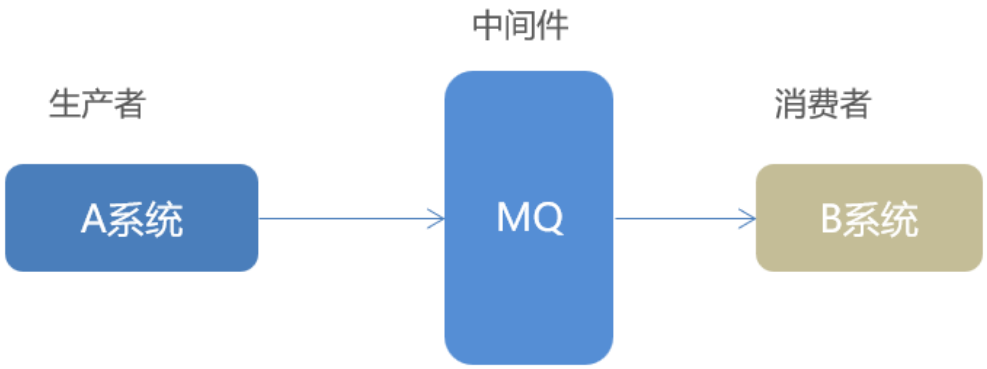
MQ基础知识及各产品

1.基本概念

- MQ，消息队列，存储消息的中间件
- 分布式系统通信两种方式：直接远程调用 和 借助第三方 完成间接通信
- 发送方称为生产者，接收方称为消费者

MQ全称 Message Queue（消息队列），是在消息的传输过程中保存消息的容器。多用于分布式系统之间进行通信。



2.MQ的优势和劣势

优势

- 应用解耦：提高系统容错性和可维护性
- 异步提速：提升用户体验和系统吞吐量
- 削峰填谷：提高系统稳定性

劣势

- **系统可用性降低**

系统引入的外部依赖越多，系统稳定性越差。一旦 MQ 宕机，就会对业务造成影响。如何保证MQ的高可用?

- **系统复杂度提高**

MQ 的加入大大增加了系统的复杂度，以前系统间是同步的远程调用，现在是通过 MQ 进行异步调用。如何保证消息没有被重复消费？怎么处理消息丢失情况？那么保证消息传递的顺序性?

- **一致性问题**

A 系统处理完业务，通过 MQ 给B、C、D三个系统发消息数据，如果 B 系统、C 系统处理成功，D 系统处理失败。如何保证消息数据处理的一致性?

既然 MQ 有优势也有劣势，那么使用 MQ 需要满足什么条件呢?

①生产者不需要从消费者处获得反馈。引入消息队列之前的直接调用，其接口的返回值应该为空，这才让明明下层的动作还没做，上层却当成动作做完了继续往后走，即所谓异步成为了可能。

②容许短暂的不一致性。

③确实是用了有效果。即解耦、提速、削峰这些方面的收益，超过加入MQ，管理MQ这些成本。

3.常见的MQ

	RabbitMQ	ActiveMQ	RocketMQ	Kafka
公司/社区	Rabbit	Apache	阿里	Apache
开发语言	Erlang	Java	Java	Scala&Java
协议支持	AMQP，XMPP，SMTP，STOMP	OpenWire,STOMP，REST,XMPP,AMQP	自定义	自定义协议，社区封装了http协议支持
客户端支持语言	官方支持Erlang，Java，Ruby等,社区产出多种API，几乎支持所有语言	Java，C，C++，Python，PHP，Perl，.net等	Java，C++（不成熟）	官方支持Java,社区产出多种API，如PHP，Python等
单机吞吐量	万级（其次）	万级（最差）	十万级（最好）	十万级（次之）
消息延迟	微妙级	毫秒级	毫秒级	毫秒以内
功能特性	并发能力强，性能极其好，延时低，社区活跃，管理界面丰富	老牌产品，成熟度高，文档较多	MQ功能比较完备，扩展性佳	只支持主要的MQ功能，毕竟是为大数据领域准备的。

