# 红包业务设计文档

支付平台

2015年11月

**修订历史**

|  |  |  |
| --- | --- | --- |
| **时间** | **修订人** | **描述** |
| 2015-11-5 | 赵一飞 | 设计红包业务。 |
| 2015-12-3 | 赵一飞 | 抢红包使用 redis，对设计方案调整。 |
| 2015-12-8 | 赵一飞 | 发红包交互改为先生成红包订单，支付后支付系统回调通知红包系统。 |
| 2015-12-19 | 赵一飞 | 补充系统架构图及明细文档。抢红包部分目前逻辑和之前文档有差异，更新同步一下。 |

**目录**

[红包业务设计文档 1](#_Toc438317008)

[1 背景 4](#_Toc438317009)

[2 名词解释 4](#_Toc438317010)

[3 红包业务概要设计 4](#_Toc438317011)

[3.1. 系统概述 4](#_Toc438317012)

[3.2. 系统架构 6](#_Toc438317013)

[3.3. 系统功能 6](#_Toc438317014)

[3.3.1. 流程图 6](#_Toc438317015)

[3.3.2. 时序图 16](#_Toc438317016)

[4 数据库设计 20](#_Toc438317017)

[5 接口设计 24](#_Toc438317018)

[5.1.对外接口 24](#_Toc438317019)

[5.2.调用外部接口 24](#_Toc438317020)

[5.2.1 支付系统回调通知红包系统； 24](#_Toc438317021)

[5.2.2转移资金； 24](#_Toc438317022)

[5.2.3解冻资金； 24](#_Toc438317023)

[6 数据存储 24](#_Toc438317024)

# 背景

红包是钱包的一个场景，针对企业场景下的使用频率和方式没有成熟的先例，目前是个尝试。因此，以下的设计方案主要参考了微信的个人红包业务。

# 名词解释

拼手气红包：红包金额按照总金额/总数目 随机生成。

普通红包：红包总金额是 红包个数\*单个金额。

群红包：由于企信群的最大人数为150人，因此150人以下的红包会采取群红包的方式。

部门红包：150人以上的红包（如管理者发的部门红包），采用服务号方式，通过服务号向大家发送红包消息（模板消息），点击进入领红包页。

手气最佳是全部人抢到之后，金额最高的一个。（如果有多个金额相同的，选择抢到时间最早的一个）

# 红包业务概要设计

## 3.1. 系统概述

抢红包类似于淘宝秒杀，压力主要是并发高，海量支付。

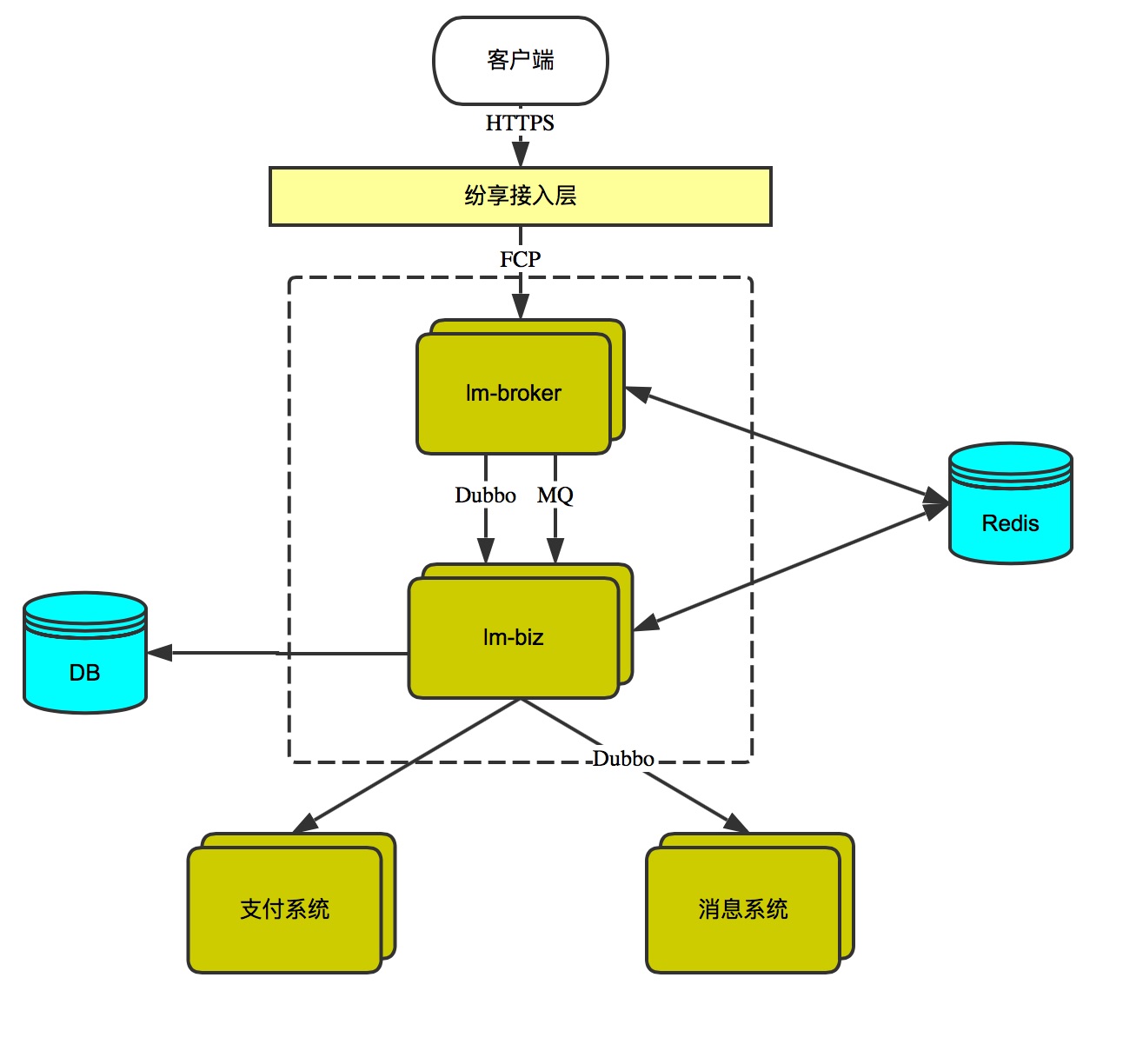
因此设计原则是追求高可用和快速响应。选择牺牲一部分数据一致性和完整性从而保证核心功能绝大多数运行。

