# 银行转账项目总结

## 前言

整个银行项目分为前台和后台两个子系统，银行转账项目属于前台系统中的一部分。项目主要功能有行内转账、跨行转账、跨境转账、资金归集和主动收款这几个功能模块。

行内转账在同一银行的同一个或者不同服务器中，可以直接调用执行方法或者调用行内提供的接口。跨行和跨境转账是属于不相关的服务器上，所以不可能直接调用其他银行的接口，而使用高级消息队列联系起来。资金归集是银行提供的一项可选业务，副卡只能绑定一张主卡，主卡可以被多个副卡绑定。通过设定归集金额，在每个月的固定时间内，自动地把副卡中超过归集金额部分的金额转到主卡账户中，归集不收取手续费。主动收款是为一些借贷关系的账户之间，提供催款提醒功能。

## 项目介绍

随着互联网技术日新月异的发展，日常的生活服务越来越离不开网络。支付宝、微信支付等网上交易平台如雨后春笋般出现，这些网上交易系统在为客户提供便利的同时，也创造了巨大的经济效益，促进了互联网经济的繁荣。在这巨大的市场的推动下，各大银行也开始开发自己的网上银行系统。本项目正是网上银行的转账业务的实现。

银行转账项目主要分为三个主要的功能模块：同行转账、跨行转账和跨境转账。在这三种主要业务之下，还包括了资金归集、主动收款等业务。

从程序实现的角度考虑，转账业务功能的实现，实际上是数据库中一个账户（账户甲）扣钱、一个账户（账户乙）加钱的过程，需要考虑到资金流动的正确性，不能够造成资金纠纷，否则这个系统是不合格的。如果把账户甲的转账款项给扣除了，但是乙账户没有款项添加，这样不仅会造成巨大的经济损失，还会失去顾客的信任，这样显然是不符合我们的开发原则的。所以银行转账这个项目需要极大地保证数据的一致性。

对于保证数据的一致性，一般来说，我们选择利用事务来进行管理。如果是同行转账业务，只需要将“甲账户扣款（数据库数据减去转账的数额）”和“已账户加款（数据库数据加上转账的数额）”作为一个事务，要么一起成功，要么一起失败，由事务的属性管理数据库的回滚，就能够保证数据的一致性。但是对于不同的银行之间的转账，扣款和加款是在不同的数据库上进行操作的，这时候事务就变得相当地麻烦。其他银行不可能提供修改数据api接口让我们访问到它们的业务系统直接进行加款，这意味着外部可以调用它的api修改它们的数据，这显然是不安全也不合理的。对于这种情况，我们决定使用消息队列进行处理。在进行跨行跨境转账时，本地银行对数据进行扣款后，将加款的信息发送到消息队列中，在某个夜深人静，业务量较小的时候，将消息通知发送给其他银行指定的api接口中，通知其他银行对转账接收进行加款。如果加款成功，则本次转账完成。如果失败，返回一个状态码，通知本地银行转账失败，本地银行再进行资金冲正（把转出去的钱加回来）。

资金归集和主动收款业务是在转账业务的基础上进行的，归集和收款是调用转账的方法来保证资金的流动。当然资金归集需要签约，所以存在对应的签约方法和解约方法。主动收款需要调用外部消息通知客户进行还款。

## 客户端

### 同行转账

该功能模块主要是为了实现本行客户转账给本行另一客户的功能。页面提供了境内转账的操作页面，简洁直观，用户只需选择境内转账，输入想要转入的卡号和用户名，系统会自动验证卡号和用户名是否一致，确保转账的正确性；提供余额查询接口，用户只需点击余额查询即可获知余额。不仅如此，用户还可以点击常用收款人选项，自动填充卡号和用户名，简化用户操作，实现一键转账功能。转账分为立即转账和2小时后转账两种模式，满足用户多样化需求，定时转账执行前还可取消，确保了用户使用的最大自由度 。

