消息队列 MQ

产品简介

产品简介

消息队列（Message Queue，简称 MQ）是阿里巴巴集团中间件技术部自主研发的专业消息中间件。产品基于高可用分布式集群技术，提供消息发布订阅、消息轨迹查询、定时（延时）消息、资源统计、监控报警等一系列消息云服务，是企业级互联网架构的核心产品。MQ 历史超过9年，为分布式应用系统提供异步解耦、削峰填谷的能力，同时具备海量消息堆积、高吞吐、可靠重试等互联网应用所需的特性，是阿里巴巴双11使用的核心产品。

MQ 是阿里云正式商用的产品，目前在阿里云多个地域（Region）提供了高可用消息云服务，单个域内采用多机房部署，可用性极高，即使整个机房都不可用，仍然可以为应用提供消息发布服务，产品稳定性及可用性完全按照阿里巴巴内部标准来实施，无单点。

MQ 目前提供 TCP、HTTP、MQTT 三种协议层面的接入方式，支持 Java、C++ 以及 .NET 不同语言，方便不同编程语言开发的应用快速接入 MQ 消息云服务。用户可以将应用部署在阿里云 ECS、企业自建云，或者嵌入到移动端、物联网设备中与 MQ 建立连接进行消息收发，同时本地开发者也可以通过公网接入 MQ 服务进行消息收发。

消息队列 RocketMQ、Apache RocketMQ、消息队列 Kafka、Apache Kafka、RabbitMQ 产品对比

消息队列秉持开放、共享的原则拥抱开源生态，无技术绑定。2016年阿里巴巴正式宣布将 MQ 内核引擎 RocketMQ 捐赠给 Apache 软件基金会；与此同时，全面融合 Kafka 生态，做到无缝迁移，打造更安全、更可靠、更易运维的 Kafka 企业级消息服务。

产品对比：

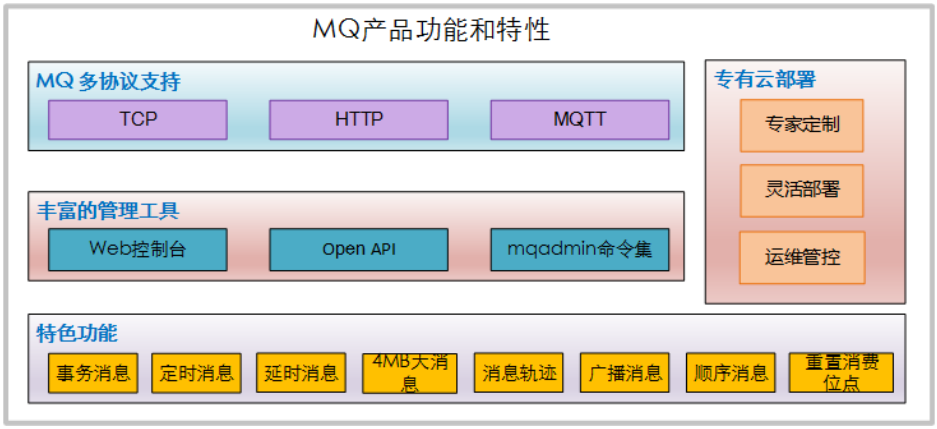
功能	消息队列 RocketMQ	Apache RocketMQ	消息队列 Kafka	Apache Kafka	RabbitMQ（开源）