红包业务可以拆分为以下场景：

1. 发红包 –支付
2. 抢红包 – 抢资格
3. 拆红包 – 资金入账

分拆红包步骤，确保最短关键路径即可。

## 3.2. 系统架构



## 3.3. 系统功能

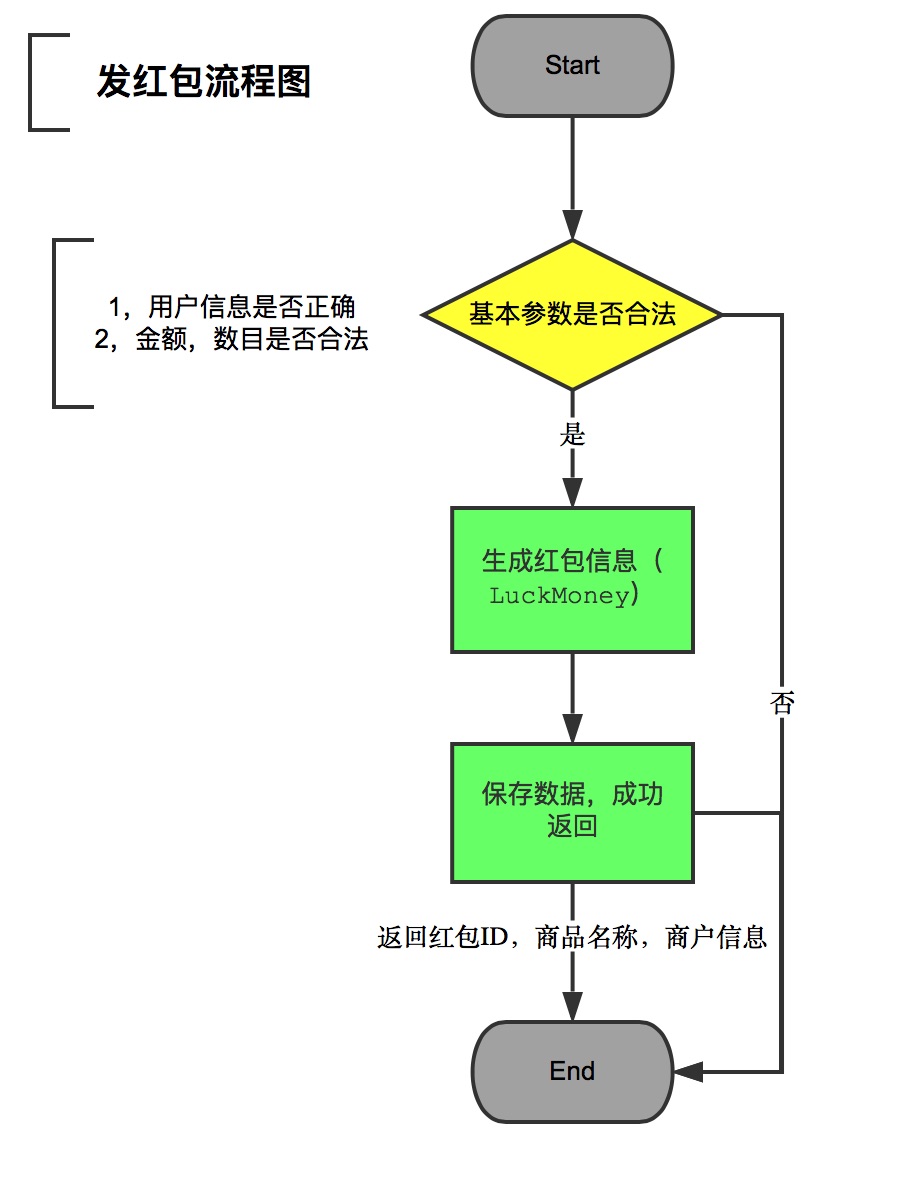
### 3.3.1. 流程图

##### 发红包

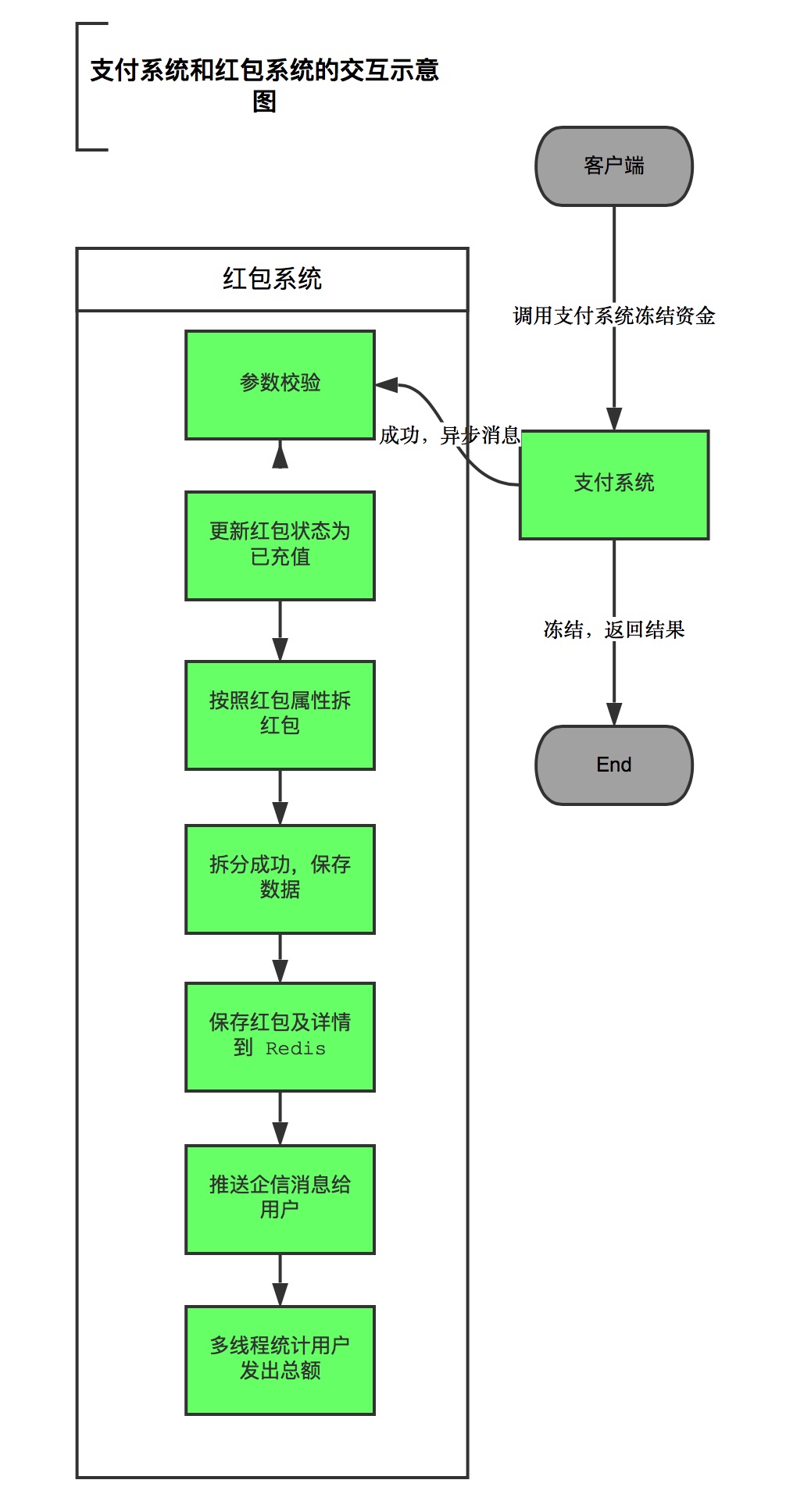
发红包操作从页面上和交互上分为以下几个步骤（以下以群红包为例）：

1. 用户在群对话框选择发红包，输入红包必要的信息：红包类型，个数，金额（等额红包的时候金额是计算出来的），吉利话。
2. 点击准备发红包按钮，调用红包系统生成红包订单（如下图）；
3. 调用支付系统获得用户余额及可用银行卡信息；
4. 进入标准支付流程，选择付款类型（余额还是银行卡），然后输入支付密码；
5. 支付完成；