用户的卡号和用户名验证采用了ajax技术实现异步请求，将用户名和卡号传至后台系统，查询验证转入用户的存在与否，及时反馈用户；使用事务管理确保用户资金的安全性，事务管理符合以下四个特点：

①原子性（atomicity）：一个事务必须被视为一个不可分割的最小工作单元，整个事务中的所有操作要么全部提交成功，要么全部失败回滚，对于一个事务来说，不可能只执行其中的一部分操作，这就是事务的原子性。

②一致性（consistency）：数据库总是从一个一致性的状态转换到另一个一致性的状态。在前面的例子中，一致性确保了，即使在执行第三、四条语句之间时系统崩溃，前面执行的语句也不会生效。因为事务最终没有提交，所以事务中所做的修改也不会保存到数据库中。

③隔离性（isolation）：通常来说，一个事务所做的修改在最终提交以前，对其他事务是不可见的。当执行完第三条语句、第四条语句还未开始时，此时有另外一个程序开始运行，则看不到第三条语句做出的改变。

④持久性（durability）：一旦事务提交，则其所做的修改就会永久保存到数据库中。此时即使系统崩溃，修改的数据也不会丢失。持久性是个有点模糊的概念，因为实际上持久性也分很多不同的级别。有些持久性策略能够提供非常强的安全保障，而有些则未必。而且不可能有能做到100%的持久性保证策略。

## 该模块的实现让我对事务管理有了更近一步的了解，了解了银行转账如此简单操作下，其背后的业务逻辑的复杂性。转账的安全性比效率要求更高，分布式部署实现了在高并发条件下，服务能轻松应付，减轻服务器压力。多机热备保证了数据的安全性，配合熔断器，保证了程序能24小时不间断提供服务。

### 跨行转账

该功能模块主要是为了实现本行向其他银行转账的功能，用户选择境内转账，有两种模式可供选择：立即转账和两小时后转账。为了确保转账顺利，采用消息队列这个中间件收发消息。消息队列有以下三个优点：

①解耦：系统A要发送一个消息到多个系统，如果此时每增加一个系统，系统A都需要通过修改源码来增加接口，此时耦合非常高，但是如果中间使用消息队列的话，系统只需要发送一次到消息队列，别的系统就能复用该信息，当增加或删除系统调用接口的时候，不需要额外的更新代码。

②异步：用户调用一个接口的时候，可能该接口调用了别的方法。例如：用户注册的时候，后台可能需要调用：查询数据库，插入数据库，发送邮件，发送用户指南等等，但是用户可能并不需要后台将所有的任务执行完毕，那么此时在初入数据口后面加入mq队列，用户就能很快得到注册成功的响应而去做一些别的事情。mq的机制又能保证最终的一致性，所以使用起来很安全很稳定。

③削峰：如果存在用户使用的高峰期，例如存在大量的请求访问数据库（mysql每秒2000个请求），超过就会卡死，我们使用MQ作为类似于缓冲区的作用，高峰取时在MQ中进行大量请求积压，处理器按照自己的最大处理能力取请求量，等请求期过后再把它消耗掉。

## 当本行发起跨行转账时，先在本行把钱扣了，本地事务结束，然后发送消息到RabbitMQ，记录流水账，状态标记为“处理中”，转入银行从消息队列中获取消息，并完成转入用户的加钱操作，事务完成，返回一个状态码给本行，本行根据状态码信息修改转出用户的流水账状态为“完成”，整个转账过程结束。若转账过程出现异常，比如消息丢失、网络阻塞，采取每个一段时间发送相同通知，根据流水号来区别每条消息，直至消息顺利首发。超出一定时间，则终止转账操作，把本行转出用户的转账事务回滚，把钱加回去，以此实现资金的安全性。

### 跨境转账

## 跨境转账首先需要当前用户在自己的转账页面中选中跨境，输入转入账户卡号，调用第三方接口根据卡号信息查询所属地区、银行和用户姓名，显示到转账页面。在阅读并同意收费标准后，才能选中汇款币种及金额。选择安全工具后，根据手机号接收验证码验证成功后才能完成转账。

