01 | 业界的主流消息队列是如何发展起来的?

许文强 · 深入拆解消息队列47讲

你好，我是文强。

作为一个消息队列的老兵，我经常被团队的新同学或者客户的研发人员问：我希望在我们的业务中引入消息队列，应该选择哪一款合适？是不是用最近很火的 Pulsar 就好了，但是业务团队推荐用 RocketMQ 或 RabbitMQ，而大数据团队推荐用 Kafka，有没有什么选择的标准可以参考呢?

这个问题你之前一定疑惑过，可能也有了一套自己的选择标准，不过遇到这种情况，我都会先反问：你觉得消息队列是做什么的?

自从负责消息队列云服务之后，我接触了很多客户，发现基本 90% 以上的用户（大概率你也在其中），业务的数据量都不大，场景也不复杂。只用到了消息队列最基本的生产和消费功能，把消息队列当做缓冲分发的管道，基本不会用到消息队列的高级功能，比如死信队列、事务消息、延时消息等等。

所以，很多情况下，我们不需要很复杂的消息队列，有些时候甚至都不需要引入业界标准的消息队列产品。是不是和你之前的想法不太一致？今天我们就从这个有点反常识的观点开始讨论。

你真的需要标准消息队列吗?

我第一次用消息队列是在我负责游戏论坛开发的时候，有一个功能是需要把用户对帖子的评论和点赞数据发给下游分析。你可以想一下，如果你接到这个需求，会怎么做呢?

我的第一个想法就是在流程里面加一个函数，封装一下，再把数据通过 HTTP API 发送给下游，就完成了。

不过我上级选择了另外一套方案，引入 Kafka，在主流程里面先把数据发送到 Kafka，再让下游去消费数据。那时候 Kafka 还是 0.8.* 的版本，公司用得很少，我们团队也是第一次用，还特意找运维搭了套集群。

> Apache Kafka 是 LinkedIn 在 2011 年贡献给了 Apache 软件基金会的分布式消息队列。主要满足大数据领域中的高吞吐量、低延迟场景。核心功能较为简单，只提供生产和消费，后来加了幂等和事务。采用通用分布式架构实现发布 – 订阅模型，能高效地处理海量数据。

当时我不太理解加 Kafka 的意义是什么。

因为我们的需求非常简单，只涉及生产和消费两个动作，而且现网一直在用 Redis。当时我认为使用 Redis 的 List 实现 Push/Pop 就可以满足需求，效果是一样的，还省去 Kafka 的维护和学习成本。进一步，如果担心 Redis 的数据没做持久化丢失了，也可以通过 MySQL 的表，把数据 insert 进去，下游去 select 出来，也能实现 Push/Pop 的效果。

那为什么还是用 Kafka 呢?

当时有个业务背景：用户的回复会包含图片和表情等复杂结构数据，导致内容可能会特别长。另外遇到高峰时，回帖数量会特别多，短时间内可能有几千万的回帖，回帖的总内容很大。

Redis 的问题在于数据都存在内存里，如果数据没有及时消费，就会打爆内存。而且因为回帖内容可能很大，在 Redis 的 Value 里存大的数据会有性能和稳定性问题。而 MySQL 在 insert 上是有性能瓶颈的，短时间内大量回帖，插入会特别频繁，性能就扛不住。

而 Kafka 就没有这个问题，作为一个消息队列，它的特点就是高吞吐、大消息、高并发、持久化，不会存在性能、功能、稳定性方面的问题。这就是我们选择 Kafka 的理由。

其实如果没有大消息和大流量等复杂场景，是可以选用非标准消息队列产品的。比如在用户状态审核的场景中，只需要向下游传递用户 ID 和审核结果，结构简单，数量有限。这时候选择非标准消息队列比如 Redis 和 MySQL 也是可以的。

所以，是否选择使用标准消息队列产品，取决于你的数据和业务场景的需求。当数据量大、场景复杂后，才必须引入标准消息队列，因为它有高吞吐、持久化、长久堆积的特性。

既然用非标准消息队列也可以满足相应需求，那到底什么是消息队列呢?

