# 腾讯云消息队列CMQ-分布式消息队列 分布式消息系统

| | |
|---|---|
| 产品名称 | 腾讯云消息队列CMQ-分布式消息队列<br>分布式消息系统 |
| 公司名称 | 昆山昱唯网络科技有限公司 |
| 价格 | 5.00/1000次 |
| 规格参数 | 品牌:腾讯云<br>产品:消息队列CMQ |
| 公司地址 | 花桥国际商务城曹新路70号 |
| 联系电话 | 17601404160 |

# 产品详情

CMQ 消息队列介绍

CMQ（Cloud Message Queue）是基于腾讯自研消息引擎的分布式消息队列系统，CMQ 通过腾讯自研分布式 Raft 算法保证消息强一致，消息同步3副本落盘保障消息高可靠，提供消息队列、发布订阅、消息回溯、延时消息、顺序消息、消息轨迹等服务。具有高可靠、高可用、高性能、动态伸缩等优势。 CMQ 的开发迭代历程超过7年，为腾讯内部包括微信、webank、QQ 秀、手机 QQ 等大型业务提供异步的消息服务支持。

CMQ 已经正式商用，目前在腾讯云全球多个地域提供了高可用消息云服务，机房硬件设施按照腾讯自建 IDC 的高标准来实施。单个地域内采用多机房部署，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。同时在深圳金融专区、上海金融专区均有部署，提供金融级数据高可靠消息队列服务。

CMQ 目前提供 HTTP(S)、TCP 协议的接入。提供 PHP、Java、Python 等丰富语言的 SDK 接入。

接入方式HTTP(S) 接入TCP 接入面向场景提供基于 HTTP(S) 协议的同步接入方式，支持 Restful API 和多语言的 SDK 简单方便接入。提供 TCP 同步/异步接入方式，支持多语言 SDK，提高生产端和消费端的效率，提供更高性能的消息队列服务。亮点能力消息无限堆积、水平扩展金融级消息高可靠，消息实时落盘。TCP 异步非阻塞方式收发消息，提升效率。消息无限堆积、水平扩展金融级消息高可靠，消息实时落盘。

说明：

CMQ TCP 接入方式正在内测中，欢迎联系售后工程师，将有专人为您开通内测服务。

CMQ 支持私有化部署，欢迎联系售后工程师，将有专人为您开通内测服务。

应用场景概述

在需要进行异步通信的应用情景中推荐使用 CMQ。例如：

应用需要确保消息的可靠传递，即使发送消息时接收者由于断电、宕机或 CPU 负载过高等原因不可用，消息也可以在接收者可用时被送达。传统的消息队列把消息储存在内存中，故而不具备这一功能。腾讯云 CMQ 分布式消息队列中的消息会被持久化保存，直到接收者成功获取它。

需要在访问量与日俱增、囤积在队列中的消息数日益增长的情况下也能正常运转。传统的消息队列把消息存在本地内存中，单机的处理能力和内存容量都是有限的，不具备可扩展性。腾讯云 CMQ 的分布式架构保证了其扩容的简易性，更重要的是扩容对 CMQ 的使用者是完全透明的。

两个服务在网络不能互通或者应用的路由信息（例如 IP 和端口）不确定的情况下需要通信。例如，两个腾讯云上的服务在不知道对方地址的情况下需要进行通信，则可以通过约定队列名，一个向队列发送消息，一个从队列中收取消息而实现。

系统组件之间或者应用之间通信较多，需要组件或者应用自身维护彼此的网络连接，而且通信的内容不仅一种。这时，使用传统的架构会使得系统设计复杂。例如：当有一个中央处理服务需要向多个任务处理服务分派任务时（类似于 master-worker 模式），master 需要维护与所有 worker 的连接，并判断 worker 是否开始处理任务从而决策是否需要重新分派任务。同时，worker 的任务结果也需要汇报给 master。要在一个层面维护这样的系统会导致设计复杂，实现难度和维护成本大。如下图所示，使用腾讯云 CMQ 减轻两方之间的耦合性会使系统简洁高效许多。

