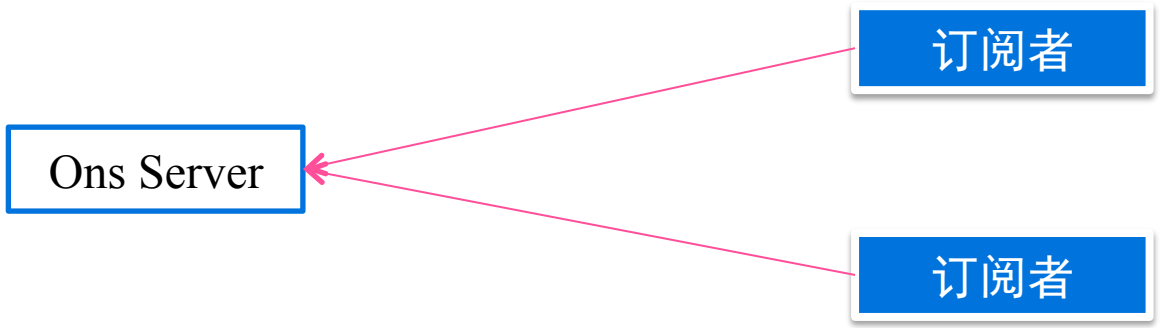
消息重复问题

消息重复问题

- 产生原因
- 阿里中间件经验谈

产生原因

- 网络不可达问题



阿里中间件经验谈

- 最好的解决方法是
- 恰好不需要 – 幂等
  - S*S = S
  - 某个操作无论重复多少次，结果都一样

阿里中间件经验谈

- 幂等 – 无论做多少次结果都一样
  - insert into T (col1) values (1)
  - update T set col = 2 where col = 1
  - delete from T where col = 1
- 非幂等
  - update set col = col + 1

阿里中间件经验谈

- 非幂等消息去重
  - 保证有个唯一ID标记每一条消息
  - 保证消息处理成功与去去重表日志同时出现
- 代价？

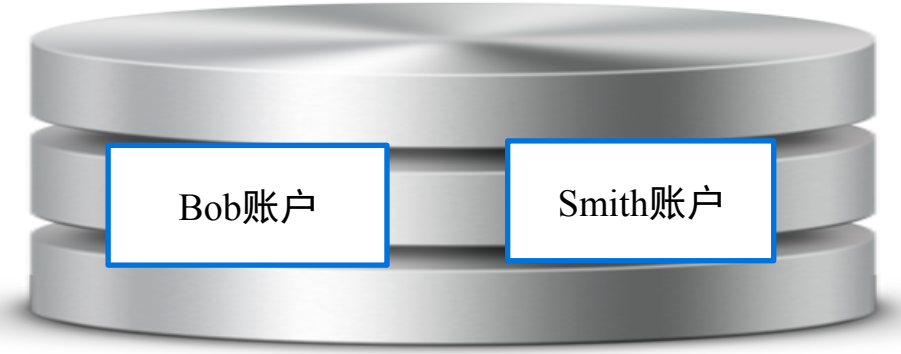
分布式事务与ONS

DRDS实践 – 事务的分布式优化

事务单元

操作指令	耗时	总耗时
锁定Bob账户	0.001ms	5.004ms
锁定Smith账户	0.001ms	
查看Bob是否有100元	1ms	
从Bob账号中减少100元	2ms	
给Smith账户中增加100元	2ms	
解锁Bob账户	0.001ms	
解锁Smith账户	0.001ms	

