

腾讯云消息队列 CMQ

产品简介

产品文档



腾讯云

【版权声明】

©2013-2017 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】



及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

文档声明.....	2
产品简介.....	4
消息队列概述.....	4
应用场景.....	8
应用场景概述.....	8
一对多生产.....	10
异步通知.....	11
系统解耦.....	12
削峰填谷.....	14
可靠传递，多次复用.....	15
屏蔽平台差异.....	16
跨用户数据交换.....	17
产品功能.....	18
产品优势.....	19
服务等级协议（SLA）.....	22

产品简介

消息队列概述

CMQ产品系列介绍

CMQ(Cloud Message Queue) 是腾讯云基于开源消息引擎自研的一个是分布式消息系统，消息生产实时3副本落盘、分布式Raft算法保证消息强一致，提供消息发布订阅、消息回溯、消息一对多投递、顺序消息等服务。具有高可靠、高可用、跨IDC、透明动态伸缩、消息接近生产消费等优势。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 是腾讯云正式商用的产品，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。在深圳、上海金融专区，提供同城双活、多活的部署方案，为金融核心交易系统保驾护航。

CMQ 目前提供 HTTPS、MQTT、MQTT等多种协议的接入，兼容开源RabbitMQ、RocketMQ、Kafka的客户端接口。提供php、java、python等丰富语言的SDK接入。

项目	CMQ (HTTPS接入)	CMQ (TCP接入)	CKafka	CMQ(MQTT接入)
面向场景	金融级高可靠消息队列，适用于核心架构中系统间解耦、异步调用等场景。在腾讯云上提供https接入，并提供多语言的SDK接入支持。	产品功能、客户端接口完全兼容开源RocketMQ等开源产品，快速迁移上云“0门槛”！	CMQ团队已是 Kafka 开源生态的主要代码贡献者。开发者通过简单修改ip地址即可无缝迁移到腾讯云上（支持0.8、0.9、0.10等主流版本）。为开发者提供更安全、更可靠、免除运维的 Kafka 企业级消息服务	面向智能家居、工业互联网领域开发者，CMQ提供MQTT等物联网协议的完美支持
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 提供跨机房双活容灾	基于CMQ存储引擎，消息高可靠实时落盘 兼容RabbitMQ 3.6.11版本、RocketMQ	超高读写性能，同等设备情况下是kafka开源版本2倍+性能。对比购买虚拟机搭建kafka，节省30-50%成	提供基于MQTT的发布/订阅消息模式，支持1对多、1对1消息交互 提供TLS/SSL安全加

项目	CMQ (HTTPS接入)	CMQ (TCP接入)	CKafka	CMQ(MQTT接入)
	能力 支持顺序消息、事务消息、死信队列、消息回溯、消息加密等高级能力	4.1.0版本 开源原生版本客户端轻量配置，即可上云	本 免除运维烦恼，自持水平扩容，offset堆积无上限 提供上下游生态的技术咨询服务，针对您的Hadoop、Spark等实时、离线计算平台，提供快速接入、故障定位，性能调优等服务	密通信。支持百万级海量客户端长连接访问 支持上下线通知、Last will等亮点功能

注：

1、CMQ

(TCP接入)、CMQ(MQTT)接入，正在内测中，欢迎工单联系售后工程师，将有专人为您开通内测服务

2、支持私有化部署，欢迎工单联系售后工程师，将有专人为您开通内测服务

产品概述

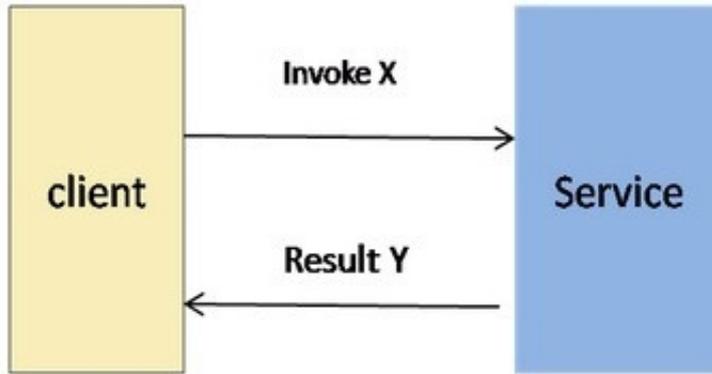
腾讯云消息队列 (Cloud Message Queue，以下简称CMQ) 是分布式的消息队列服务，用于存储进程间传输的消息，为分布式部署的不同应用之间或者一个应用的不同组件之间提供基于消息的可靠的异步通信服务。消息被存储在高可靠、高可用的消息队列中，多进程可以同时读写，互不干扰。使用腾讯云CMQ，用户可以在执行不同任务的应用程序的分布式组件之间传递信息，既不会丢失消息，也不要求各个组件始终处于可用状态。