从宏观上来讲，我会认为具有缓冲作用、具备类发布和订阅能力的存储引擎都可以称做消息队列。因为消息队列的最基本功能就是生产和消费，在发布订阅之上，扩展如死信队列、顺序消息、延时消息等高阶能力，并实现高吞吐、低延时、高可靠等特性，就成为了我们所熟知的功能齐全的标准消息队列。

现在业界都有哪些标准的消息队列呢？我们一起来梳理下。

业界都有哪些消息队列?

还是跟随我的时间线来看。后来我负责的广告业务用到了另一款消息队列 RabbitMQ，当时我最直观的感觉就是：它功能比 Kafka 多好多，支持延时消息、死信队列、优先级队列、事务消息等等，业务上也是因为要用到这些功能才选择的 RabbitMQ。

> RabbitMQ 是 2007 年由国外一家叫做 Rabbit 的公司开源的，用 Erlang 写成的消息队列。主要满足业务中消息总线的场景。特点是功能丰富，低流量下稳定性较高，基本具备消息队列所应该具备的所有功能。缺点是在大流量的情况下会有明显的瓶颈和稳定性风险。

当时开源的消息队列并不丰富。基于 JMS 协议发展出来的 ActiveMQ 因为功能和稳定性问题，用的人比较少。Kafka 刚开源不久，功能只有生产和消费。AMQP 协议的 Erlang 实现 RabbitMQ，因为功能丰富，稳定性较高，成为主流选择。

> ActiveMQ 项目最初是由 LogicBlaze 的创始人于 2004 年创建的，支持完成 JMS 规范的消息队列。因为生态、功能定位方面的原因，在国内用的人并不多。
>
> AMQP 是一个消息队列协议规范，它不是一款具体的消息队列。因为不同消息队列的访问协议是不一样的，导致不同的消息队列需要用不同的 SDK 访问，客户的切换成本很高。2003 年，多个金融服务机构希望制定一个消息队列的协议规范，希望不同的消息队列的协议都根据这个标准实现，这样就可以不需要重复开发 SDK，不同的应用程序之间的交互和切换可以更简单、更方便。这就是 AMQP 的由来。

不过开源选择少，并不意味着那时候没有好的消息队列可用，商业化的闭源的消息队列一直在蓬勃发展，但需要购买才能使用，而且不便宜。

比如微软 1997 年推出的 MSMQ、AWS 2004 年推出的 SQS，而消息队列领域最牛的，某种意义上（从收入维度看）一直都是 IBM MQ，在金融、证券行业用得特别多，几乎一统天下。数据显示，到了 2020 年 IBM MQ 每年在全球仍有近 10 亿美金的营收。现在国内各大云厂商消息队列的收入加起来，可能都不够 IBM MQ 的零头。

后来开源社区逐渐完善，阿里的 RocketMQ 在 2017 年从 Apache 基金会毕业，Pulsar 也在 2018 成为了 Apache 的顶级项目，开源社区生态逐渐繁荣，选择慢慢变多。

RocketMQ 在定位上和 RabbitMQ 很像，功能丰富，在业务消息中经常会用到。不过 RocketMQ 是在移动互联网浪潮下发展起来的，业务场景更加复杂，也支持更多功能，比如消息 Tag、消息轨迹、消息查询等等。

除了功能层面，在架构和性能层面，RabbitMQ 开发设计早，当时分布式的设计理念还不成熟，导致它在架构层面的设计存在较大的缺陷，遇到大流量、高并发的时候，容易出现集群不可用、网络分区等情况无法解决。而 RocketMQ 在分布式架构上实现得更合理优雅，在大流量、高并发的场景下表现优秀稳定。

> RocketMQ 是 2013 年阿里研发，2016 年开源，可满足大规模微服务场景的消息队列产品。可以理解为是 RabbitMQ 的高可用、分布式升级版。功能丰富，基本可以满足业务消息场景下的所有需求。稳定性、数据可靠性方面的表现都较好。性能介于 RabbitMQ 和 Kafka 之间。