前置条件：用户填完红包的必要信息，红包类型，个数，金额，进入发红包流程（Start）。



支付系统和红包系统的交互图如下：



支付系统会回调通知红包系统，接到回调后，执行以上操作。

支付系统会调用8次，时间间隔为 0， 2m，10m，30m，60m，2h，6h，15h。

**红包随机算法：**

基本原则：单个红包最小金额是 1分（保证每个红包都有钱拿），最大金额是 (当前红包剩余总金额/当前未计算金额的红包数目) \* 2。详见代码。

**方法定义：**

static List<Long> getRandomList(Long totalAmount, Integer number)

**入口参数意义：**

total：红包总金额（单位为分）

num：剩余红包数

**返回值：**本次随机金额 List

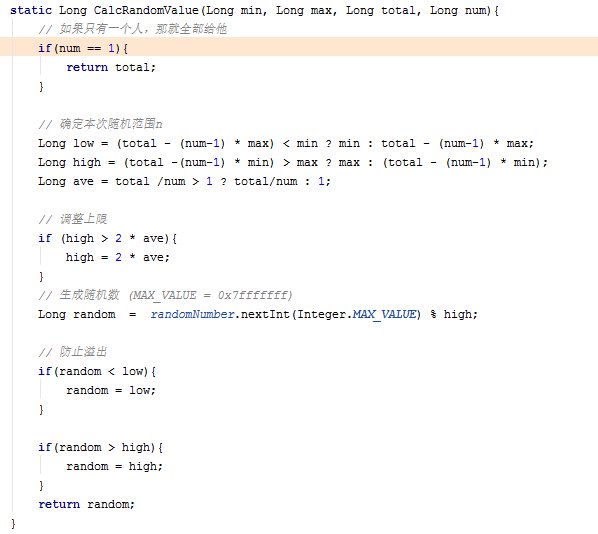