队列在数据发送端以及数据接收端之间起到缓冲作用。这样，在数据发送端的工作速度快于数据接收端的情况下，或者在数据发送端或数据接收端仅间歇性地连接到网络的情况下，队列可解决因此而产生的问题。

消息队列中的“消息”指同一台计算机/不同计算机的进程间传送的数据，“消息队列”是在消息的传输过程中

保存消息的容器。消息被发送到队列中，队列充当中间人将消息从它的源中继到它的目标。

下图是一个描述传统进程通信模式的例子：传统的进程通信模式由客户端请求服务器端的服务并等待服务器的响应。然而这种模式有很多弊端，比如当网络状况不好的情况下客户端的请求可能会丢失；又如服务器端处理时间过长可能导致客户端长时间等待以致请求超时而失败等。



典型的Invoke/Respond模型

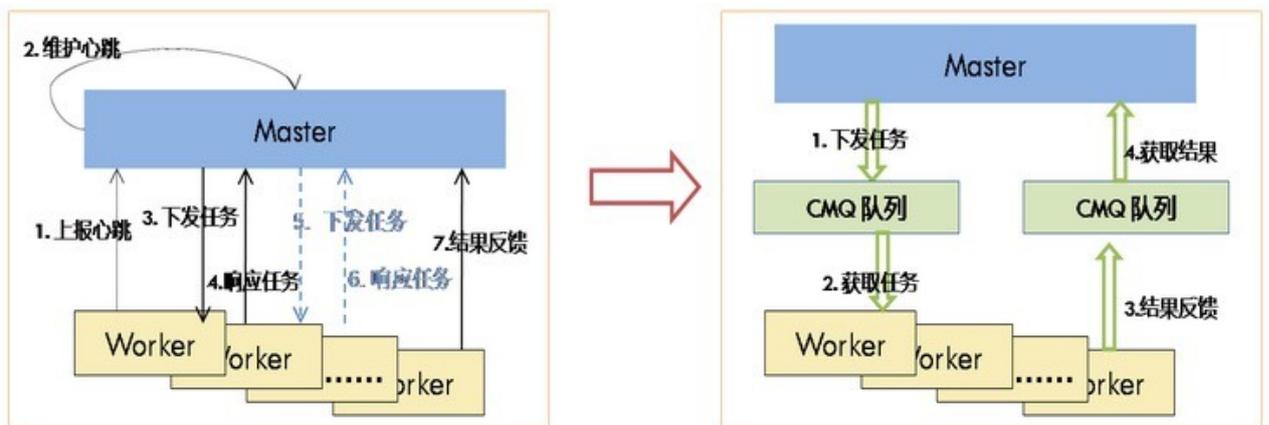
为此，腾讯云引入了消息队列服务进行消息分发和管理。使用腾讯云CMQ，您可以分离应用程序的组件以便其独立运行，同时还可以简化组件间的消息管理。分布式应用程序的任何组件均可将消息存储在队列中，腾讯云CMQ 确保每条消息至少传送一次，并且支持多次读取和写入。单个队列可由多个分布式应用程序组件同时使用而无需这些组件之间的互相协作。所有组件均可使用 CMQ API 以编程方式检索和操作消息。

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（比如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减轻两方的之间耦合性会使系统简洁高效许多。



