**备选方案模板**

Table of Contents

[需求介绍 1](#_Toc74257659)

[需求分析 2](#_Toc74257660)

[复杂度分析 3](#_Toc74257661)

[备选方案 4](#_Toc74257662)

[备选方案评估 4](#_Toc74257663)

# 需求介绍

[需求介绍主要描述需求的背景、目标、范围等]

随着前浪微博业务的不断发展，业务上拆分的子系统越来越多，目前系统间的调用都是同步调用，由此带来几个明显的系统问题：

* 性能问题：当用户发布了一条微博后，微博发布子系统需要同步调用“统计子系统”“审核子系统”“奖励子系统”等共 8 个子系统，性能很低。
* 耦合问题：当新增一个子系统时，例如如果要增加“广告子系统”，那么广告子系统需要开发新的接口给微博发布子系统调用。
* 效率问题：每个子系统提供的接口参数和实现都有一些细微的差别，导致每次都需要重新设计接口和联调接口，开发团队和测试团队花费了许多重复工作量。基于以上背景，我们需要引入消息队列进行系统解耦，将目前的同步调用改为异步通知。

# 需求分析

[需求分析主要全方位地描述需求相关的信息]

5W [5W 指 Who、When、What、Why、Where。Who：需求利益干系人，包括开发者、使用者、购买者、决策者等。When：需求使用时间，包括季节、时间、里程碑等。What：需求的产出是什么，包括系统、数据、文件、开发库、平台等。Where：需求的应用场景，包括国家、地点、环境等，例如测试平台只会在测试环境使用。Why：需求需要解决的问题，通常和需求背景相关]

消息队列的 5W 分析如下：

* Who：消息队列系统主要是业务子系统来使用，子系统发送消息或者接收消息。
* When：当子系统需要发送异步通知的时候，需要使用消息队列系统。
* What：需要开发消息队列系统。
* Where：开发环境、测试环境、生产环境都需要部署。
* Why：消息队列系统将子系统解耦，将同步调用改为异步通知。

1H [这里的 How 不是设计方案也不是架构方案，而是关键业务流程。消息队列系统这部分内容很简单，但有的业务系统 1H 就是具体的用例了，有兴趣的同学可以尝试写写 ATM 机取款的业务流程。如果是复杂的业务系统，这部分也可以独立成“用例文档”]

消息队列有两大核心功能：

* 业务子系统发送消息给消息队列。
* 业务子系统从消息队列获取消息。

8C [8C 指的是 8 个约束和限制，即 Constraints，包括性能 Performance、成本 Cost、时间 Time、可靠性 Reliability、安全性 Security、合规性 Compliance、技术性 Technology、兼容性 Compatibility] 注：需求中涉及的性能、成本、可靠性等仅仅是利益关联方提出的诉求，不一定准确；如果经过分析有的约束没有必要，或成本太高、难度太大，这些约束是可以调整的。

性能：需要达到 Kafka 的性能水平。

成本：参考 XX 公司的设计方案，不超过 10 台服务器。

时间：期望 3 个月内上线第一个版本，在两个业务尝试使用。

可靠性：按照业务的要求，消息队列系统的可靠性需要达到 99.99%。

安全性：消息队列系统仅在生产环境内网使用，无需考虑网络安全；如消息中有敏感信息，消息发送方需要自行进行加密，消息队列系统本身不考虑通用的加密。

合规性：消息队列系统需要按照公司目前的 DevOps 规范进行开发。

技术性：目前团队主要研发人员是 Java，最好用 Java 开发。兼容性：之前没有类似系统，无需考虑兼容性。

# 复杂度分析

[分析需求的复杂度，复杂度常见的有高可用、高性能、可扩展等，具体分析方法请参考专栏前面的内容] 注：文档的内容省略了分析过程，实际操作的时候每个约束和限制都要有详细的逻辑推导，避免完全拍脑袋式决策，具体请参考专栏第 10 期的分析。

高可用

对于微博子系统来说，如果消息丢了，导致没有审核，然后触犯了国家法律法规，则是非常严重的事情；对于等级子系统来说，如果用户达到相应等级后，系统没有给他奖品和专属服务，则 VIP 用户会很不满意，导致用户流失从而损失收入，虽然也比较关键，但没有审核子系统丢消息那么严重。综合来看，消息队列需要高可用性，包括消息写入、消息存储、消息读取都需要保证高可用性。

高性能

前浪微博系统用户每天发送 1000 万条微博，那么微博子系统一天会产生 1000 万条消息，平均一条消息有 10 个子系统读取，那么其他子系统读取的消息大约是 1 亿次。将数据按照秒来计算，一天内平均每秒写入消息数为 115 条，每秒读取的消息数是 1150 条；再考虑系统的读写并不是完全平均的，设计的目标应该以峰值来计算。峰值一般取平均值的 3 倍，那么消息队列系统的 TPS 是 345，QPS 是 3450，考虑一定的性能余量。由于现在的基数较低，为了预留一定的系统容量应对后续业务的发展，我们将设计目标设定为峰值的 4 倍，因此最终的性能要求是：TPS 为 1380，QPS 为 13800。TPS 为 1380 并不高，但 QPS 为 13800 已经比较高了，因此高性能读取是复杂度之一。

可扩展

消息队列的功能很明确，基本无须扩展，因此可扩展性不是这个消息队列的关键复杂度。

# 备选方案

[备选方案设计，至少 3 个备选方案，每个备选方案需要描述关键的实现，无须描述具体的实现细节。此处省略具体方案描述，详细请参考专栏第 11 期]

备选方案 1：直接引入开源 Kafka [此处省略方案描述]

备选方案 2：集群 + MySQL 存储 [此处省略方案描述]

备选方案 3：集群 + 自研存储 [此处省略方案描述]

# 备选方案评估

[备选方案 360 度环评，详细请参考专栏第 12 期。注意备选方案评估的内容会根据评估会议的结果进行修改，也就是说架构师首先给出自己的备选方案评估，然后举行备选方案评估会议，再根据会议结论修改备选方案文档]