## 服务器收到跨界转账的请求，首先验证用户信息及其操作状态，验证通过才允许其执行之后操作。转账首先根据其选中的币种和金额，调用汇率转换接口，计算所需外汇。再查询其账户余额是否充足，不足则直接返回余额不足交易失败。转账成功与失败依赖于事务，转出方账户金额的减少和转入方金额的增加需要同时完成，才能事务完成提交，否则只要其中一方失败就回滚事务。但是由于跨境转账中，转出用户和转入用户不在同一个服务器，所以这里事务处理会比较复杂。若是使用同步机制，调用接口等待境外银行回复操作是否成功，再根据状态提交或回滚。这样做会存在很大的漏洞和不足，首先同步等待会使得响应很慢，不仅用户等待时间较久，而且也占用服务器很大资源，降低了服务器的处理能力。再次仍然会存在极端情况下的操作错误，比如境外银行给转入账户加钱成功后返回时，由于网络原因使得转出方银行接收不到回应认为操作失败而回滚。

## 针对与不同服务间的事务处理，比较好的方法就是使用高级消息队列。首先使用高级消息队列是处于异步处理的，这大大减少了执行的时间和降低了服务器性能的消耗。第二，高级消息队列提供了安全可靠的传输，保证了消息不会丢失。消息队列的数学模型中分为生产方，消息中间件，消费方。在这里转出方银行为生成方，转入银行为消费方，使用rabbitmq作为消息中间件。生成方中，将转出方的账户减少和发送消息作为同一个事务执行。在使用消息队列的事务中，可能丢失数据的情况有：

## 1）生成方发送消息到队列途中丢失消息；

## 2）消费方处理完消息返回ack时，丢失消息；

## 对于第一种情况有两个解决方案：a.是消息队列采用生成方的消息确认机制，在消息发送到队列并持久化成功后才返回确认消息给生成方。生成方通过获取返回的确认消息确定消息已成功发送并保存到消息队列中，若没有收到确认则可以执行自定义的操作，回滚或是重发。b.是在发送方采取定时发送未完成的交易消息。

## 对于第二种情况，消息队列未正确地收到消费方的返回消息可能出现的问题是：该条消息一直未nack状态，直到消费方断开连接为止，可能造成消息队列阻塞；或者可能是消息重发，使一条消息被多次消费。对于可能造成消息队列阻塞的情况，一个解决方法是消费方集群，采用多个消费方轮询，极大避免了只有一个消费方阻塞了使得大量业务无法继续执行。对于一条可能多次消费的问题，可以在消息中添加一个全局唯一的标识，通过唯一标识判断该消息是否被消费过，而且需要确保消费方集群处理消息使用的是同一个数据源。

## 在消费方处理完消息返回给消息队列的时候，同样也返回一个完成的消息给另一个队列发送给生成方，通知生成方交易完成。

### 资金归集

资金归集是指将集团公司中所有下属公司指定账户上的资金归集到总公司指定的账户的服务，也称资金清扫。2013年1月初，某股份制商业银行推出一项新业务——可以给老公的银行卡设个存款上限，比如1000元，从此老公卡上的资金只要超过1000元，超过的部分就一分不留地自动划到老婆的银行卡上，不仅不需要每月转账，而且免费转主动收款。该项业务需要用户先进行签约，才能够使用。我们在用户选择资金归集功能时，首先查询当前登录的用户的资金归集签约状态，分为三种情况：未签约资金归集、已签约资金归集和作为资金归集主账号。

对于未签约资金归集的用户，我们提供资金归集签约的表单。需要用户填写签约的账号以及账号的持有人（用于验证），利用ajax验证需要签约的卡号是否正确，且为用户所填写的账号持有人持有。而且签约的卡号不能和别的账号签约过，否则页面js阻止表单提交，无法签约。签约时，需要制定该账号超过多少资金时，在特定的时间将超过的资金全部自动转账到签约的账号中。如果验证通过，表单就可以提交到后台，后台调用程序进行签约操作。签约成功后用户就可以查看自己的签约信息了。