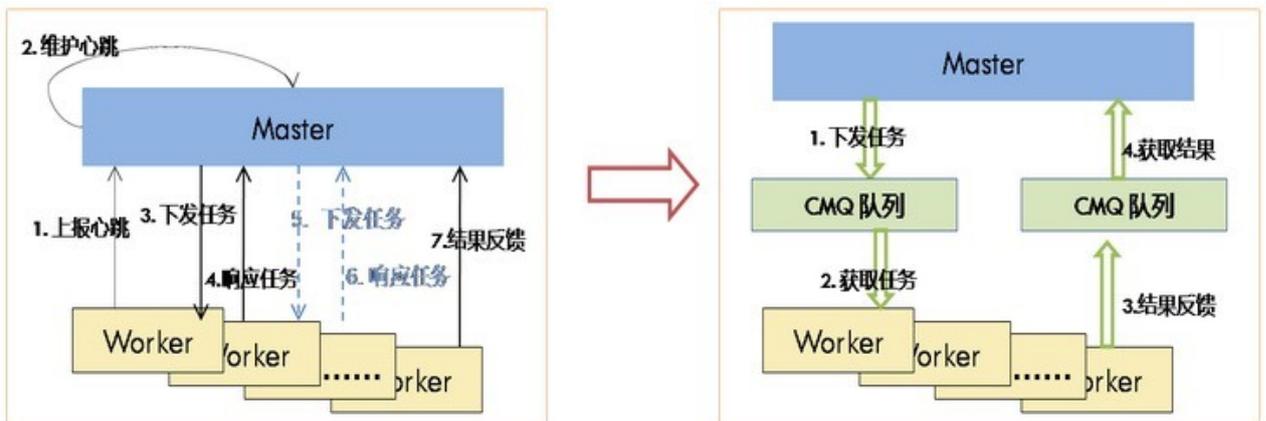
- 系统组件之间或者应用之间耦合较紧，尤其对依赖的组件可控性较弱的情况下，希望降低耦合度。比如公司业务CGI收到用户提交的内容，将部分数据存储在自己的系统中，并将处理后的数据转发给其他业务应用（如数据分析系统、数据存储系统等）。传统的解决方案是服务间通过socket建立连接，此时如果接收方的IP或端口改变，或换了另一个接收方，则需要数据发送者进行修改。使用腾讯云CMQ，发送者和接收者对彼此信息无感知，耦合度大大降低。

应用场景

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（比如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。



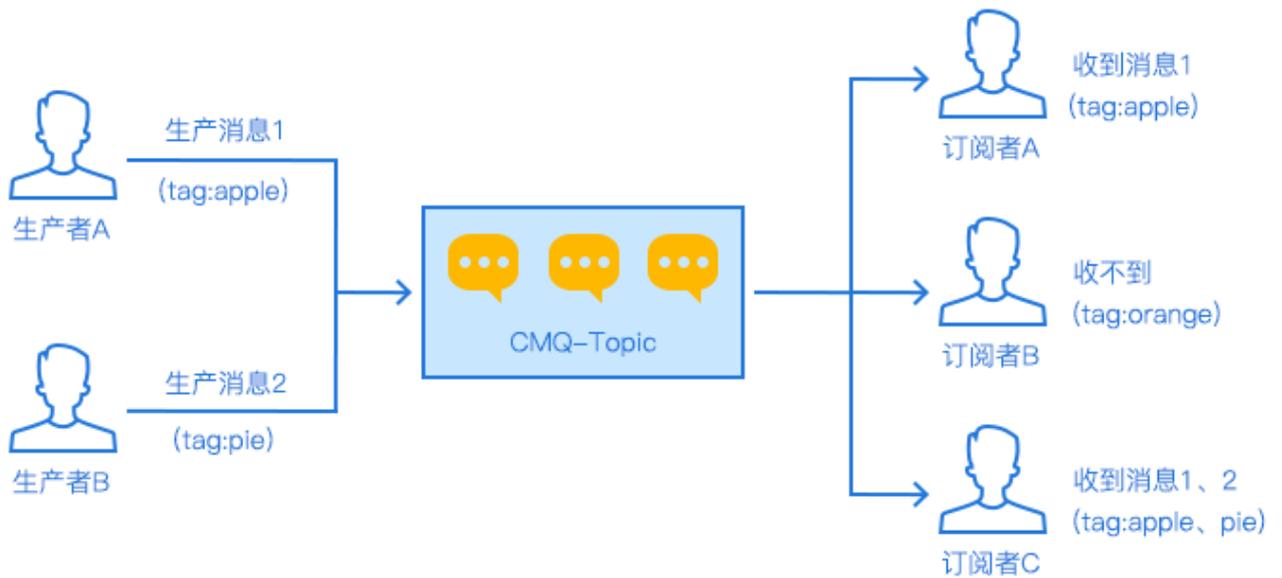
- 系统组件之间或者应用之间耦合较紧，尤其对依赖的组件可控性较弱的情况下，希望降低耦合度。比如公司业务CGI收到用户提交的内容，将部分数据存储在自己的系统中，并将处理后的数据转发给其他业

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。

一对多生产

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/https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。