系统组件之间或者应用之间耦合较紧，尤其对依赖的组件可控性较弱的情况下，希望降低耦合度。例如公司业务 CGI 收到用户提交的内容，将部分数据存储在自身的系统中，并将处理后的数据转发给其他业务应用（如数据分析系统、数据存储系统等）。传统的解决方案是服务间通过 socket 建立连接，此时如果接收方的 IP 或端口改变，或换了另一个接收方，则需要数据发送者进行修改。使用腾讯云 CMQ，发送者和接收者对彼此信息无感知，耦合度大大降低。

在需要进行异步通信的应用情景中推荐使用 CMQ。例如：

系统组件之间或者应用之间通信较多，需要组件或者应用自身维护彼此的网络连接，而且通信的内容不仅一种。这时，使用传统的架构会使得系统设计复杂。例如：当有一个中央处理服务需要向多个任务处理服务分派任务时（类似于 master-worker 模式），master 需要维护与所有 worker 的连接，并判断 worker 是否开始处理任务从而决策是否需要重新分派任务。同时，worker 的任务结果也需要汇报给 master。要在一个层面维护这样的系统会导致设计复杂，实现难度和维护成本大。如下图所示，使用腾讯云 CMQ 减轻两方之间的耦合性会使系统简洁高效许多。

CMQ 的 Topic
主题订阅模型，支持生产者向多个订阅者，同时异步投递消息的能力。您可将消息投递到不同 HTTP 终端，或投递到 Queue 队列内。CMQ 还提供消息、订阅 TAG，提供消息过滤能力。

说明：

Topic 不支持投递到 VPC 下的 HTTP 终端。

参考图示如下：

CMQ 支持将queue/topic模式混合使用。典型的场景是client端发起异步调用请求，后端为重逻辑，无法实时反馈结果。传统的做法是client端多次重复轮询。使用CMQ添加订阅，可配置在后端重逻辑完成时，将通知投递给client端用户

参考图示如下：

以电商的IT架构作为例子，在传统紧耦合订单场景里，客户在电商网站下订单（如买一台手机），订单系统接收到请求后，立即调用库存系统接口，库存减一；但这种模式存在巨大风险：

订单系统与库存系统强耦合，假如库存系统无法访问（升级、业务变更、故障等），则订单减库存将失败，从而导致订单失败；

传统的解决方案是服务间通过订单系统与库存系建立 socket 连接，但是如果库存系统的 IP/端口变更、增加库存系统的接收者，都需要订单系统进行修改；

短时间内大量的请求，对库存系统的 SQL，频繁查询库存，修改库存，库存系统负载极大；

用户的感受：订单失败，重试，依然失败，导致顾客流失。

引入 CMQ 后的方案如下图：其中几个系统分别工作，解除强耦合性：

订单系统：用户下单后，订单系统完成持久化处理，将消息写入消息队列，返回用户订单下单成功。此时客户可以认为手机已经买到了。CMQ 提供异步的通信协议，消息的发送者将消息发送到消息队列后可以立即返回，不用等待接收者的响应。消息会被保存在队列中，直到被接收者取出。

库存系统：从 CMQ 获取下单信息，库存系统根据下单信息进行库存操作。

这样，哪怕用户在下单时库存系统宕机，也不影响正常下单（库存系统修复后再从 CMQ 中取出订单进行操作）。订单系统写入腾讯云 CMQ 消息队列后，就无需再关心其他后续操作了。实现订单系统与库存系统的应用解耦。

像电商这样需要保证消息被可靠传递的业务，即使发送消息（订单系统）时，接收者（库存系统）由于断电、宕机或 CPU 负载过高等原因不可用，消息也可以在接收者恢复可用时被送达。腾讯云 CMQ 的分布式消息队列存储保证了消息的持久化保存，直到接收者成功获取它，而不用担心某些消息队列方案存储在单机内存中而导致的数据丢失。