**内部机制：**

生成随机数（*randomNumber*.nextInt(Integer.***MAX\_VALUE***），然后

**随机数 \*（（剩余金额/剩余红包数目）\*2）** ，

乘以2是为了扩大值范围，判断是否在目前允许的最小，最大范围内。如果没有超过，就使用这个值。如果低于最小值，使用最小值，超过最大值，使用最大值。

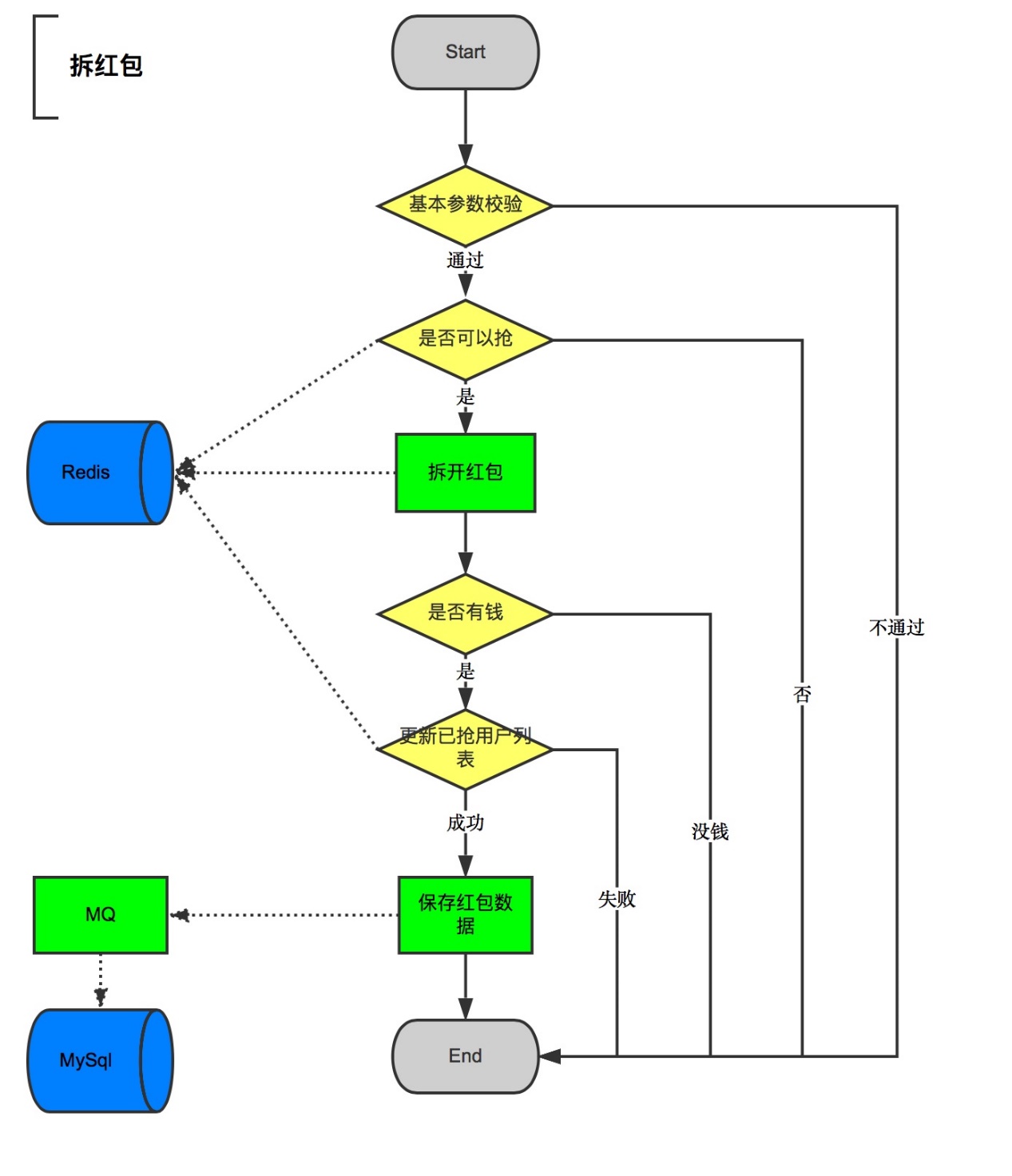
详细代码如下：



**数据持久化机制：**

先写DB，然后写redis。保证数据一致性。发红包可以慢一点。毕竟发的人远远少于抢的人。

**3.3.1.2 抢/拆红包（先抢后拆）**



**数据持久机制：**

先写Redis，同时发消息到MQ系统，然后另一个线程负责读取MQ的消息，将数据同步到Mysql DB。