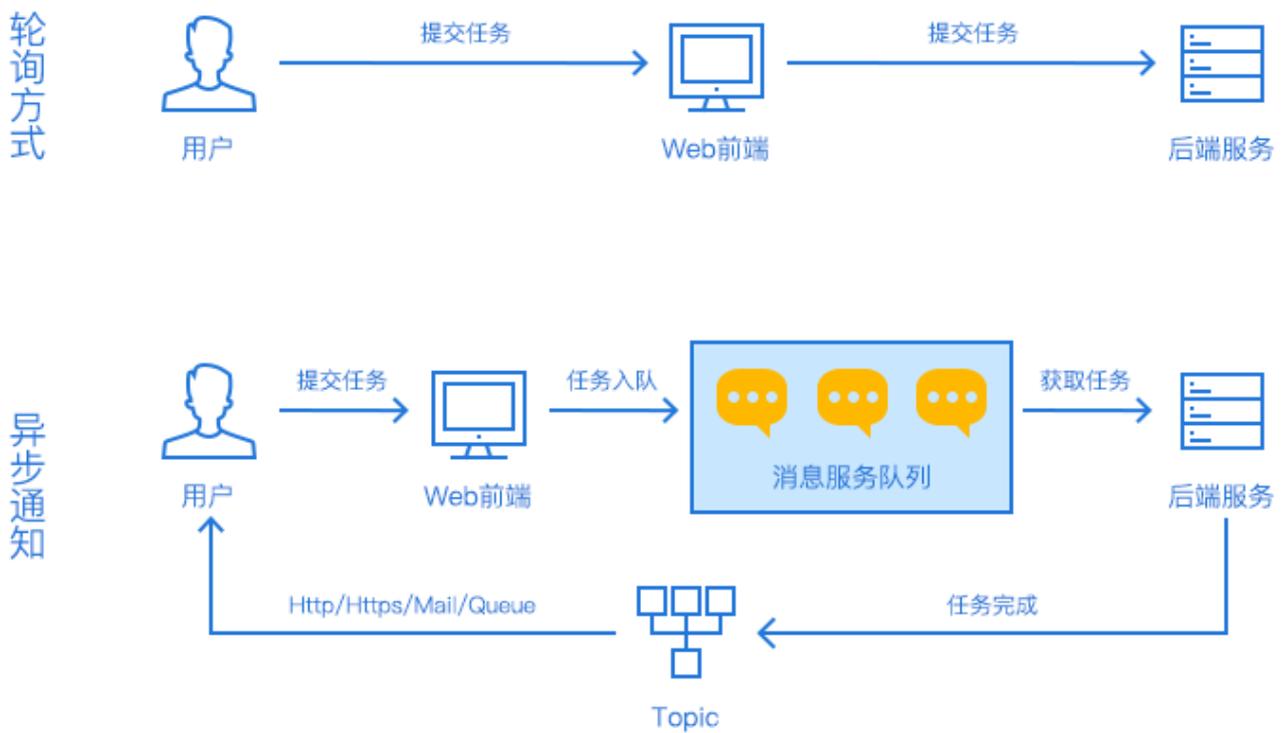
- 参考图示如下：



异步通知

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：

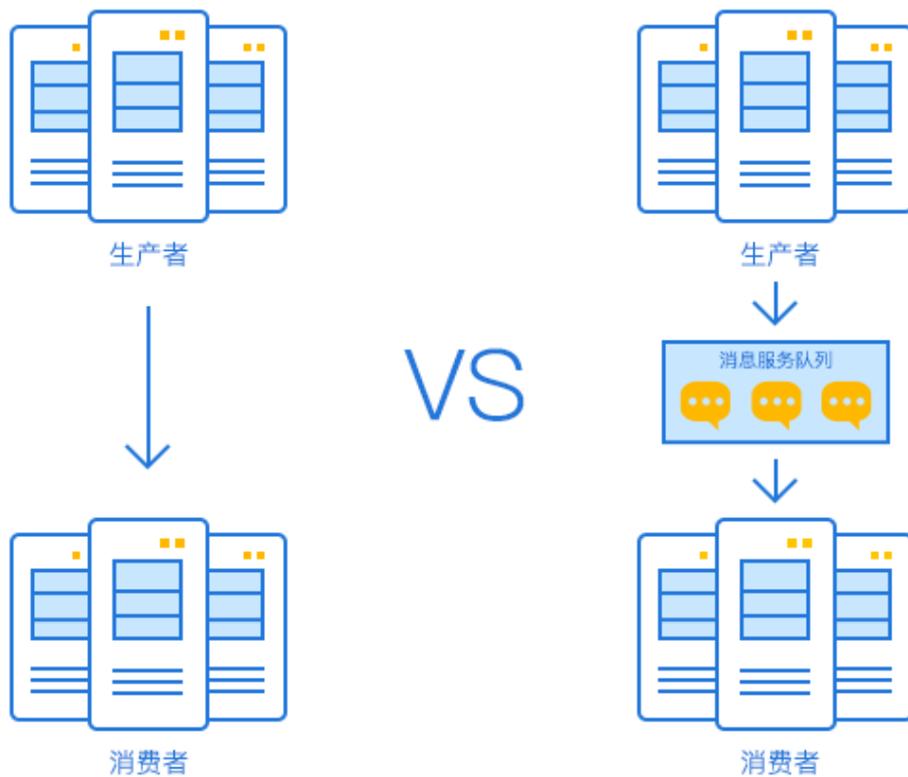


系统解耦

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。

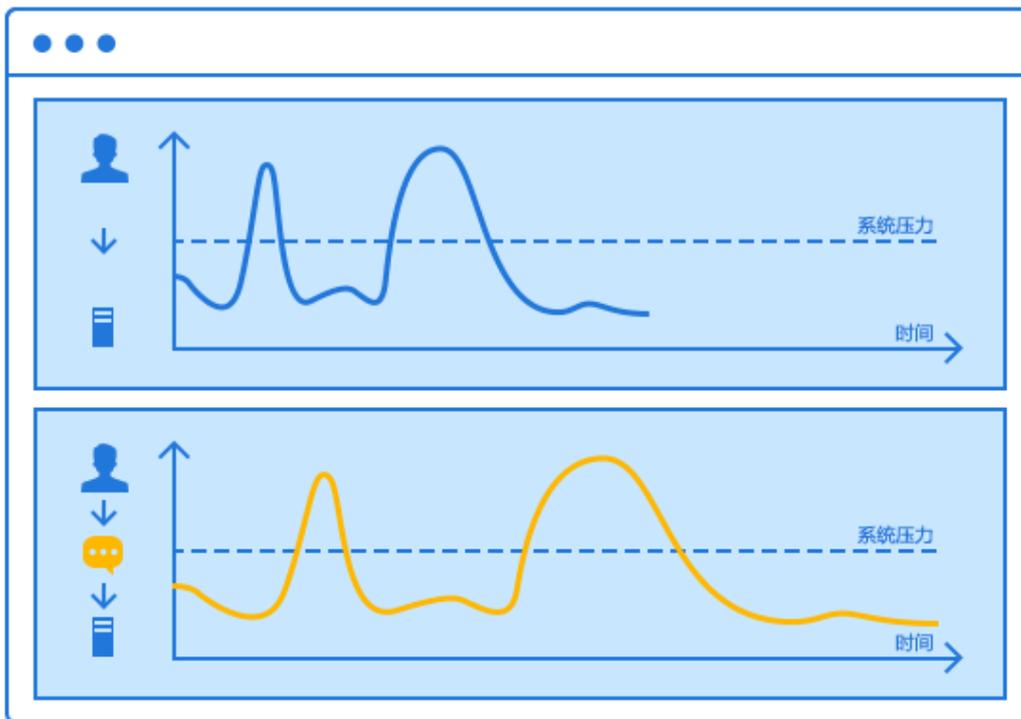
这样，哪怕用户在下单时库存系统宕机，也不影响正常下单（库存系统修复后再从CMQ中取出订单进行操作）。订单系统写入腾讯云CMQ消息队列后，就无需再关心其他后续操作了。实现订单系统与库存系统的应用解耦。

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。

削峰填谷

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。比如服务接收请求、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。