Pulsar 和 Kafka 很像，主要定位在流领域，主打大吞吐的流式计算。但 Kafka 的功能比较简单，支持基本的发布订阅、幂等、事务消息。Pulsar 在满足这些功能的基础上，也希望支持 RocketMQ 和 RabbitMQ 的功能，所以功能最丰富。

除了功能层面，在架构和性能层面上，Pulsar 的架构设计比 Kafka 更符合当前云原生架构，它的定位是 Kafka 的升级版，主要解决 Kafka 当前的一些痛点问题，比如集群扩缩容慢、分区迁移需要 Rebalance、无法支持超多分区等。性能目前没有特别大的差距。

不过 Pulsar 发展时间较短，架构较复杂，功能支持较多，当前阶段在稳定性上 Kafka 会比 Pulsar 好非常多。

> Pulsar 2017 年由 Yahoo 开发的消息队列系统。一开始定位是流计算领域，可以理解为 Kafka 的升级版，近期希望同时发展消息和流两个方向。其架构上的设计理念较为优秀，比如计算存储分离、弹性、多租户。在功能上目前正在追赶 RabbitMQ/RocketMQ。性能层面，和 Kafka 没有明显的差异。但当前阶段的稳定性还需要提升。

这里总结一下，当前开源社区用的较多的消息队列主要有 RabbitMQ、Kafka，RocketMQ 和 Pulsar 四款。下面我整理了一张消息队列发展的时间脉络图，供你参考。

在开源社区发展的这段时间，国内大厂也一直在自研消息队列，比如阿里的 RocketMQ、腾讯的 CMQ 和 TubeMQ、京东的 JMQ、字节的 BMQ。只是发展程度不一样，有的开源了成为顶级项目，有的慢慢消亡了，有的仅限在公司内部使用。

虽然有这么多的消息队列，但它们发展的方向其实是一致的。

消息队列的发展脉络

结合我的个人经历，我们可以从消息队列的历史中抽象出两条交织的发展脉络：上层的需求变化和下层的架构演进。

> 从需求发展路径上看，消息队列的发展趋势是：**消息 -> 流 -> 消息和流融合**。
>
> 从架构发展的角度来看，消息队列的发展趋势是：**单机 -> 分布式 -> 云原生 /Serverless**。

在 90 年代到 21 世纪初，以 IBM MQ 和 AMQP 为代表的消息队列主要满足业务上对消息的需求，即异步通讯、架构解耦。

2010 年左右，移动互联网发展，大数据兴起，传统的消息队列在架构上无法满足大流量的吞吐需求，就发展出了以 Kafka 为代表的消息队列，主打大吞吐、大流量。我们也进入了分布式的时代，现在大家熟知的消息队列都是分布式的架构，所以会有分区、副本、一致性概念。

随着业务场景越来越复杂，业务消息的数据量也越来越大。基于开源 AMQP 的 RabbitMQ 在性能和架构上已经无法满足消息场景的需求，从而发展出了 RocketMQ。

近几年随着云计算的发展、云原生和 Serverless 的理念兴起，在弹性、成本的驱动下，消息队列的架构往云原生 /Serverless 方向演变，简单来说，就是利用云上的弹性计算、存储等基础设施去实现架构的 Serverless，按需使用、按量付费，最终达到使用端感受到的免运维、低成本。

基于云原生架构设计的 Pulsar 开始走向成熟，业界的 MQ 也出现了**计算存储分离、分层存储、多租户、弹性计算等概念**。

云原生 /Serverless 的发展趋势，你应该听得比较多了，“消息和流融合”的趋势可能有些陌生，我稍微解释一下。

什么是消息和流?

消息、流分开来看都比较好理解。

> 消息就是业务消息，在业务架构（比如微服务架构）中用来作消息传递，做系统的消息总线，比如用户提交订单的流程。
>
> 流，就是在大数据架构中用来做大流量时的数据削峰，比如日志的投递流转。

消息和流融合就是这两个事情都能做。不过为什么会有消息和流融合的这个趋势呢?