1.如果Redis挂掉，抢红包挂掉。重启Redis之后，需要从DB重新加载数据。

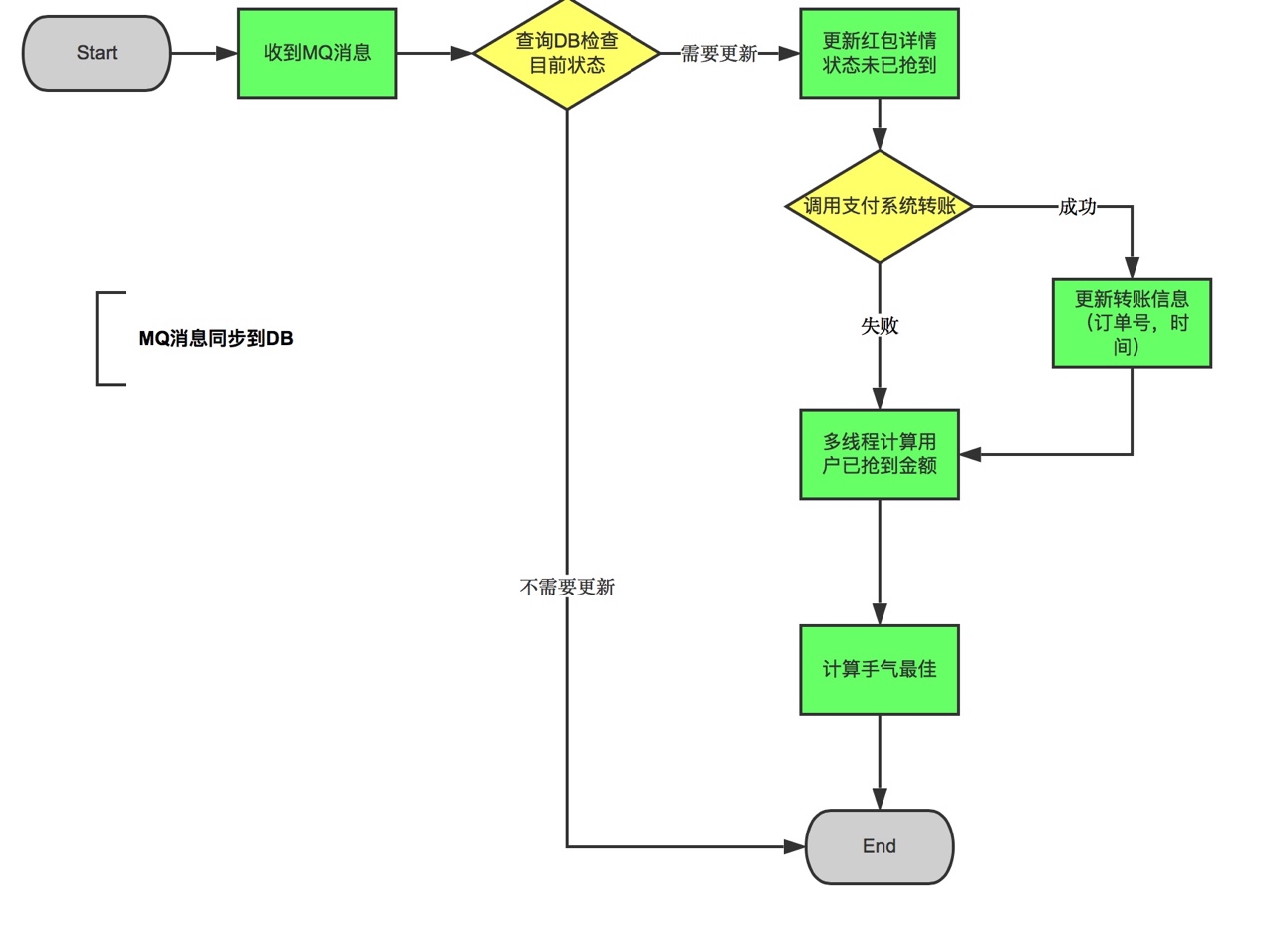
2. 如果MQ挂掉，写入MQ报警，紧急处理。

1. 如果DB挂掉，可以等DB恢复后重新处理消息。这时候其实不会调用支付系统转账。

PS: Redis，MQ，DB的持久化选项都打开。先保证他们崩溃了可以自恢复，如果不行，在依赖其他系统恢复数据。

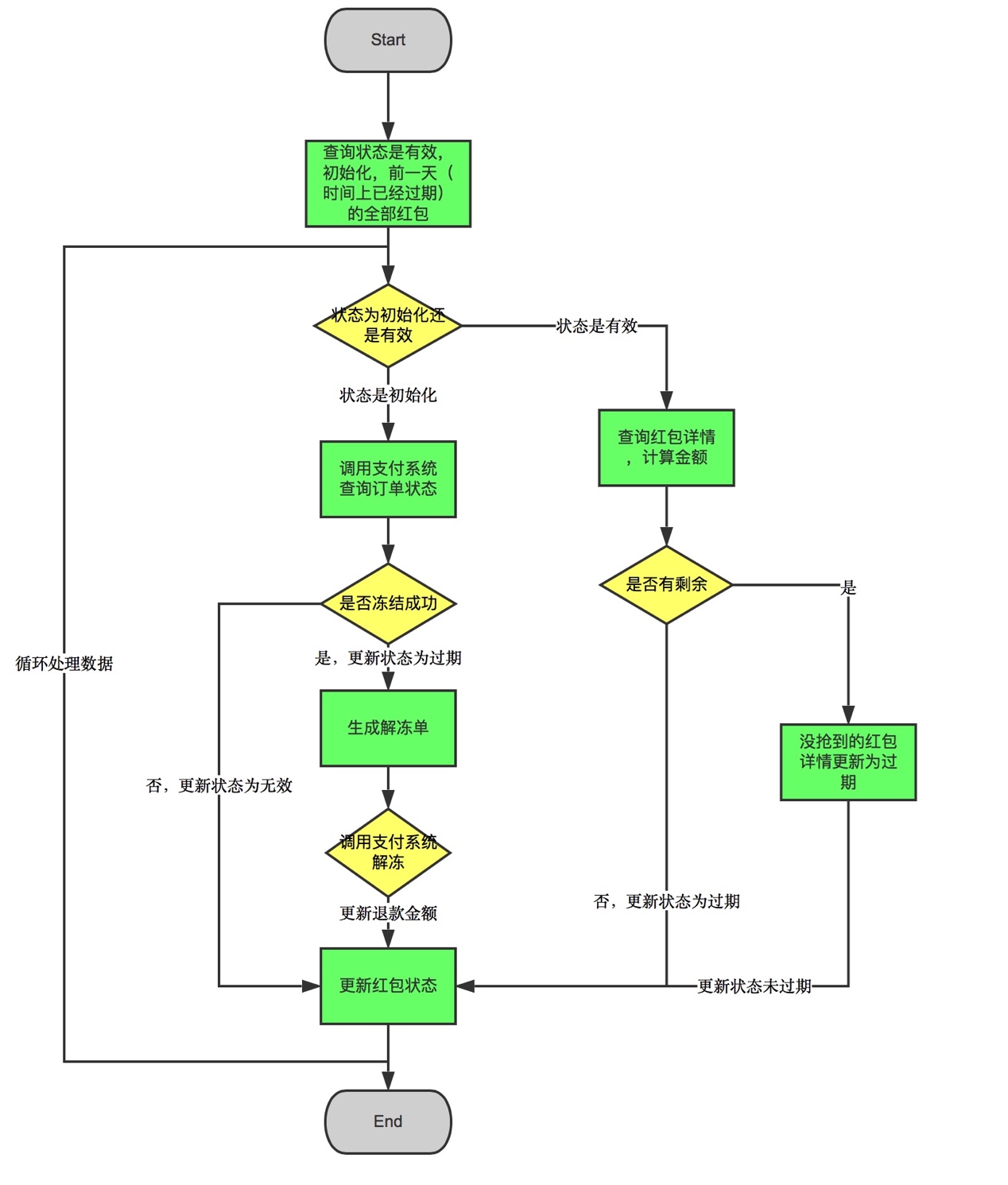
由于抢红包使用了Redis，而Redis是单线程的，天生实现了队列。

**MQ持久化到DB：**

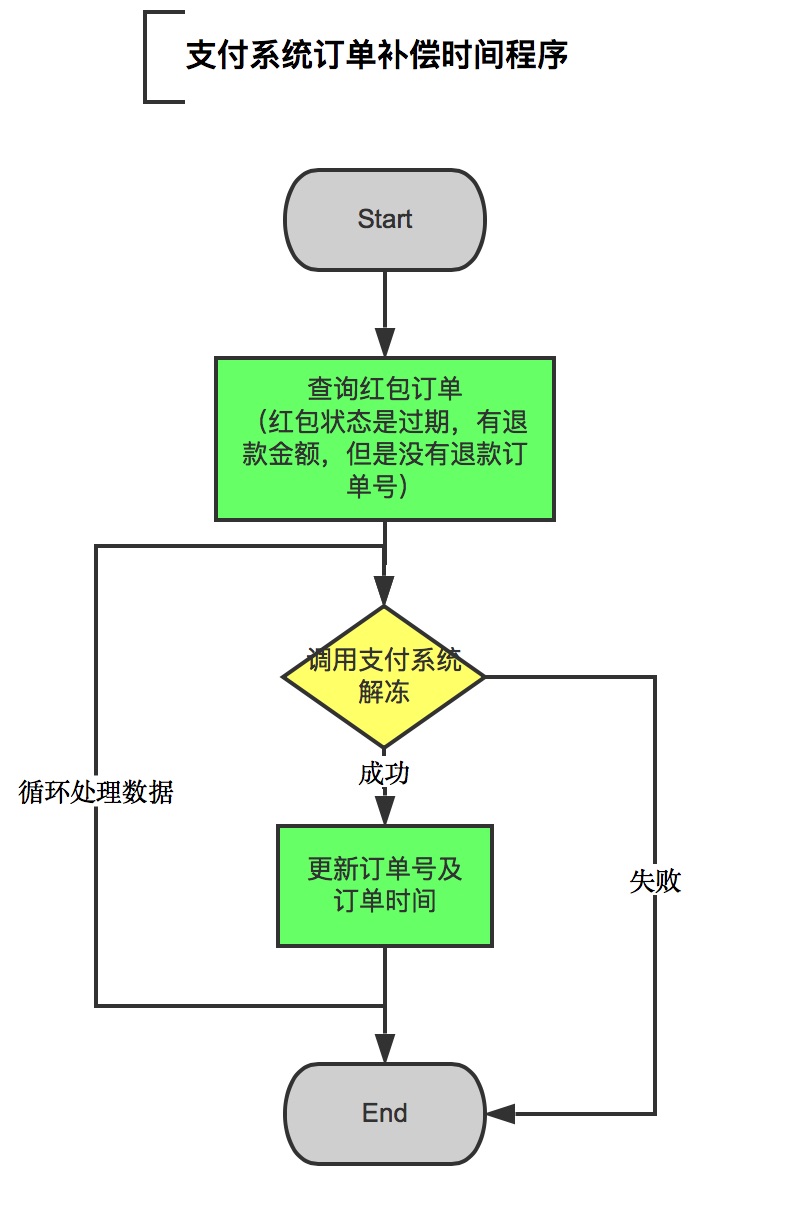
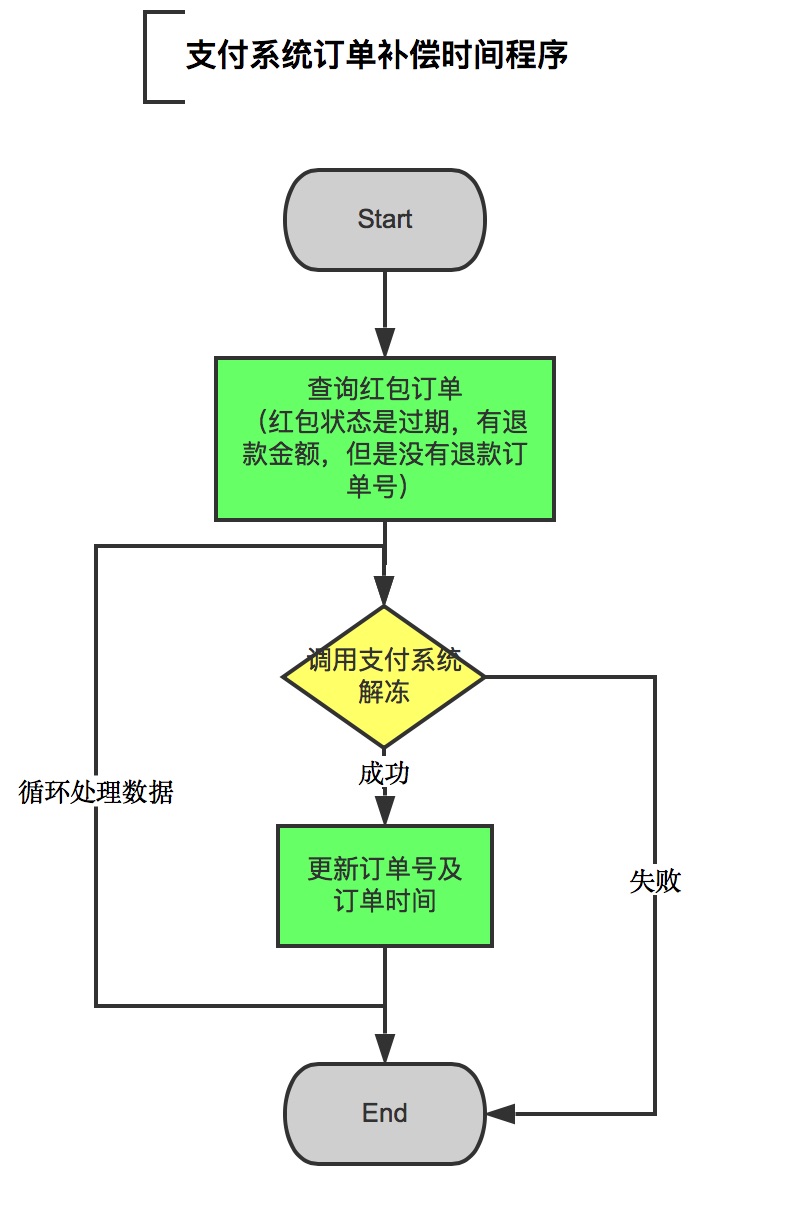