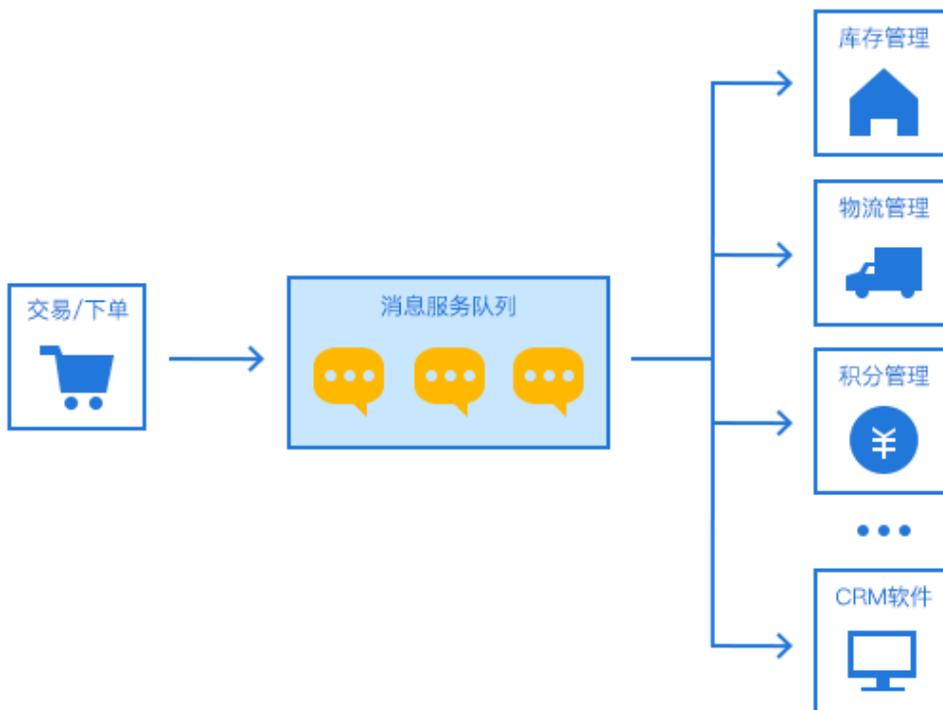
可靠传递，多次复用

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：

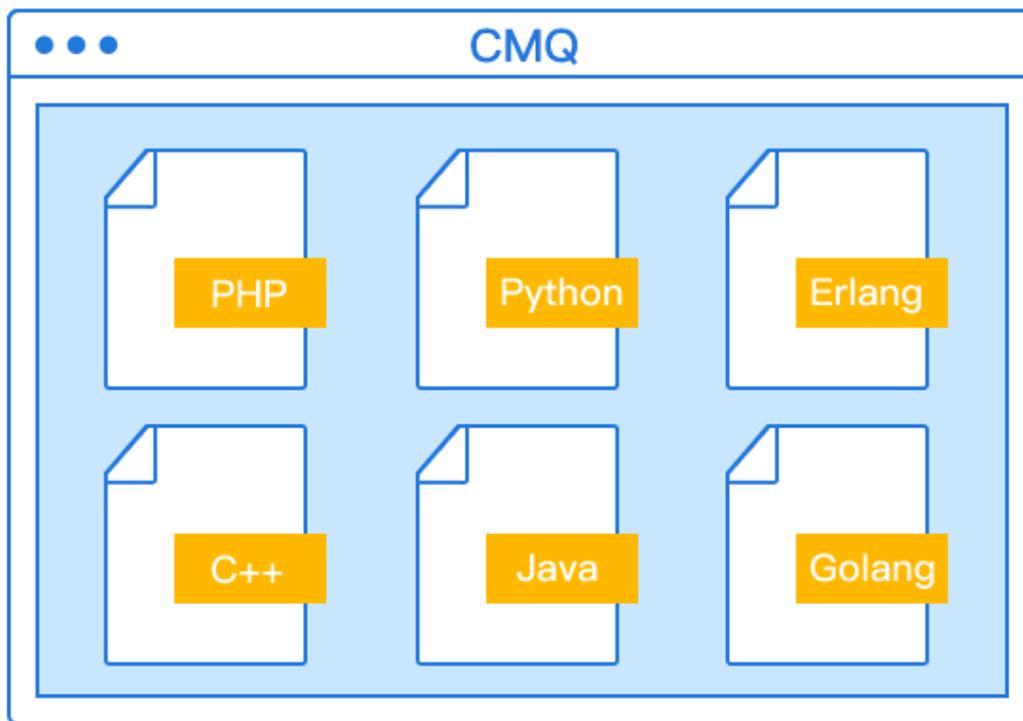
- CMQ保证了消息被可靠传递：即使发送消息时，接收者由于断电、宕机或 CPU 负载过高等原因不可用，CMQ系统也保证消息在接收者可用时被送达。CMQ的分布式消息队列，消息会被持久化保存，直到接收者成功获取它；
- 生产一次数据可被不同消费场景同时消费，多次复用。例如订单数据生产一次，在 CMQ 中持久保存并可被逻辑、业务、计费、监控、统计等多个模块消费；
- 可根据不同的业务特性，自定义消息的生命周期，对消息进行延迟处理、多次处理。