其实都是钱的原因。虽然消息队列是基础组件，但是功能比较单一，主要是缓冲作用，在消息、流的方向上，功能需求一直是相对固定的，细分的市场也都有领头组件，流领域目前是 Kafka 一家独大，消息领域的头部玩家，国外是 RabbitMQ，国内是 RocketMQ。

对 MQ 厂商来说，如果希望扩大产品份额或者新品抢占市场，就需要有**独特竞争力，核心就是自己产品的功能和成本，**即功能更多更丰富、成本更低。这里的成本指的是能为客户，也就是消息队列使用者，节省多少资源、人力成本。

因为对我们使用者来说，业务和大数据都不陌生，日常工作都需要进行业务和数据系统的开发。但是麻烦的问题是：在业务消息中，我们经常需要使用 RocketMQ 或 RabbitMQ，在大数据系统中，我们又需要使用 Kafka 或 Pulsar。

所以，**运维侧需要运维多款 MQ，研发侧需要学习使用多款 MQ**，这在一定程度上拉升了研发和运维的成本。如果有一款消息队列满足所有场景，只需要部署一款消息队列，就能满足所有业务的需求，这种设计思想是非常有商业价值的。

现在我们也可以看到业界主流的 4 款消息队列，在消息和流的融合上各有动作。

> RabbitMQ 因为开发语言、架构和社区的活跃度、定位的原因，基本不会走这条路。
>
> Kafka 虽然也强调云原生，但目前主要工作在自身的架构优化上，比如去 ZooKeeper，暂时在消息方向没有提出明确概念。但在我看来，未来 Kafka 应该会往这个方向转变，因为流的场景始终会有瓶颈，打通一个新方向在商业上肯定是有价值的。

RocketMQ 在消息领域已经非常成熟，社区也希望打通流的场景，扩展使用范围，提升竞争力，抢占市场，也在往这个方向努力。

Pulsar 是一个新兴架构，没有历史包袱，主打的就是云原生的消息和流的融合架构，希望满足更多场景，解决更多业务需求。

总结

现在你应该能解答我们开头提出的问题了，广义上讲，消息队列是有缓冲作用、具备类发布和订阅能力的存储引擎。而技术方案如何选择消息队列，来源于业务场景和数据量。如果只需要最基本的生产和消费功能，可以不用标准消息队列产品，大部分公司或业务的场景下，一款消息队列就能满足所有需求了。

技术的演进都是商业驱动的，消息队列的演进，无论是从需求发展路径上看是消息 -> 流 -> 消息和流融合，还是从架构发展角度的单机 -> 分布式 -> 云原生 /Serverless，本质其实都是在思考如何降低成本和吸引客户。

为了降低成本，弹性是最基础的要求。所以消息队列在技术上，对计算弹性的需求提出了计算存储分离架构，对低存储成本的需求提出了分层存储的概念，对资源复用的需求提出了多租户的概念。

为了吸引客户，各个消息队列都在尽量提高自己的竞争力，围绕着功能、容灾、多架构、生态建设展开。

不过要注意，消息和流只是业界的趋势，不是我们作为使用者必然的非此即彼的选择。在开发者实际使用的时候，我也发现很多人会将 Kafka 当做一个业务消息总线在用，也有人使用 RocketMQ 传递大流量的日志，当做大数据架构中的管道在用。

思考题

你现在做的项目业务场景和数据量是什么样的，是否会考虑引入消息队列，你会引入哪款消息队列呢?

期待看到你的留言，也欢迎你把这节课分享给感兴趣的朋友。我们下节课再见！

精选留言 (12)

Geek_206e82
2023-06-26 来自广东

我是做游戏的，目前我在使用pulsar，功能有瞬时消息，也有顺序消息，能同时满足这2个就pulsar了

作者回复: 瞬时消息的特性确实很多主流消息队列都不支持，是一个很特殊的场景。 之前接触过公司内部的一个大型游戏团队，也是因为瞬时消息选择使用pulsar。

9

张申傲
2023-06-26 来自北京

请问下老师，消息和流的本质区别是什么呢？我理解流也可以看作是大数据领域的业务消息，只不过消息体是日志、指标等等这些数据，为什么要强调消息和流的融合呢？

作者回复: 我理解是商业化层面的考虑，文章中我有写了一下我的观点

共 5 条评论>

6

顾庆隆
2023-06-27 来自北京

文强老师，能不能讲一下Nats，Golang编写的消息队列，CNCF孵化的项目，设计上非常有特点，功能和性能都挺出色，与课程中四个消息队列想比，在分布式集群通信、消息持久化设计、集群互通和集群级联通信上别具一格，很想听听老师的分析。