如果调用支付系统转账失败，可以使用时间程序继续补偿处理该记录。

**3.3.1.3 未领取红包金额过期返还**



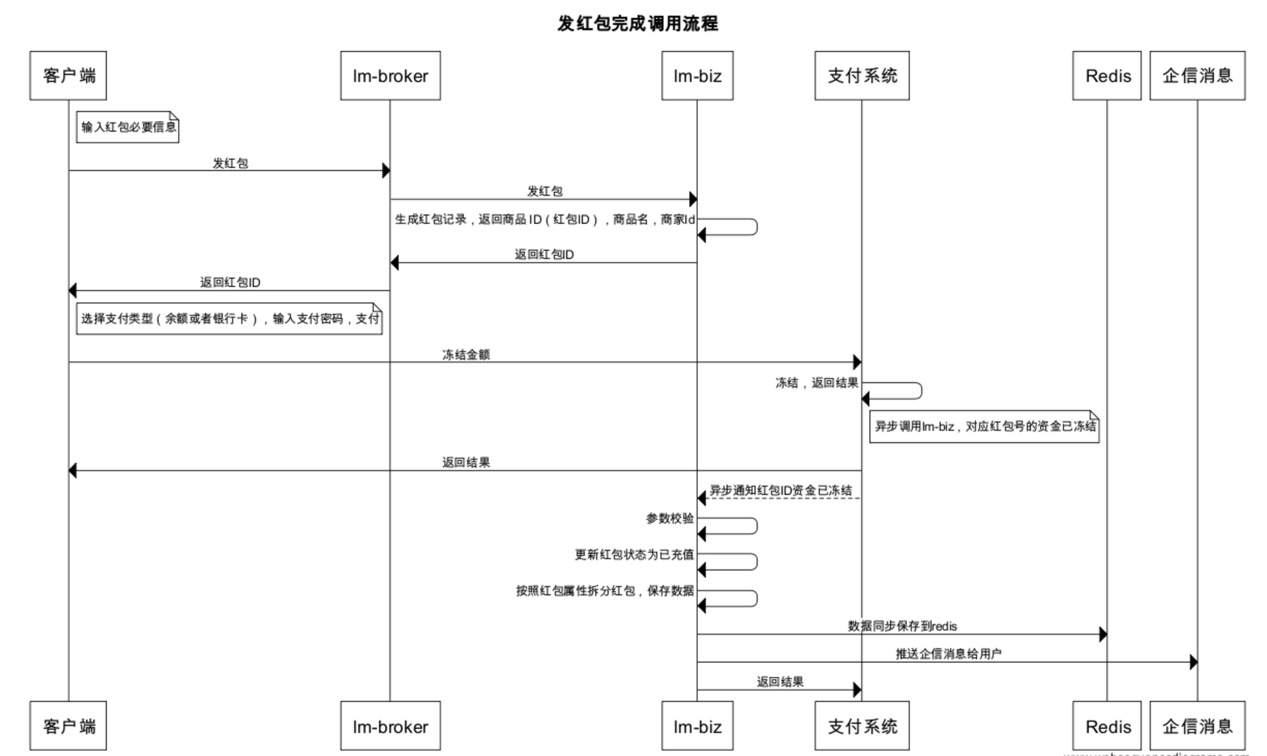
**3.3.1.4 支付系统订单补偿**



以上是红包订单，红包详情订单处理也是类似的。

### 3.3.2. 时序图

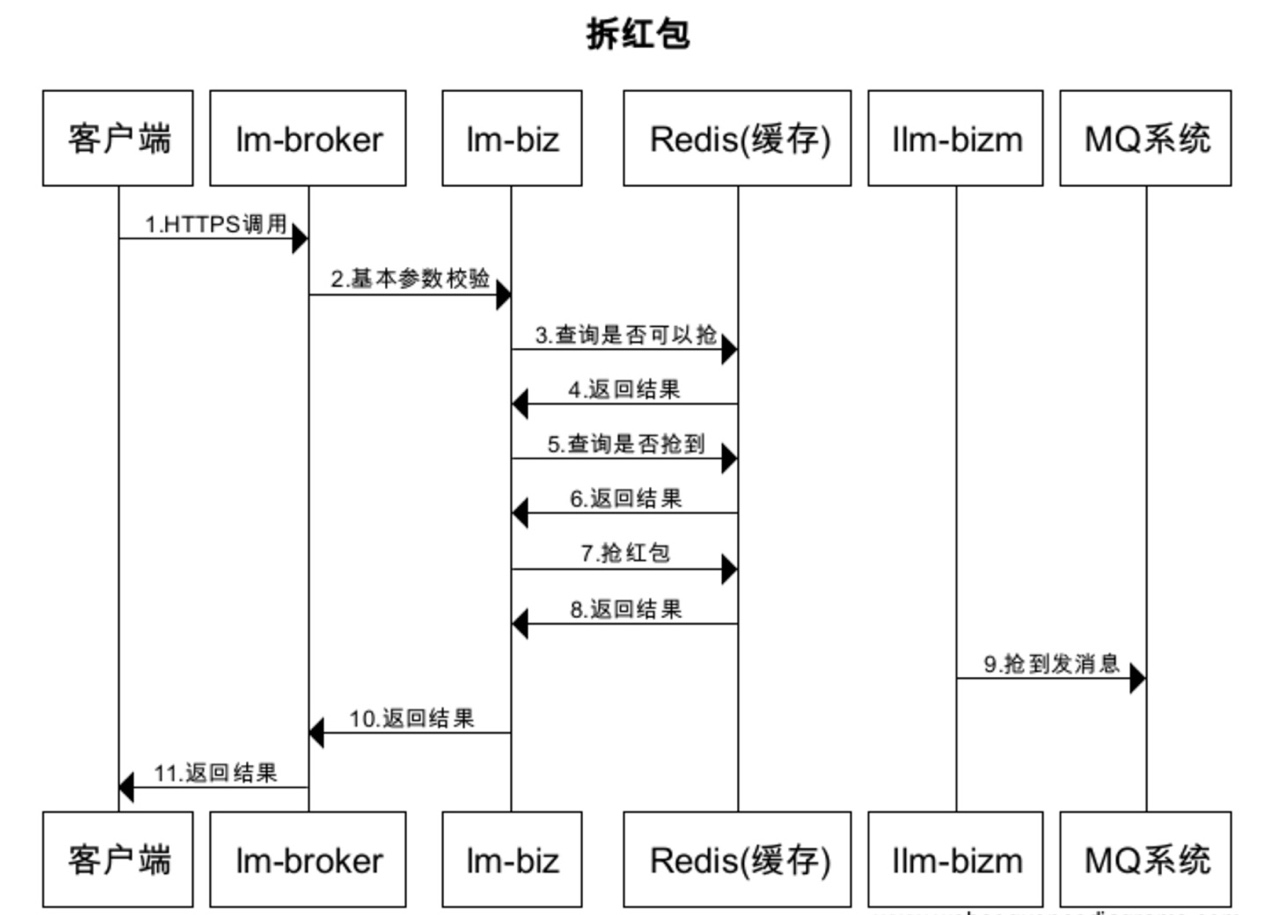
##### 3.3.2.1 发红包



描述：

1. 客户端填写红包的一些固定信息，类型（拼手气，普通），范围（个人，群，部门），金额，个数，调用红包系统生成红包；
2. 客户端收到红包 ID，调用通用的支付组件（余额，银行卡信息，支付密码输入框），用户完成支付流程，调用支付系统进行支付；
3. 金额足够（银行卡金额足够或者余额足够），支付完成；
4. 返回用户成功；
5. 支付系统回调通知红包系统，以红包ID 做标识；
6. 红包系统更新红包信息，计算红包详情，入库；
7. 保存红包信息到 Redis 缓存，推送模板消息到企信；
8. 企信下发消息，用户可以抢红包了；