事务时间序



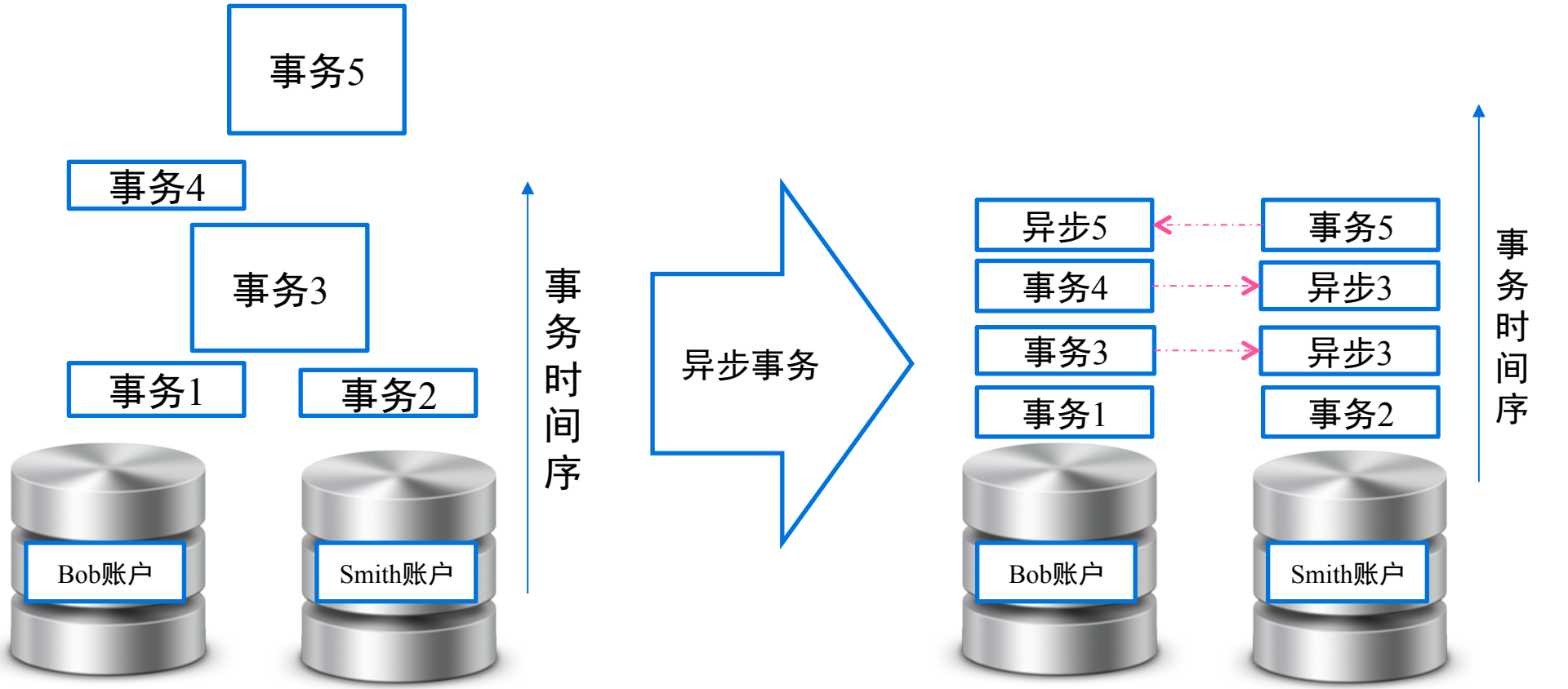
DRDS实践 – 事务的分布式优化

延迟增加
用户体验
下降

操作指令	耗时	总耗时
锁定Bob账户	0.001ms	11.004ms
通过网络锁定Smith账户	2ms+0.001ms	
查看Bob是否有100元	1ms	
从Bob账号中减少100元	2ms	
通过网络给Smith账户中增加100元	2ms+2ms	
解锁Bob账户	0.001ms	
通过网络解锁Smith账户	2ms+0.001ms	

事务时间序

DRDS实践 – 事务的分布式优化



DRDS实践 – 事务的分布式优化

事务单元

操作指令	耗时
锁定Bob账户	0.001ms
查看Bob是否有100元	1ms
从Bob账号中减少100元	2ms
解锁Bob账户	0.001ms

异步并行消息

异步事务单元

操作指令	耗时
锁定Smith账户	0.001ms
给Smith账户中增加100元	2ms
解锁Smith账户	0.001ms

事务时间序

ONS消息与事务转账

- 关键设计难点
  - 如何保证消息发出与Bob账户减钱同时成功或同时失败？
  - 消息处理超时如何解决？
  - 消息处理失败如何解决？

同时成功、同时失败（事务消息）

消息发送者

发消息

事务单元

事务操作
Trx.begin()
查看Bob是否有100元
减少Bob 100元
Trx.commit()

ONS消息集群

消息接收者

收消息处理

同时成功、同时失败（事务消息）

消息发送者

事务单元

事务操作
Trx.begin()
查看Bob是否有100元
减少Bob 100元
Trx.commit()

发消息

ONS消息集群

消息接收者

收消息处理

同时成功、同时失败（事务消息）

消息发送者

发消息

事务单元

事务操作
Trx.begin()
查看Bob是否有100元
减少Bob 100元
Trx.commit()

确认消息发送

ONS消息集群

消息接收者

收消息处理

处理超时问题（重复）

消息发送者

发消息

事务单元

确认消息发送

ONS消息集群

消息接收者

事务单元

事务操作
Trx.begin()
Smith加一百元
插入去重表
Trx.commit()

小结

ONS已经正式登录阿里云

- 阿里使用最广泛的消息服务系统
  - 包含交易、商品几乎所有的的应用都在使用。
- 高峰期流量
  - 双11 每秒8亿笔交易
- 堆积消息，系统写入不受到影响
- 支持事务消息模式

ONS已经正式登录阿里云

- http://www.aliyun.com/product/ons
- DRDS/ONS QQ群
  - 326140964
  - 可扫码加群

DRDS 讨论群

扫一扫二维码，加入该群。