某电商网站新手机发布在即，拥有预约码的用户可优先购买手机。预约方式为：注册账户即可获得预约码，预计预约用户超过1000万。

像双11秒杀、手机预约抢购等对 IO 时延敏感业务环境下，当外部请求超过系统处理能力时，如果系统没有做相应保护，可能由于历史累计的超时请求负荷过多而导致系统处理的每个请求都因超时而无效，系统对外呈现的服务能力为0，且这种情况下服务不能自动恢复。这种情形下，引入腾讯云消息中间件 CMQ，将非即时处理的业务逻辑进行异步化。例如服务接收请求、处理请求和返回请求三个不同的业务逻辑。

引入 CMQ 后，当预约活动开始时，海量并发访问汹涌袭来：

所有客户的预约申请，页面均立即返回成功。客户便可关闭网页进行其他活动。预约码稍后推送到客户的邮箱/手机；

超过千万级别的注册、预约申请，先暂存在腾讯云 CMQ 消息队列集群；

后端服务进行处理，按照数据库实际的 select、insert、update 能力处理注册、预约申请；

处理成功后返回结果给用户。预约结束后，用户大约在5 - 30min内，都收到了预约码。

继续上文使用电商的例子。假设客户下单购买了一台新手机，关联的子系统有以下系列动作：

付款确认后，赠送会员成长值

赠品系统同步给客户发送赠品

优惠券系统在订单完成三个月后发送折扣券至用户

客户ERP系统记录客户购买行为进行分析

可以发现这些任务都是独立的，互相之间没有依赖关系，即不需等待其他模块的结果就可独立执行。引入腾讯云 CMQ，能带来以下价值：

CMQ 保证了消息被可靠传递：即使发送消息时，接收者由于断电、宕机或 CPU 负载过高等原因不可用，CMQ系统也保证消息在接收者可用时被送达。CMQ的分布式消息队列，消息会被持久化保存，直到接收者成功获取它；

生产一次数据可被不同消费场景同时消费，多次复用。例如订单数据生产一次，在 CMQ 中持久保存并可被逻辑、业务、计费、监控、统计等多个模块消费；

可根据不同的业务特性，自定义消息的生命周期，对消息进行延迟处理、多次处理。

当电商系统架构逐渐成长，差异性带来的问题可能逐渐凸显出来：假如订单系统（order_module）采用 Java 架构，库存系统（ inventory_module）采用 Erlang 架构，而发货系统使用的是 Python 架构.....使用传统的解决方案时，开发人员需要长期维护一些冗余的代码来将各个模块间传入的 HTTP 请求转化为应用程序中的函数调用。当引入腾讯云 CMQ 服务后：

可以屏蔽不同平台，不同编程语言之间的差异。CMQ 提供标准 Restful API 接入，系统之间的数据交互变得异常简单。

提供主流的 C++、PHP、Python 等多种语言 SDK。只需安装 SDK 即可轻松使用腾讯云 CMQ。

腾讯云 CMQ 还能很好地支持跨 IDC 机房、跨用户之间的数据交换需求。若需要通信的双方在网络不能互通或者应用的路由信息（例如 IP 和端口）不确定的情况下（例如两家公司的业务都部署在腾讯云上，但彼此都不知道对方服务地址），可以通过约定相同的队列名，一个向队列中发消息，一个从队列中收消息来进行数据交换。

腾讯云 CMQ 支持跨机房的数据交换能力。无需打通企业 A 和 B 的内网，公司内部局域网也无须暴露在公网环境下，轻松实现企业 A 向企业 B 数据交换、同步的需求。

腾讯云 CMQ 提供 HTTPS+密钥 的形式，满足公网数据交换的需求，兼顾了数据安全与效率。

异步通信协议

消息的发送者将消息发送到消息队列后可以立即返回，不用等待接收者的响应。消息会被保存在队列中，直到被接收者取出。消息的发送与处理是完全异步的。

提高可靠性

传统模式下消息可能因为长时间等待而导致请求失败。消息队列模式下，如果发送消息时接收者不可用，消息队列会保留消息直到成功传递它。

进程解耦

消息队列帮助减少两个进程间的耦合度。只要消息格式不变，即使接收者的接口、位置或者配置改变，也不会给发送者带来任何改变。并且，消息发送者无需知道消息接收者是谁，使得系统设计更清晰；相反的，进程间使用远程过程调用（RPC）或者 socket 连接，当一方接口、IP
或端口改变了，另一方则必须修改请求配置。