		（开源）		（开源）	
安全防护	支持	不支持	支持	不支持	不支持
主子账号支持	支持	不支持	支持	不支持	不支持
可靠性	- 同步刷盘 - 同步双写 - 超3份数据副本 - 99.99999999%	- 同步刷盘 - 异步刷盘	- 同步刷盘 - 同步双写 - 超3份数据副本 - 99.99999999%	异步刷盘，丢数据概率高	同步刷盘
可用性	- 非常好，99.95% - Always Writable	好	- 非常好，99.95% - Always Writable	好	好
横向扩展能力	- 支持平滑扩展 - 支持百万级 QPS	支持	- 支持平滑扩展 - 支持百万级 QPS	支持	- 集群扩容依赖前端 - LVS 负载均衡调度
Low Latency	支持	不支持	支持	不支持	不支持
消费模型	Push / Pull	Push / Pull	Push / Pull	Pull	Push / Pull
定时消息	支持（可精确到秒级）	支持（只支持18个固定 Level）	暂不支持	不支持	不支持
事务消息	支持	不支持	不支持	不支持	不支持
顺序消息	支持	支持	暂不支持	支持	不支持
全链路消息轨迹	支持	不支持	暂不支持	不支持	不支持
消息堆积能力	百亿级别 不影响性能	百亿级别 影响性能	百亿级别 不影响性能	影响性能	影响性能
消息堆积查询	支持	支持	支持	不支持	不支持
消息回溯	支持	支持	支持	不支持	不支持
消息重试	支持	支持	暂不支持	不支持	支持
死信队列	支持	支持	支持	不支持	支持
性能（常规）	非常好 百万级 QPS	非常好 十万级 QPS	非常好 百万级 QPS	非常好 百万级 QPS	一般 万级 QPS
性能（万级 Topic 场景）	非常好 百万级 QPS	非常好 十万级 QPS	非常好 百万级 QPS	低	低
性能（海量消息堆积场景）	非常好 百万级 QPS	非常好 十万级 QPS	非常好 百万级 QPS	低	低

MQ 产品功能

MQ 提供了多种协议和开发语言的接入方式以及多维度的管理工具，同时针对不同的应用场景提供了一系列的特色功能。

MQ 产品功能和特性概览



多协议接入

- 支持 HTTP 协议：支持 RESTful 风格 HTTP 协议完成收发消息,可以解决跨语言使用 MQ 问题。
- 支持 MQTT 协议：支持主动推送模型，多级 Topic 模型支持一次触达 1000万+ 终端，可广泛应用于物联网和社交即时通信场景。
- 支持 TCP 协议：区别于 HTTP 简单的接入方式，提供更为专业、可靠、稳定的 TCP 协议的 SDK 接入。

管理工具

- Web 控制台，支持 Topic 管理、发布管理、订阅管理、消息查询、消息轨迹、资源报表以及监控报警管理。
- Open API，提供 API 允许用户将 MQ 管理工具集成到自己的控制台。
- mqadmin 命令集，专有云输出提供一套丰富的管理命令集，以命令方式对 MQ 服务进行管理。

特色功能

- 事务消息，实现类似 X/Open XA 的分布事务功能，以达到事务最终一致性状态。
- 定时（延时）消息，允许消息生产者指定消息进行定时（延时）投递，最长支持40天。
- 大消息，目前默认支持最大 256KB 消息，华北2 地域支持最大 4MB 消息。
- 消息轨迹，通过消息轨迹，用户能清晰定位消息从发布者发出，经由 MQ 服务端，投递给消息订阅者的完整链路，方便定位排查问题。
- 广播消息，允许一个 Consumer ID 所标识的所有 Consumer 都会各自消费某条消息一次。
- 顺序消息，允许消息消费者按照消息发送的顺序对消息进行消费。

- 重置消费进度，根据时间重置消费进度，允许用户进行消息回溯或者丢弃堆积消息。

专有云部署

- 专家定制，提供技术方案设计；专家现场技术支持与培训。
- 灵活部署，支持专有云独立部署，同时支持混合云架构。
- 运维管控，专有云支持 mqadmin 命令集、Open API 运维管理工具，方便管控平台集成以及统一运维。