ActiveMQ

优点

- 单机吞吐量：万级
- topic数量都吞吐量的影响：
- 时效性：ms级
- 可用性：高，基于主从架构实现高可用性
- 消息可靠性：有较低的概率丢失数据

- 功能支持：MQ领域的功能极其完备

缺点:

官方社区现在对ActiveMQ 5.x维护越来越少，较少在大规模吞吐的场景中使用。

Kafka

号称大数据的杀手锏，谈到大数据领域内的消息传输，则绕不开Kafka，这款为大数据而生的消息中间件，以其百万级TPS的吞吐量名声大噪，迅速成为大数据领域的宠儿，在数据采集、传输、存储的过程中发挥着举足轻重的作用。

Apache Kafka它最初由LinkedIn公司基于独特的设计实现为一个分布式的提交日志系统(a distributed commit log)，之后成为Apache项目的一部分。

目前已经被LinkedIn，Uber, Twitter, Netflix等大公司所采纳。

优点

- 性能卓越，单机写入TPS约在百万条/秒，最大的优点，就是吞吐量高。
- 时效性：ms级
- 可用性：非常高，kafka是分布式的，一个数据多个副本，少数机器宕机，不会丢失数据，不会导致不可用
- 消费者采用Pull方式获取消息, 消息有序, 通过控制能够保证所有消息被消费且仅被消费一次;
- 有优秀的第三方Kafka Web管理界面Kafka-Manager；
- 在日志领域比较成熟，被多家公司和多个开源项目使用；
- 功能支持：功能较为简单，主要支持简单的MQ功能，在大数据领域的实时计算以及日志采集被大规模使用

缺点:

- Kafka单机超过64个队列/分区，Load会发生明显的飙高现象，队列越多，load越高，发送消息响应时间变长
- 使用短轮询方式，实时性取决于轮询间隔时间；
- 消费失败不支持重试；
- 支持消息顺序，但是一台代理宕机后，就会产生消息乱序；
- 社区更新较慢；

RabbitMQ

RabbitMQ 2007年发布，是一个在AMQP(高级消息队列协议)基础上完成的，可复用的企业消息系统，是当前最主流的消息中间件之一。

RabbitMQ优点：

- 由于erlang语言的特性，mq 性能较好，高并发；
- 吞吐量到万级，MQ功能比较完备
- 健壮、稳定、易用、跨平台、支持多种语言、文档齐全；
- 开源提供的管理界面非常棒，用起来很好用
- 社区活跃度高；

RabbitMQ缺点：

- erlang开发，很难去看懂源码，基本职能依赖于开源社区的快速维护和修复bug，不利于做二次开发和维护。
- RabbitMQ确实吞吐量会低一些，这是因为他做的实现机制比较重。
- 需要学习比较复杂的接口和协议，学习和维护成本较高。

RocketMQ

RocketMQ出自 阿里公司的开源产品，用 Java 语言实现，在设计时参考了 Kafka，并做出了自己的一些改进。

RocketMQ在阿里集团被广泛应用在订单，交易，充值，流计算，消息推送，日志流式处理，binglog分发等场景。

RocketMQ优点：

- 单机吞吐量：十万级
- 可用性：非常高，分布式架构
- 消息可靠性：经过参数优化配置，消息可以做到0丢失
- 功能支持：MQ功能较为完善，还是分布式的，扩展性好
- 支持10亿级别的消息堆积，不会因为堆积导致性能下降
- 源码是java，我们可以自己阅读源码，定制自己公司的MQ，可以掌控

RocketMQ缺点：

- 支持的客户端语言不多，目前是java及c++，其中c++不成熟；

- 社区活跃度一般
- 没有在 mq 核心中去实现JMS等接口，有些系统要迁移需要修改大量代码