oad.com/ ”https://www.jiguang.cn/accounts/login/form”) 的推送，并不是自研的推送，成本太大/维护成本太高，直接买就完事了，当大量推送消息到达时，线程池也有顶不住的时候，比如后台对50w用户进行推送活动相关的消息，线程池肯定扛不住啊，到时候不仅推送业务顶不住，还拖垮了这个服务中其他的业务，业务出现瓶颈，优化机会来了，下面都是改造过程中的一些配置(已脱敏)。

> RocketMQConfig 配置

```
@Configuration
@Import({RocketMQAutoConfiguration.class})
public class RocketMQConfig {
}
```

```
# rocketmq配置
rocketmq:
  name-server:
${rocketmq.server.host:127.0.0.1}:${rocketmq.server.port:9876}
  # 生产者
  producer:
    group: producer_xxxxx
    # 消息发送失败重试次数
    retry-times-when-send-failed: 3
    # 异步消息发送失败重试次数
    retry-times-when-send-async-failed: 3
```

> 生产者

```
@Slf4j
@Component
```

```
public class PushMessageProducerService {

  private static final Gson GSON = new Gson();

  @Autowired
  private RocketMQTemplate rocketMQTemplate;

  public void asyncPushToUser(PushContents pushContents, int userId,
      int appFlag, int activityId, boolean passThough, int pushType) {
    PushToUserDto pushMessage = new PushToUserDto();
    pushMessage.setPushContents(pushContents);
    pushMessage.setUserId(userId);
    pushMessage.setAppFlag(appFlag);
    pushMessage.setActivityId(activityId);
    pushMessage.setPassThough(passThough);
    pushMessage.setPushType(pushType);

    String requestId = ThreadContext.get("requestId");
    pushMessage.setRequestId(requestId);
    // 异步发送消息
    log.info("push message producer send message {} {}", userId,
        GSON.toJson(pushMessage));
rocketMQTemplate.asyncSend(BusinessConstants.TOPIC_PUSH_MESSA
GE,
        pushMessage, new SendCallback() {
          @Override
          public void onSuccess(SendResult sendResult) {
            log.info("push message async message succeeded
sendResult:{}",
                sendResult);
          }

          @Override
          public void onException(Throwable throwable) {
            log.error("push message async message failed: {}",
                throwable.getMessage());
          }
        });
  }
}

```

> 消费者

```
@Slf4j
@Component
@RocketMQMessageListener(topic =
BusinessConstants.TOPIC_PUSH_MESSAGE,
    consumerGroup = BusinessConstants.GROUP_CONSUMER_TOOLS)
public class PushToUserConsumerService implements
    RocketMQListener<PushToUserDto>,
RocketMQPushConsumerLifecycleListener {

  @Override
  public void onMessage(PushToUserDto pushToUserDto) {
    String requestId = pushToUserDto.getRequestId();
    if (!Strings.isNullOrEmpty(requestId)) {
      ThreadContext.put("requestId", requestId);
    }
    log.info("push message consumer receipt message {} {}",
        pushToUserDto.getUserId(),
        new Gson().toJson(pushToUserDto));
    try {
      //xxx 业务处理
    } catch (Exception e) {
      log.error("push message consumer receipt message fail {} {}",
          pushToUserDto.getUserId(), e.getMessage(), e);
      throw new RuntimeException(e);
    }
  }

  @Override
  public void prepareStart(DefaultMQPushConsumer consumer) {
    // 设置消费者重试次数
    consumer.setMaxReconsumeTimes(3);
    // 每次提取的最大消息数
    consumer.setPullBatchSize(32);
  }

```

2.5 Webhook处理等业务场景

* webhook 回调的场景我们这边特别多，比如appstore购买订阅回调，google/paypal/stripe 购买订阅的回调等等，由于我们这边有多个数据中心，但是上游平台只能配置一个地址，我们也不知道这次的回调数据要回调到哪个数据中心，只能回调到其中一个，再进行转发到其他数据中心，现在转发的策略是通过new Thread() 进行异步转发，不考虑消息是否成功/失败，监听不到消息是否请求，对我们上层来说无感/危险性极大，这不改造机会来了，下面都是改造过程中的一些代码(已脱敏)。

> 生产者

```
@Slf4j
@Component
public class CallBackProducerService {

  private static final Gson GSON = new Gson();

  @Autowired
  private RocketMQTemplate rocketMQTemplate;


  /**
   * 异步发送 callback消息.
   *
   * @param appStoreCallBackDto
   */
  public void asyncSendAppStoreCallBackMessage(
      AppStoreCallBackDto appStoreCallBackDto) {
    log.info("appstore callback producer send message {}",
        GSON.toJson(appStoreCallBackDto));
rocketMQTemplate.asyncSend(BusinessConstants.TOPIC_APPSTORE_CALLBACK,
        appStoreCallBackDto, new SendCallback() {
          @Override
          public void onSuccess com/
"https://blog.csdn.net/x763795151/article/details/122828134")
* [基于 RocketMQ Prometheus Exporter 打造定制化 DevOps 平台-阿里云开发者社区](http://cxyroad.com/
"https://developer.aliyun.com/article/783490")
* [hub.docker.com/r/apache/ro...](http://cxyroad.com/
"https://hub.docker.com/r/apache/rocketmq-exporter")
* juejin.cn/post/726253...
* [rocketmq客户端发送消息报错和超时问题-阿里云开发者社区](http://cxyroad.com/ "https://developer.aliyun.com/article/1161535")
* [RocketMQ消息消费过后会被清理吗? \_rocketmq消息消费后还在吗-CSDN博客](http://cxyroad.com/
"https://blog.csdn.net/qq_15687611/article/details/129029599")
* [RocketMQ——消息文件过期原理 - 薛定谔的风口猪](http://cxyroad.com/
"https://jaskey.github.io/blog/2017/02/16/rocketmq-clean-commitlog/")
* juejin.cn/post/722979...
```