屏蔽平台差异

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构建而成，库存系统 (inventory_module) 采用Erlang，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++、C#、PHP、Golang、Python 等多种语言 SDK。只需安装 SDK 即可轻松使用腾讯云 CMQ

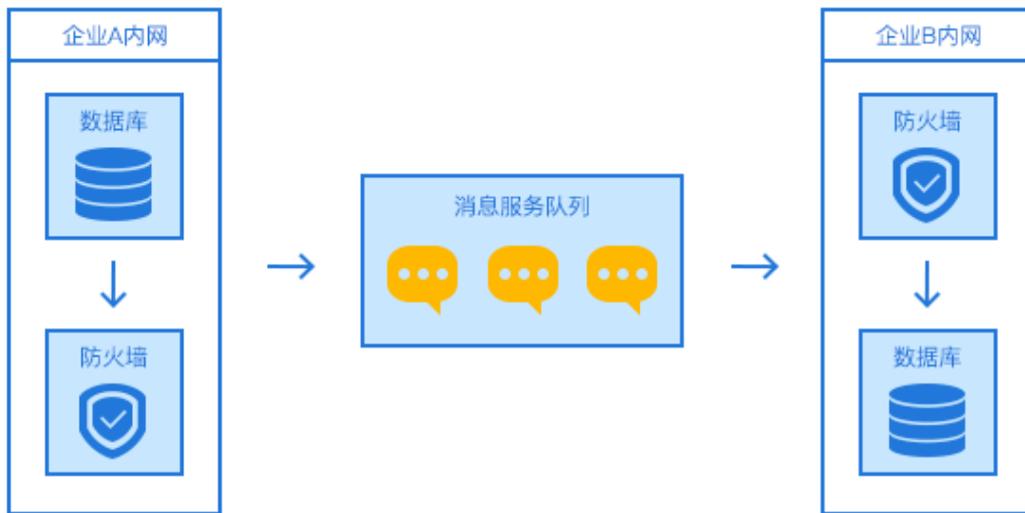


跨用户数据交换

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（比如 IP 和端口）不确定的情况下（比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



产品功能

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，另一方则必须修改改写请求配置。

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。

产品优势

市面上已有林林种种的开源产品，CMQ的产品优势如下：

对比Kafka的优势

1. Kafka不适合金融关键业务：CMQ的同步刷盘机制，保证了数据的可靠生产。客户端生产的消息在set中超过半数的broker刷盘成功后会返回确认消息告知生产消息成功。如果在一定时间之内客户端没有收到确认信息需要重试来确保消息发送成功。Kafka使用异步刷盘方式，异步Replication，没法保证可靠传递，有丢的可能性。
2. kafka 消费失败不支持重试：例如充值类应用，当前时刻调用运营商网关，充值失败，可能是对方压力过多，稍后在调用就会成功，如支付宝到银行扣款也是类似需求。这里的重试需要可靠的重试，即失败重试的消息不因为consumer宕机导致丢失。CMQ支持多次重试消费。
3. Kafka定位问题较繁琐，CMQ提供消息轨迹能力：
消息轨迹指的是一条消息从生产方发出到消费方消费处理，整个过程中的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。便于金融业务快速定位问题。

对比RabbitMQ的优势

1. CMQ-QPS优秀：在保证高可靠前提下，同等物理设备，CMQ 吞吐量优于RabbitMQ的四倍以上。单集群QPS超过10万。
2. RabbitMQ不支持消息回溯：
rabbitMQ不支持消息回溯，CMQ支持按照时间来回溯消息，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。
3. 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。

4. 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

1. 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。
2. 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 微信红包、彩票等业务上得到广泛使用	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

	腾讯云消息队列	开源消息中间件软件
	<p>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</p> <p>可用，可靠性达99.999999%</p> <p>引入改良后的 Raft 一致性算法，保证数据强一致性</p> <p>业务可用性承诺：99.95%</p>	<p>据重新均衡，引起可用性急剧下降</p> <p>如 Kafka 使用异步刷盘方式，异步 Replication，无法保证数据强一致性</p>
业务安全	<p>多纬度的安全防护和防DDoS攻击服务</p> <p>每个消息服务提供单独命名空间，客户间数据严格隔离</p> <p>支持HTTPS访问</p> <p>支持跨地域的安全消息服务</p>	<p>安全防护功能有限</p> <p>考虑到公网的网络威胁，常常无法提供跨地域、跨IDC的公网服务</p>