MQ 产品优势

本文主要介绍 MQ 相比其他消息中间件所具备的优势。

专业

- 消息领域业内专业的消息中间件，产品历史超过 9 年，消息保证不丢，技术体系丰富成熟。
- 阿里内部产品名 MetaQ、Notify；开源社区产品名为 RocketMQ；产品多次在国内外获奖。
- 阿里内部 1000+ 核心应用使用，每天流转几千亿条消息，经过双11交易、商品等核心链路真实场景的验证，稳定可靠。

高可靠

- 一份消息多份落盘存储，经过严格的断电测试，消息依然保证不丢失。
- 允许海量消息堆积，单个 Topic 可堆积 100亿+条消息，系统高流量压力下依然可靠。
- 默认消息持久化存储 3 天，支持重置消费位点消费3天之内任何时间点的消息。

高性能

- 同一网络内，消息传输网络时延在 10 毫秒之内，性能测试下，网卡可被打满。
- 默认单 Topic 发送消息上限为每秒 5000 条，最高可申请扩展至 10W 以上。
- 默认单条消息大小最大支持 256KB，华北2 地域支持 4MB 大消息。

多协议接入

- 支持 HTTP 协议：支持 RESTful 风格 HTTP 协议完成收发消息，可以解决跨语言使用 MQ 问题。
- 支持 MQTT 协议：支持主动推送模型，多级 Topic 模型支持一次触达1000万+ 终端，可广泛应用于物联网和社交即时通信场景。
- 支持 TCP 协议：区别于 HTTP 简单的接入方式，提供更为专业、可靠、稳定的 TCP 协议的 SDK 接入。