作者回复: 你这一说，我也很有兴趣分析一下。这个我记录一下，可能要放到最后了。这个MQ 目前主流比较少用。不过从技术设计上来看，是蛮有参考分析价值的。

共 2 条评论>

5

Geek.Kwok

2023-06-27 来自上海

我司就在用 Kafka 当做一个业务消息总线在用，当然同时也用来传递日志。从运维成本的角度考虑，一家公司一般只会选择使用一个MQ。

作者回复: 在我看来，业务侧的功能需求是第一考虑项。如果功能上一个MQ都能满足的话，选择只运维一个MQ是可以的，反之就得考虑是否运维多款MQ。

2

aoe

2023-06-27 来自浙江

回忆
以前因为 Kafka 吞吐量最强选择使用，但实际数据量用 MySQL 支撑都绰绰有余；
后来 Pulsar 出来了，看了介绍比 Kafka 还厉害，心动了，没行动
最近在使用 RocketMQ 是因为公司决定使用的

业务场景
「购买商品成功」统计用户消费排名、计算平台与商户收益、计算活动有效期内的相关数据；
处理第三方回调（因为有一次调用方疯狂请求接口压暴了服务器）；
「商品秒杀」将 TCP 请求压力转移到 MQ

数据量
最高峰值：10 万 QPS/秒

2

——

2023-06-30 来自浙江

老师，你好，有没有可能带我们搭一个数据量相对大，并发量相对大的业务测试环境，再接入各种消息队列进行横向测评，对比使用消息队列和不使用消息队列的区别，这样，对我们这种传统行业，平时很少接触到高并发，大数据量的业务场景的小白来说，会不会更加直观，好理解一些？毕竟有了直观的数据，才能印象深刻。

作者回复: 你能加一下微信群吗，我们在群里聊。这个是一个很有意思的case，我们现网也经常做这个事情。

共 3 条评论>

2

OAuth
2023-06-26 来自广东

我是做企业供应链的 主要的应用场景是做外围系统与供应链系统的异步数据交换 数据量不大 目前用的rocketmq

作者回复: 业务消息类的消息队列，rocketmq我觉得是第一选择

2

justxhk
2023-07-10 来自广东

集群扩缩容慢、分区迁移需要 Rebalance、无法支持超多分区 kafka有这些问题是不是主要是因为计算和存储没分离

作者回复: 集群扩缩容慢、分区迁移需要 Rebalance：是的，存算分离从理论上可以解决这些问题的。
无法支持超多分区：和存算分离关系不是特别大，主要和元数据存储(基于Zookeeper)、系统架构有关(比如单机分区的文件的存储方式、Controller等)。

杯莫停
2023-07-09 来自四川

我们公司只用到了发布订阅场景，但数据吞吐量比较大，只用到了Kafka

作者回复: 是的，很多用户都是这么用的。其实大部分用户的场景不复杂。

1

运维夜谈
2023-07-07 来自广东

老师，请教个问题，数据量大选择消息队列，但有一个问题就是消息的消费效率取决于消费者，消费者消费不过来就导致消息积压，这这样也会影响业务，这如何避免?
另外，咱有微信群吗?

作者回复: 我是这么理解的，这种有两种情况：
1. 瓶颈在MQ
2. 瓶颈在消费者

如果是1，就得考虑MQ的调优、扩容、扩分区或者换消息队列等
如果是2，得排查一下消费者的瓶颈在哪里，然后针对性处理

课程详情有一个微信群二维码，你可以加一下，群里刚好在讨论这个。

1

春
2023-06-29 来自广东

通俗易懂，条条有理，支持老师

清泉
2023-06-28 来自上海

感觉没什么人提到NSQ的，难道没什么人玩吗

作者回复: 在我接触客户的过程中，确实有接触到用NSQ的客户，但是不多，感觉在国内看起来用的人不多。