* [微服务架构中是否应该把诸如redis、mq之类的中间件也封装成一个服务? - 知乎](http://cxyroad.com/”https://www.zhihu.com/question/306592456/answer/734320458”)[SpringBoot整合RocketMQ，老鸟们都是这么玩的！ - JAVA日知录 - 博客园](http://cxyroad.com/”https://www.cnblogs.com/jianzh5/p/17301690.html”)
* RocketMQ与SpringBoot整合进行生产级二次封装 - 掘金
* [github.com/maihaoche/r...](http://cxyroad.com/”https://github.com/maihaoche/rocketmq-spring-boot-starter”)
* [RocketMq生产者组和消费者组_rocketmq的生产者组和消费者组-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/qq_36559868/article/details/106305164”)
* [www.cnblogs.com/CF1314/p/17...](http://cxyroad.com/”https://www.cnblogs.com/CF1314/p/17681814.html”)
* [springboot整合rocketmq，支持多连接生产者和消费者配置。不同topic适配不同业务处理类_rocketmq 多连接-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/ke7025/article/details/119982155”)
* [RocketMQ同一Topic、消费组创建多个消费者失败问题_rocketmq一个topic多个group-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/qq_43128724/article/details/117324847”)
* [RocketMQ 消费者_rocketmq消费者代码-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/JemeryShen/article/details/126855342”)
* [rocketmq-spring的consumer设置消费失败最大重试次数_rocketmq springboot 设置消费失败默认重试次数-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/x763795151/article/details/118023942”)
* [在Apache RocketMQ中这种情况队列数量应该设置多少比较合理呢？_问答-阿里云开发者社区](http://cxyroad.com/”https://developer.aliyun.com/ask/575677”)
* [RocketMQ入坑指南（五）：SpringBoot集成RocketMQ和具体使用方式_rocket mq springboot实际应用-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/u014374743/article/details/136255200”)
* [RocketMQ 添加监控和系统告警通知-腾讯云开发者社区-腾讯云](http://cxyroad.com/”https://cloud.tencent.com/developer/article/1432219”)
* [使用Prometheus监控RocketMQ-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/sawyerlan/article/details/118152802”)
* [RocketMQ的监控和管理工具有哪些_rocketmq管理工具-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/weixin_41860630/article/details/134984499”)[rocketMQ中，消费者、消费者组、Topic、队列的关系_rocketmq topic和queue关系-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/yuanchangliang/article/details/119190281”)
* [面试官：RocketMQ同一个消费组内的消费者订阅了不同tag，会有问题吗？_rocketmq 订阅多个tag-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/zjj2006/article/details/122014993”)
* [RocketMQ订阅关系一致_rocketmq 消费组订阅同一topic-CSDN博客](http://cxyroad.com/

”https://blog.csdn.net/meser88/article/details/122344534?utm_medium=distribute.pc_relevant.none-task-blog-2~default~baidujs_baidulandingword~default-0-122344534-blog-122014993.235%5Ev43%5Epc_blog_bottom_relevance_base6&spm=1001.2101.3001.4242.1&utm_relevant_index=3”)
* [订阅关系一致 | RocketMQ](http://cxyroad.com/”https://rocketmq.apache.org/zh/docs/4.x/bestPractice/07subscribe/”)
* [RocketMQ Promethus Exporter | RocketMQ](http://cxyroad.com/”https://rocketmq.apache.org/zh/docs/4.x/deployment/04Exporter”)
* juejin.cn/post/684490…
* juejin.cn/post/684490…
* [cluster 集群参数](http://cxyroad.com/”https://learn.lianglianglee.com/%E4%B8%93%E6%A0%8F/RocketMQ%20%E5%AE%9E%E6%88%98%E4%B8%8E%E8%BF%9B%E9%98%B6%EF%BC%88%E5%AE%8C%EF%BC%89/15%20RocketMQ%20%E5%B8%B8%E7%94%A8%E5%91%BD%E4%BB%A4%E5%AE%9E%E6%88%98.md”)
* [Rocketmq消息批量发送&消息批量消费-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/l123lgx/article/details/130841226”)
* [Consumer消息拉取和消费流程分析_consumeconcurrentlymaxspan-CSDN博客](http://cxyroad.com/”https://blog.csdn.net/qq_32099833/article/details/120229506”)
原文链接: https://juejin.cn/post/7372020457125052450