消息路由

发送者无需与接收者建立直接连接，双方通过消息队列保证消息能够从发送者路由到接收者，甚至对于本来网络不易互通的两个服务，也可以提供消息路由。

多终端

用户系统的多个部分可以同时发送或接收消息，腾讯云 CMQ 通过消息状态来进行消息可用性的控制。

多样性

每个队列均可独立配置，并非所有队列都要完全相同。在不同业务场景下的队列可以进行个性化的配置，例如一个队列中消息处理时间较长，可以针对队列属性进行优化。

云函数触发

CMQ 主题模型可将消息传递给云函数 SCF，并将消息内容和相关信息作为参数来调用该函数。

对比 RabbitMQ 的优势

CMQ-QPS 优势：在保证高可靠前提下，同等物理设备，CMQ 的吞吐量优于 RabbitMQ
四倍以上。单集群 QPS 超过10万。

RabbitMQ 不支持消息回溯：RabbitMQ 不支持消息回溯，CMQ
支持按照时间回溯消息。例如从一天之前的某时某分某秒开始重新消费消息。典型业务场景如 Consumer 做订单分析，但是由于程序逻辑或者依赖的系统发生故障等原因，导致今天消费的消息全部无效，需要重新从昨天零点开始消费，那么以时间为起点的消息重放功能对于业务非常有帮助。

一致性算法对比：CMQ 和 RabbitMQ 都能够使用多台机器进行热备份，提高可用性。CMQ 基于 Raft
算法实现，简单易维护。RabbitMQ 使用自创的 GM 算法（Guaranteed Multicast），学习难度较高。

RabbitMQ 运维难度大：RabbitMQ 的开发语言用的是 Erlang，较小众、学习成本高。

对比 RocketMQ 的优势

RocketMQ 在极端情况下，会丢失数据：RocketMQ
允许未刷盘就向客户端返回确认，在机器异常宕机时，会丢消息。

RocketMQ 需搭建多 Master、Slave 才能保证业务高可用：RocketMQ 只有在 ISR
中有存活节点时，才能保证可用性和可靠性，ISR
中无存活节点时，可用性和可靠性无法保证，开销较大。

因此，相比传统开源 MQ 应用，腾讯云 CMQ 具有以下优势：

腾讯云消息队列开源消息中间件软件高性能兼顾性能与可靠性，单 CMQ 实例 QPS
达到5000数据可靠性与性能无法兼顾高扩展性

队列数量及队列存储容量可扩展性强

底层系统根据业务规模，自动弹性伸缩，上层业务无感知

高效支持亿级消息收发、推送，堆积，容量不设上线

提供北京、上海、广州地域的多地域服务

队列数量和消息堆积数量有限

每个 IDC 机房必须重新部署购买设备、部署，非常繁琐

高可靠性

基于腾讯自研 CRMQ（Cloud Reliable Message Queue）分布式框架，已在腾讯内部 QQ
微信、彩票等业务上得到广泛使用

消息服务每条消息在返回给用户写成功之时就确保数据已被复制3份写到不同物理机上，并且后台数据复制机制能够保证任何一台物理机故障时其上的数据能够快速的做迁移，时刻保证用户数据3份 copy
可用，可靠性达99.999999%

引入改良后的 Raft 一致性算法，保证数据强一致性

业务可用性承诺：99.95%

数据单机或简单主从结构，存在数据单点问题，一旦丢失不可回溯

开源的 replicia
算法，在集群新增、删除服务器节点时，会引发全局的数据重新均衡，引起可用性急剧下降

如 Kafka 使用异步刷盘方式，异步 Replication，无法保证数据强一致性

业务安全

多纬度的安全防护和防 DDoS 攻击服务

每个消息服务提供单独命名空间，客户间数据严格隔离

支持 HTTPS 访问

支持跨地域的安全消息服务

安全防护功能有限

考虑到公网的网络威胁，经常无法提供跨地域、跨 IDC 的公网服务