独立部署

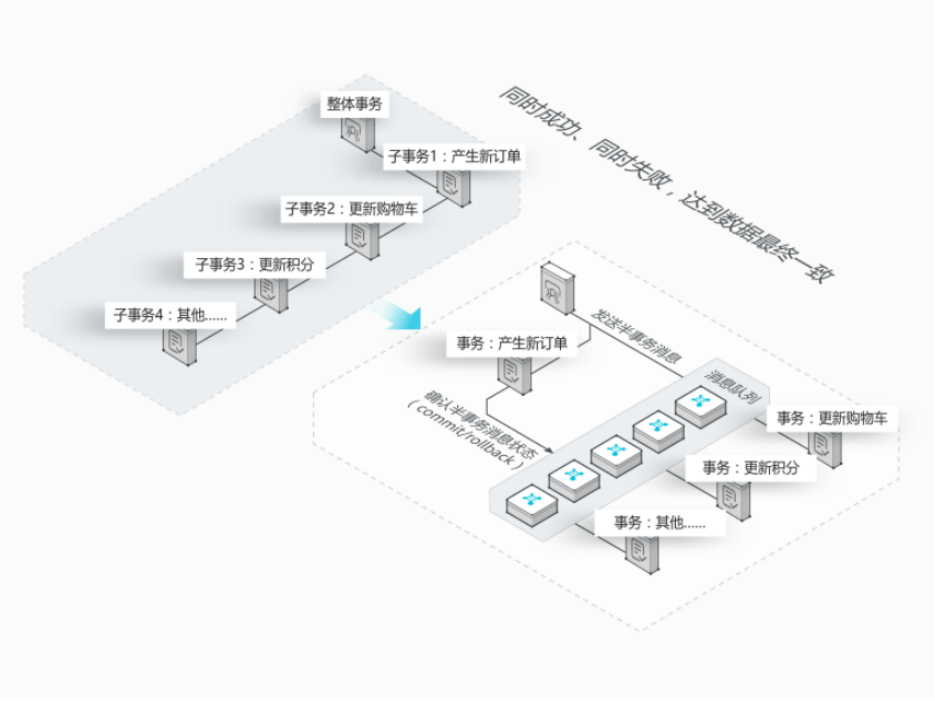
- 支持专有云独立输出，支持物理机和虚拟机，仅几台机器便可搭建完整消息云服务。

- 专有云配套 mqadmin 命令集和管理类 Open API，方便运维人员实时监控系统状态。
- 支持混合云架构，允许用户通过专线的方式接入服务。

消息队列 MQ 可应用于如下几个场景：

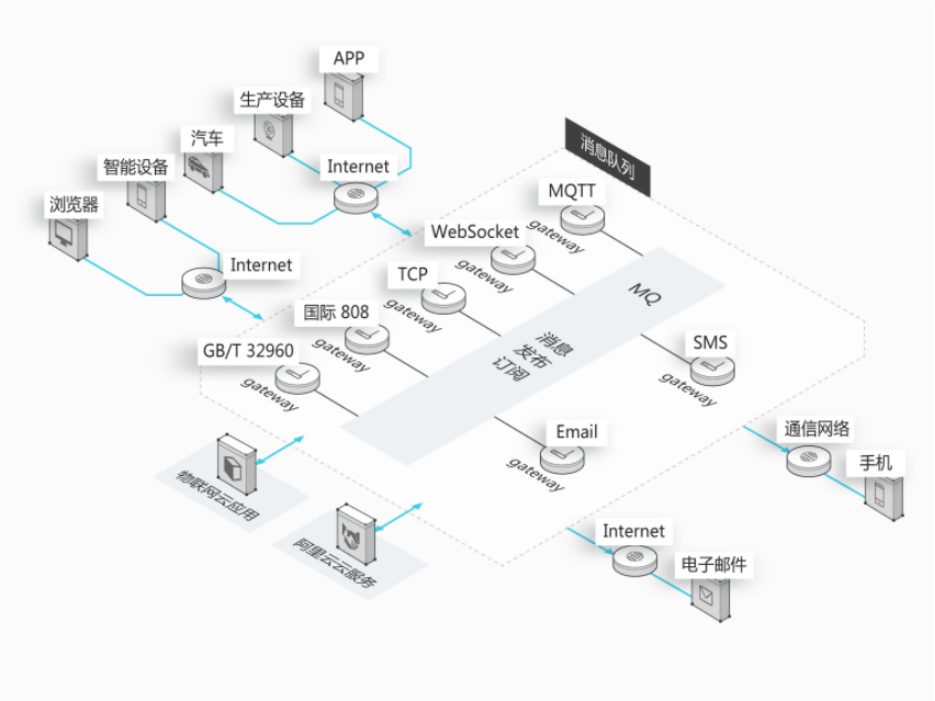
分布式事务

在传统的事务处理中，多个系统之间的交互耦合到一个事务中，响应时间长，影响系统可用性。引入分布式事务消息，交易系统和消息队列之间，组成一个事务处理，能保证分布式系统之间数据的最终一致。；下游业务系统（购物车、积分、其他）相互隔离，并行处理。