对于已经签约资金归集的用户，可以查看自己的签约信息，也可以对信息进行修改和解约。修改时利用js技术和ajax技术进行和签约样的验证。

如果当前账号是被别人签约账号，即可查询所有和它签约的账号的信息和资金归集的记录。查看资金归集的记录时，前台发送请求，我们在后台查询到相关的所有记录。为了保证查询的效率，我们将这些记录定期查询放到redis中，需要访问时，只需要查询redis中的记录即可。同时，oracle数据库中针对不同的时间段进行了表分区处理，进一步优化查询速度。

资金归集会在固定的时间查询到所有的归集信息，判断这些归集签约信息中子卡号的资金是否满足资金归集的条件。如果满足就调用转账功能自动进行资金的转移，如果钱不够则忽略。这里利用了quartz定时任务技术，指定每月1日的上午9点自动归集资金，无需人工确认。

### 收款人管理

## 该功能模块是为了对每个不同的用户对他自己的收款人管理，可以查看自己全部的收款人，进行增删查询操作，而且还可以实现转账功能，通过勾选选择框可以选择批量转账功能，为用户提供方便。

## 首先页面先显示每个用户他自己的收款人，如果用户是新用户没有收款人，可以点击添加收款人，跳转到收款人页面，只要输入卡号和收款人姓名即可，转账金额可输可不输，输入卡号和姓名，会进行验证有没有这个用户，如果没有该用户会提示没有该用户，然后添加按钮也不会生效，如果该用户有这个收款人，会提示重复添加，如果不输入可在收款人页面添加转账金额，也可以很方便的改变转账金额，然后就是删除功能，用户要是想删除收款人，先弹出一个提示框提醒他是否确定删除，确定删除后再执行删除收款人操作，收款人页面还提供了模糊查询和分页查询，有卡号输入框和姓名输入框，如果都不输入点击查询可查询所有收款人，全部的收款人会经过分页处理，每页只展示四条数据，点击全选按钮可自动勾选当页全部收款人，点击批量转账可对这些收款人进行转账，通过调用转账接口，如果转账成功都会跳转到提示页面，提示转账成功或者失败。

## 该功能模块用了ajax技术实现验证用户名和卡号是否一致和是否重复添加，还有简单的增删改和分页查询操作，通过编写此功能让我认识到了要考虑到用户的体验性，也要考虑安全性，认识到了ajax的重要性。

### 主动收款

该模块为了便于有借贷关系或者债务关系的用户双方收款便捷，也有利于解决中国传统的外债难收的困境。

进入客户端后，点击“主动收款”，呈现出的前端界面如下图所示。



在这个模块中，用户可以点击上下页的按钮，实现查看全部收款人的借贷信息，还可以通过卡号和付款人姓名的模糊查询，点击“查询”按钮，下方列表显示搜索到的信息。在列表的右边有复选框，可以勾选任意数量的复选框，或者点击全选，选择全部用户，再点击“批量收款”按钮，就可以给对应用户批量发送催款消息，对应用户登录用户后，点击“催款通知”就可以查看。特别地，这里在后台会有一个判断，当用户重复给其他某个用户发送催款通知的话，系统不会予以处理，对应接收用户只会显示一条该催款消息，但前端会显示执行成功，避免了垃圾数据堆积，降低了数据库负担。