服务等级协议 (SLA)

1 腾讯云服务

腾讯云服务：指为满足各类网站、应用等各种产品的不同需求，由腾讯云提供的云服务器、云带宽、云存储空间、云数据库、云安全、云监控、云拨测等各种不同要素组合成的云系统服务。具体服务类别以腾讯云公开发布的相关服务信息为准。

2 服务保障指标

腾讯云为您所购买的云服务制定服务等级指标，并向您承诺提供数据管理和业务质量方面最大程度的保障。同时，腾讯云有权根据变化适时对部分指标作出调整。若无特殊约定，本条款中的“月”均指30个自然日，按自然月计算。

2.1 CMQ Queue 服务

2.1.1 数据存储的持久性

每月您申请的 CMQ Queue 的持久性为

99.999999%

。

2.1.2 数据可销毁性

在您要求删除数据或设备在弃置或转售前，腾讯云将采取磁盘低级格式化操作彻底删除您所有数据，并无法复原，硬盘到期报废时将进行消磁。

2.1.3 数据私密性

CMQ 可结合 KMS 加密服务，在消息生产时，将消息 body 加密，避免明文上传。

2.1.4 数据知情权

目前用户 CMQ 服务部署在六大数据中心，分别是：上海数据中心、广州数据中心、北京数据中心、香港数据中心、新加坡数据中心、北美数据中心。

帮助用户选择网络条件最好的数据中心存储数据，用户购买云服务器时刻选择归属的地域（广州、上海、北京、香港、新加坡、北美）。

用户已知的数据中心均遵守的当地的法律和中华人民共和国相关法律。

用户所有数据不会提供给任意第三方，除政府监管部门监管审计需要。用户所有数据不会存在国外数据中心或用于国外业务或数据分析。

为保证用户数据安全性，腾讯云采用同时存储三份数据副本的存储方式，并定期进行数据冷备操作。

2.1.5 数据可审查性

腾讯云在依据现有法律法规体系下，出于配合政府监管部门的监管或安全取证调查等原因的需要，在符合流程和手续完备的情况下，可以提供云服务器相关信息，包括关键组件的运行日志、运维人员的操作记录、用户操作记录等信息。

2.1.6 业务可用性

CMQ Queue 队列服务承诺

99.95%

的业务可用性，即用户每月可用时间应为

$30 \times 24 \times 60 \times 99.95\% = 43178.4$ 分钟

，即存在

$43200 - 43178.4 = 21.6$ 分钟

的不可用时间，其中业务不可用的统计单元为用户单业务实例。

业务故障的恢复正常时间

5分钟

以下，不计入业务不可用性计算中，不可用时间指业务发生故障开始到恢复正常使用的时间，包括维护时间。超过5分钟的，纳入不可用时间。

2.1.7 故障恢复能力

腾讯云 CMQ Queue 具备故障迁移能力，可在物理服务器故障发生时，自动将业务迁移至新的母机，无需用户参与，保证客户服务的连续性。同时，腾讯云提供专业团队

7x24小时

帮助维护。

3 服务计量准确性

腾讯云服务的费用在用户的选购页面和订单页面均有明确展示，用户可自行选择具体服务类型并按列明的价格进行购买。具体价格以腾讯云官网公布的价格为准，腾讯云按照用户选购的云服务规格和使用时长进行收费。

4 补偿

4.1 适用范围

适用于由腾讯云故障原因导致用户的云服务器不可正常使用或完全不可访问，以及腾讯云故障导致的网站（开发者服务网站）无法访问时，用户要求腾讯云针对事故/故障而进行的补偿。

4.2 补偿标准原则

故障时间 = 故障解决时间 - 故障开始时间。按分钟计算故障时间，故障时间小于1分钟的按

1分钟

计算。

例如，故障时间为

1分01秒

，按

2分钟

算。

CMQ消息队列故障百倍赔偿：

后付费：赔偿方式是补偿现金券，

现金券金额 = 故障 Queue 每天费用 ÷ 24 ÷ 60 × 故障时间（以分钟为单位） × 100

。赔偿现金券的上限，不超过 CMQ 服务的总费用。