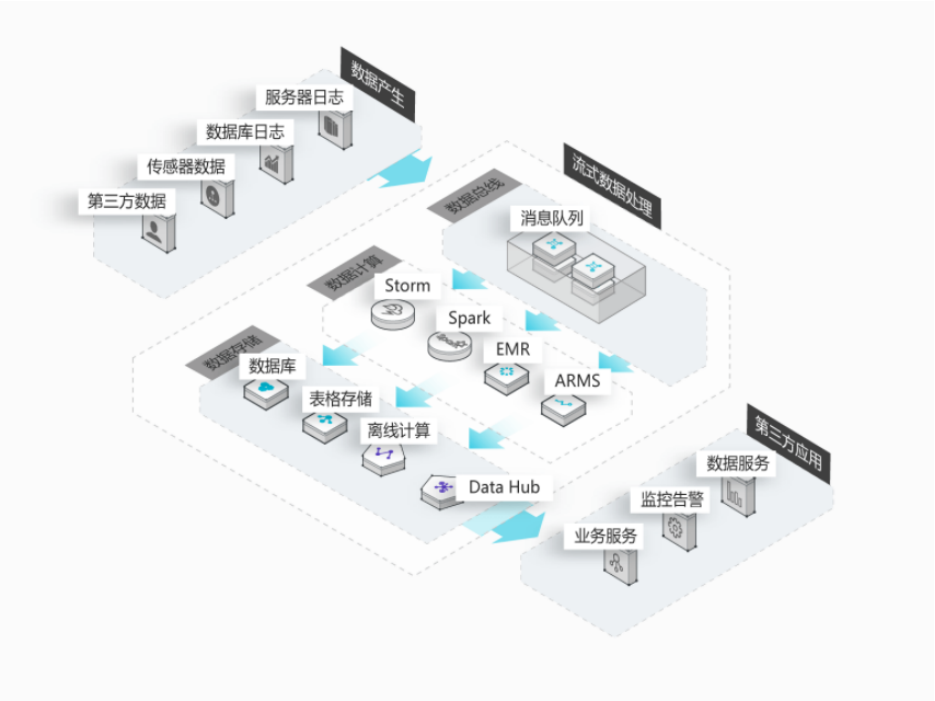
物联网应用

物联网设备通过微消息队列（LMQ）连接云端，双向通信，数据传输；设备数据通过消息队列（MQ）连接计算引擎，分析数据或者源数据实时高效写入到 HiTSDB / HiStore / ODPS 等。



实时计算

通过消息队列（MQ），将源端不停产生的数据实时流入到计算引擎，实现实时计算。可采用如下计算引擎：Spark / Storm / EMR / ARMS / BeamRunner。



大规模缓存同步

在商业大促活动中，如“双11”大促，各个分会场会有琳琅满目的商品，每件商品的价格都会实时变化；同时

，大量并发访问商品数据库，会场页面响应时间长。集中式缓存，带宽成瓶颈，无法满足对商品价格的访问需求。

消息队列（MQ）能够通过大规模缓存同步，减少页面相应时间；针对分会场的多缓存设计，满足客户对商品价格的访问需求。

本文主要对 MQ 涉及的专有名词及术语进行定义和解析，方便您更好地理解相关概念并使用 MQ。

Message Queue

消息队列，阿里云商用的专业消息中间件，是企业级互联网架构的核心产品，提供基于高可用分布式集群技术搭建的消息发布订阅、轨迹查询、资源统计、定时（延时）、监控报警等一系列消息云服务。

Message

消息，消息队列中信息传递的载体。

Message ID

消息的全局唯一标识，由 MQ 系统自动生成，唯一标识某条消息。

Message Key

消息的业务标识，由消息生产者（Producer）设置，唯一标识某个业务逻辑。

Topic

消息主题，一级消息类型，通过 Topic 对消息进行分类。

Tag

消息标签，二级消息类型，用来进一步区分某个 Topic 下的消息分类。

Producer

消息生产者，也称为消息发布者，负责生产并发送消息。

Producer ID

一类 Producer 的标识，这类 Producer 通常生产并发送一类消息，且发送逻辑一致。

Producer 实例

Producer 的一个对象实例，不同的 Producer 实例可以运行在不同进程内或者不同机器上。Producer 实例线程安全，可在同一进程内多线程之间共享。

Consumer

消息消费者，也称为消息订阅者，负责接收并消费消息。

Consumer ID

一类 Consumer 的标识，这类 Consumer 通常接收并消费一类消息，且消费逻辑一致。

Consumer 实例

Consumer 的一个对象实例，不同的 Consumer 实例可以运行在不同进程内或者不同机器上。一个 Consumer 实例内配置线程池消费消息。

集群消费

一个 Consumer ID 所标识的所有 Consumer 平均分摊消费消息。例如某个 Topic 有 9 条消息，一个 Consumer ID 有 3 个 Consumer 实例，那么在集群消费模式下每个实例平均分摊，只消费其中的 3 条消息。

广播消费

一个 Consumer ID 所标识的所有 Consumer 都会各自消费某条消息一次。例如某个 Topic 有 9 条消息，一个 Consumer ID 有 3 个 Consumer 实例，那么在广播消费模式下每个实例都会各自消费 9 条消息。