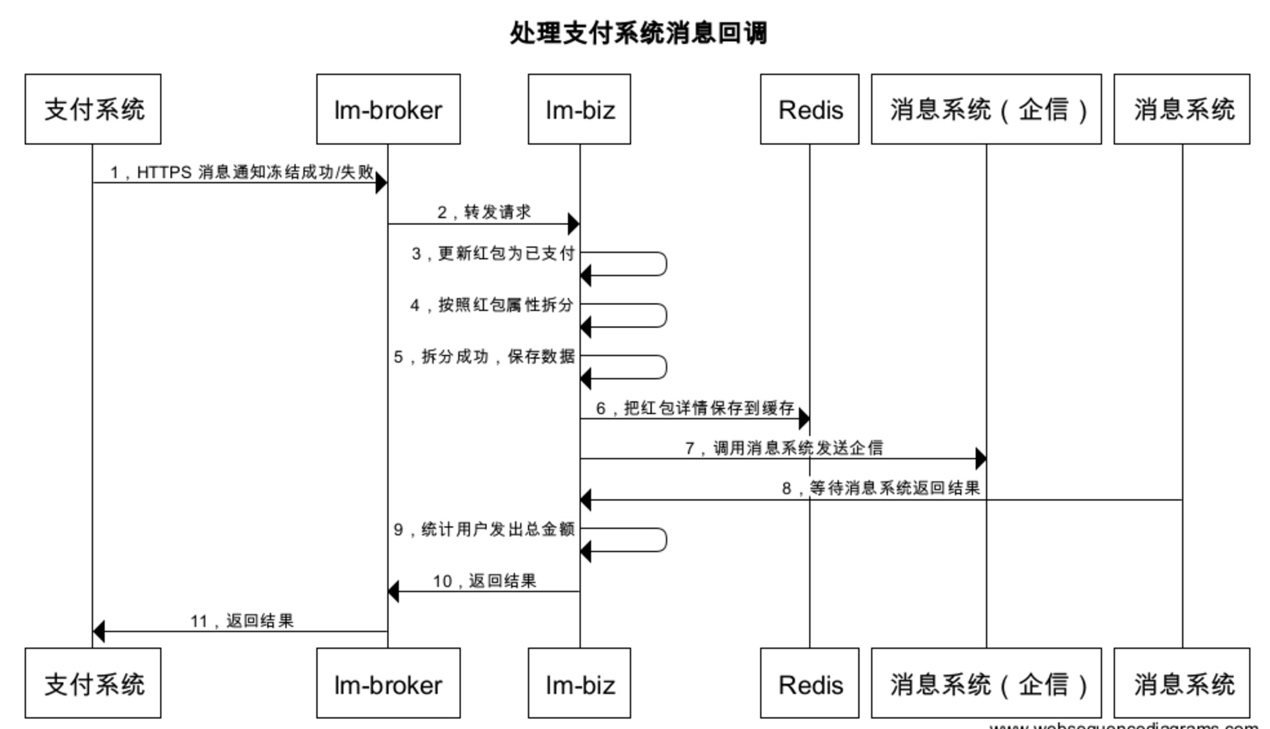
##### 3.3.2.2 拆红包



描述：

1. 客户端 HTTPS 调用红包系统；
2. 进行基本参数校验；
3. 查询是否可以抢，是否抢到了，然后开始抢（拆）红包；
4. 返回结果；
5. 抢到就要更新 Redis，同时发送消息到 MQ 系统（做 DB 同步）；
6. 返回结果；

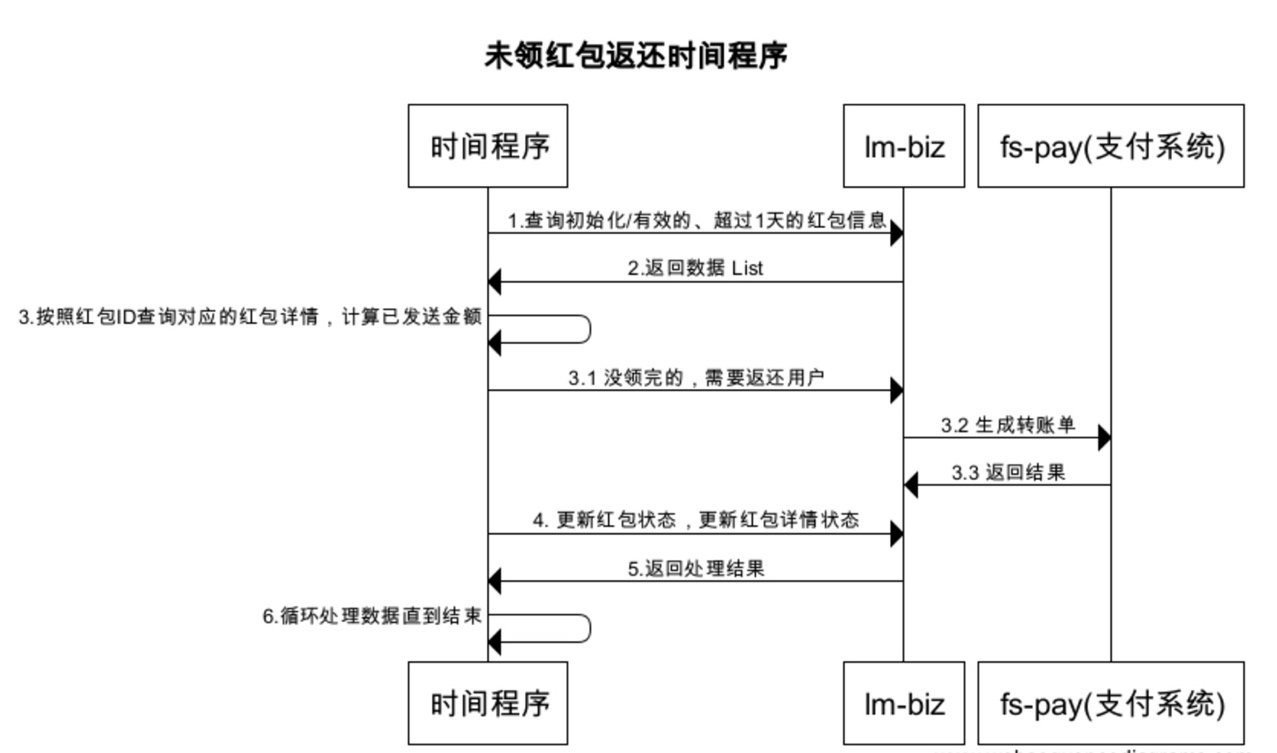
**3.3.2.3 处理支付系统回调消息**

****

**描述：**

1. 支付系统HTTPS通知红包系统有对应红包 ID 的已经被支付了；
2. 按照该红包 ID 查询目前红包的情况，如果状态是初始化的，更新为已支付；
3. 拆分红包，入库保存；
4. 发送企信消息，通知用户（确保调用成功，如果没有，需要重试）；
5. 累计用户发出总金额；
6. 返回支付系统成功（不用重发）；