定时消息

Producer 将消息发送到 MQ 服务端，但并不期望这条消息立马投递，而是推迟到在当前时间点之后的某一个时间投递到 Consumer 进行消费，该消息即定时消息。

延时消息

Producer 将消息发送到 MQ 服务端，但并不期望这条消息立马投递，而是延迟一定时间后才投递到 Consumer 进行消费，该消息即延时消息。

事务消息

MQ 提供类似 X/Open XA 的分布事务功能，通过 MQ 事务消息能达到分布式事务的最终一致。

顺序消息

MQ 提供的一种按照顺序进行发布和消费的消息类型, 分为全局顺序消息和分区顺序消息。

顺序发布

对于指定的一个 Topic，客户端将按照一定的先后顺序进行发送消息。

顺序消费

对于指定的一个 Topic，按照一定的先后顺序进行接收消息，即先发送的消息一定会先被客户端接收到。

全局顺序消息

对于指定的一个 Topic，所有消息按照严格的先入先出（FIFO）的顺序进行发布和消费。

分区顺序消息

对于指定的一个 Topic，所有消息根据 sharding key 进行区块分区。同一个分区内的消息按照严格的 FIFO 顺序进行发布和消费。Sharding key 是顺序消息中用来区分不同分区的关键字段，和普通消息的 key 是完全不同的概念。

消息堆积

Producer 已经将消息发送到 MQ 服务端，但由于 Consumer 消费能力有限，未能在短时间内将所有消息正确消费掉，此时在 MQ 服务端保存着未被消费的消息，该状态即消息堆积。

消息过滤

订阅者可以根据消息标签（Tag）对消息进行过滤，确保订阅者最终只接收被过滤后的消息类型。消息过滤在 MQ 服务端完成。

消息轨迹

在一条消息从发布者发出到订阅者消费处理过程中，由各个相关节点的时间、地点等数据汇聚而成的完整链路信息。通过消息轨迹，用户能清晰定位消息从发布者发出，经由 MQ 服务端，投递给消息订阅者的完整链路，方便定位排查问题。

重置消费位点

以时间轴为坐标，在消息持久化存储的时间范围内（默认3天），重新设置消息订阅者对其订阅 Topic 的消费进度，设置完成后订阅者将接收设定时间点之后由消息发布者发送到 MQ 服务端的消息。

中继

中继服务（Relay）是由消息队列(MQ)提供的服务发布与订阅组件，该组件的主要作用是在不同的网络环境下实现服务之间互联互通的能力。

Java SDK 版本说明

ons-client V1.7.0.Final 2017年10月23日

新特性

- 调整客户端消息缓存策略，考虑消息条数与缓存大小两个维度。

功能优化

- 优化客户端内置轨迹模块的 ProducerName，不同的用户使用不同的值；
- 优化实例配置，单时间线最大 Tag 支持扩大到 16 个。

问题修复

- 修复客户端 Trace 线程阻止客户端正常退出的问题；
- 修复 MQ 消息轨迹 ShutDownHook 可能重复创建的问题。

ons-client V1.6.1 2017年8月31日

功能优化

- 为所有的客户端 API 添加了详细的 Javadoc;
- 优化获取客户端地址的方式，不依赖 /etc/hosts 中的 hostname 配置。

ons-client V1.6.0.Final 2017年7月31日

新特性

- 客户端在源码级别进行 Shade，保证 Debug 的正确性;
- 客户端暴露 BornHost、BornTimestamp 消息属性;
- 新增 BatchConsumer 接口，允许用户以批量的方式消费消息;
- 新增顺序消息、BatchConsumer 与 Spring 集成的 Demo。

功能优化

- 针对分区有序消息，将 Sharding Key 放入到消息结构中;
- 消息属性设置支持 Int 型的 Value 值。