**3.3.2.5 未领取红包返还**



描述：  
1. 时间程序查询有效的、1天以上的红包订单或者初始化的红包订单信息；

2. 按照红包订单的主键ID查询关联的红包详情，计算领取的金额。如果全部领完，更新状态为过期；

3.1 没领完，需要返还给用户；

3.2 调用fs-pay-order生成转移单返还剩余金额给用户；

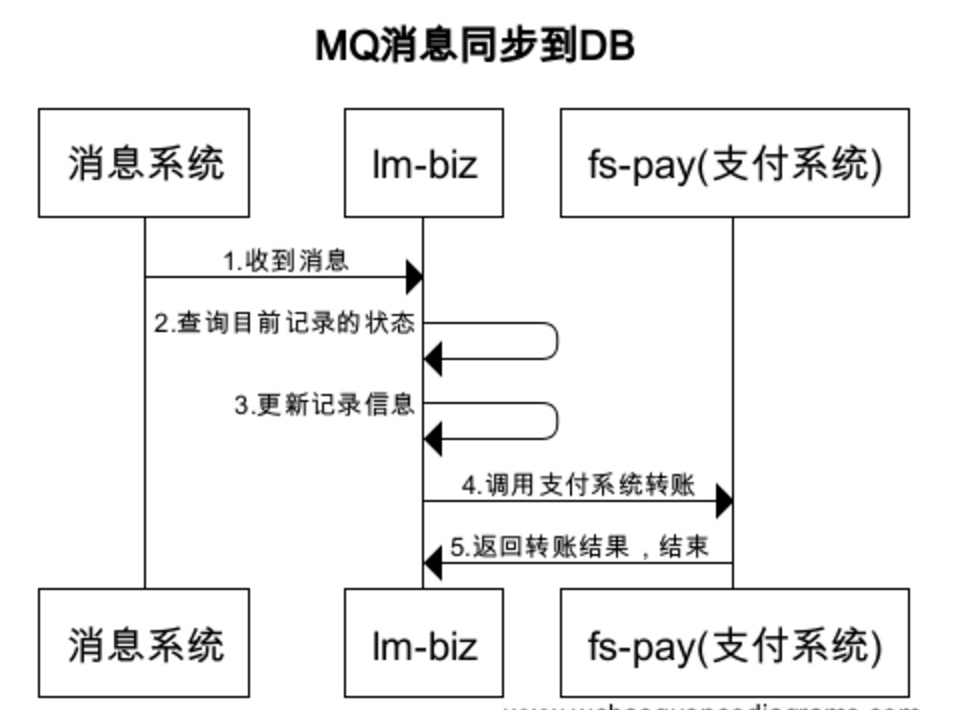
3.3 按照fs-pay-order 的返回值更新对应的转移ID到红包信息中；

4. 按照上面的情况更新红包状态（过期），红包详情状态（过期），如果有返还转移单，记录单号；

5.返回处理结果给时间程序；

6.循环处理搜索到的数据。

**3.3.2.6 MQ消息同步到DB**



描述：

1. lm-biz 收到消息；

2. 按照记录信息查询目前的记录状态；

3. 更新记录信息；

4. 异步调用支付系统转账；

5. 返回转账结果，结束；

# 数据库设计

1.红包信息表（luck\_money）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 描述 |
| id | int | 11 | 自增ID |
| fs\_user\_name | string | 50 | 纷享用户名 |
| Ea | String | 50 | 企业 |
| fs\_user\_id | int | 11 | 纷享用户Id（FCP中的 employeeId） |
| type | Int | 11 | 红包类型（1：随机红包；2：等额红包） |
| Scope | Int | 11 | 红包范围（1，单人；2，群；3，部门） |
| amount | Int | 11 | 红包总金额 |
| number | Int | 11 | 红包数量 |
| Remark | String | 100 | 红包吉利话（备注） |
| status | Int | 11 | 状态（初始化，有效，过期） |
| expire\_time | Datetime |  | 过期时间 |
| Order\_id | Int | 11 | 关联的转账ID |
| Order\_date | Datetime |  | 转账时间 |
| Refund\_id | Int | 11 | 关联的返还订单ID |
| Refund\_date | Datetime |  | 返还单调用时间 |
| Refund\_amount | Int | 11 | 返还的金额 |
| create\_time | Datetime |  | 创建时间 |
| update\_time | Datetime |  | 修改时间 |

2.红包详情表(luck\_money\_detail)

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 描述 |
| Id | int | 11 | 自增ID |
| luck\_money\_id | Int | 11 | 红包信息ID |
| ea | String | 50 | 企业 |
| fs\_user\_id | int | 11 | 纷享用户ID（FCP中的 employeeId） |
| fs\_user\_name | String | 50 | 纷享用户名 |
| total\_amount | BigDecimal | 18 | 红包总金额 |
| number | Int | 11 | 红包总数目 |
| type | Int | 11 | 红包类型 |
| Scope | Int | 11 | 红包范围 |
| expireDate | Datetime |  | 过期时间 |
|  |  |  |  |
| Receive\_ea | String | 50 | 企业 |
| fs\_receive\_id | Int | 11 | 抢到红包的用户ID（FCP中的 employeeId） |
| fs\_receive\_name | String | 50 | 抢到红包的用户名 |
| amount | Int | 11 | 红包金额 |
| Memo | String | 100 | 备注 |
| grab\_date | Datetime |  | 抢到红包的时间 |
| order\_id | Int | 11 | 关联的转账ID |
| Order\_date | Datetime |  | 转账时间 |
| Status | Int | 11 | 红包情况（1：未领取；2，已领取；3，过期；4，已退还） |
| create\_time | Long | 18 | 创建时间 |
| update\_time | Long | 18 | 修改时间 |

3 红包汇总表（luck\_money\_summary）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 描述 |
| Id | Int | 11 | 自增 ID |
| ea | String | 50 | 企业 |
| Fs\_user\_id | Int | 11 | 用户 ID（FCP中的 employeeId） |
| Fs\_user\_name | String | 50 | 用户名 |
| Year | String | 11 | 年份 |
| Type | Int | 11 | 类型（收到、发出） |
| Total\_amount | Int | 18 | 金额 |
| Best\_luck\_number | Int | 11 | 手气最佳次数 |
| create\_time | Datetime |  | 创建时间 |
| update\_time | Datetime |  | 修改时间 |

# 5 接口设计

## 5.1.对外接口

对外接口转到 @乔世魁 的git文档中。

http://git.firstshare.cn/LuckMoney/fs-luckmoney/blob/ce2fcaf8a7949d66bad4ad340d4d10aae38ff569/doc/%E7%BA%A2%E5%8C%85%E4%B8%9A%E5%8A%A1%E6%8E%A5%E5%8F%A3%E5%AE%9A%E4%B9%89.md

## 5.2.调用外部接口

### 5.2.1 支付系统回调通知红包系统；

### 5.2.2转移资金；

### 5.2.3解冻资金；

# 6 数据存储

1. 描述

红包系统存储数据主要采用MySql和Redis。Redis缓存红包数据，抢红包过程中，所有数据均通过Redis读写。当用户成功领取到红包以后，broker系统通过RabbitMQ异步通知biz模块更新MySql数据库。

2. 系统部署

MySql：采用MHA方式部署，保证系统高可用

Redis： 采用sentinel 方式部署(1主3备)。保证系统高可用。

RabbitMQ： 采用mirror方式部署（3台群集），保证系统高可用

3. 数据一致性

3.1. 发送红包

a. 如果MySql写失败，则红包发送失败

b. 如果redis 写失败，则红包发送失败。系统24小时后退款

c. 如果企信发送失败，系统24小时后退款

3.2 抢红包

a. 如果redis更新失败，则抢红包失败，不影响用户金额

b. 关于MQ失败，指定重试策略，加人工干预方式保证MQ写操作最终成功

c. MySql 更新失败。系统失败，但是不影响最终用户金额。

# 7 对外系统数据补偿

## 7.1 支付系统

### 5.2.1 冻结

冻结成功的时候，支付系统有回调消息。如果回调通知消息都没有收到，最后红包订单无效之前也会调用一次再更新为无效。

### 5.2.2 转移

用户抢红包成功，需要转移资金到用户帐户。有金额，但是没有转移订单号。就一直调用知道成功。

### 5.2.3 解冻

红包没抢完，部分金额需要解冻返还给用户。

PS:以上主要依赖时间程序轮询完成操作。

## 7.2.消息系统（企信）

这部分还需要补充。TODO