这里在查询输入框里面使用了正则匹配，保证了输入格式。特别地，在数据库中，在收款人卡号上添加索引，加快查询速度，这样提高用户体验。

界面设计简洁，功能一目了然，注重用户选择的自由度，在数据库层面提高查询速度。

缺点就是界面设计功能不够丰富，列表式的呈现信息稍显呆板，用户对于收款金额设置也不够灵活。

总的来说，该功能模块的设计使我加强了数据库优化、ajax异步请求，正则表达式的学习使用。

### 消息通知

该模块分为系统通知和催款通知两种，用户可以查看和直接处理催款消息，完成还款。

进入客户端后，点击“主动收款”，呈现出的前端界面如下图所示。



实现流程：

1. 进入首页后，点击“消息通知”，就会看到上述界面，有“收款和催款”、“转账”和系统消息三类消息通知
2. 点击“收款和催款”，出现消息通知列表，消息通知有已读和未读标记，点击一条催款消息，跳转到转账界面。
3. 跳转至相应转账界面，并附带卡号、姓名、金额三个值回显，这三个都不可改变，对于金额进行了ajax验证余额是否充足，当余额不足时，提示红色字“余额不足”，转账按钮失效，只有当余额充足，按钮有效，点击“转账”，执行成功，跳转到转账结果界面。

该功能是在后台调用了转账接口，对于转账金额有ajax验证余额是否充足，页面也用jQuery设置按钮由无效到有效的触发动作，使得客户端功能更加人性化和便捷。

## 服务端

### 用户管理

我们为管理员提供特殊登入账户，登入时需要数据管理员的管理号码和密码，进入主页后可对用户信息进行操作，如冻结用户，启用用户，查询用户详细信息，查询用户交易信息，查询用户登入信息，查询用户异常信息等。并且可以更改用户的部分信息，如交易限额，绑定情况等

用户管理主要用于对用户信息的管理操作，主要操作业务流程为查询修改删除等操作。所以对于数据库的优化等流程比较多。

数据库优化主要采用了表分区结构加分区索引和字段索引，对于常用查询条件如日期以每个月做范围添加范围分区，可以查询用户月账单信息等，并且添加分区索引，能够快速查询每个用户的详细时间内账单，

在区域内添加地区列表分区，大大提高查询速率，对常用的字段添加索引，如身份证添加唯一索引，银行卡号添加唯一索引，且均不可为空，必须要求用户在注册的时候填写相关内容，查询时可根据地区范围分区加银行卡号索引更加快速的查询到用户信息等，

在查询数据上，如数据量比较大的用户交易信息采用分页查询，每次按照分页条件查询部分信息，提高查询效率，减轻数据库和服务器的压力。

在sql语句上：因为oracle总是先解析SQL语句，把小写的字母转换成大写的再执行，所以我们采用所有字母都是大写，减少解析时间，提高效率。当在SQL语句中连接多个表时, 尽量使用表的别名并把别名前缀于每个列上。这样一来,就可以减少解析的时间并减少那些由列歧义引起的语法错误。并且所有语句均不适用\*号作为查询条件，而且把所需要字段填写上去，这样可以减少不需要的数据填充，和sql解析时间。在有多条件情况下，不使用or语句，采用交集并集语句，并且将筛选数据量大的，过滤最多数据的条件放在最后。

对不常修改或不能修改的数据采用二级缓存方式，如交易记录等，先将数据查询放入redis中，管理员查询时直接从redis库中读取，每在夜深人静的深夜，将redis和orcale数据库中信息进行更新处理，虽然这样使得数据库同步性变得很差，但因为是后台管理处理，所以可以减少这方面要求，但因为有时需要实时数据，所以提供手动的数据库同步操作，这样可以大大提高查询效率，提高应用可用性，减轻服务器压力。

## 四、页面

页面采用了HTML 4版本，页面基于JavaScript的编写，采用了jQuery技术jquery-2.1.4 版本和Bootstrap技术的 bootstrap-3.3.7-dist版本。

页面运行的特效和样式图片、以及项目图片资源等的静态文件，我们考虑了资源动静分离，项目中存在的大量的js、css、图片等静态资源，需要将这些静态资源单独部署带一个服务器中（例如:阿里的CDN）,然后通过远程连接地址（比如http：//192.168.2.1:8080/picture/cat.png）访问，提高页面的加载速度，同时减轻应用服务器的压力。

在页面地址上，我们规定必须使用HTTPS协议。它是一个安全通信通道，它基于HTTP开发，用于在客户计算机和服务器之间交换信息，它使用安全套接字层(SSL)进行信息交换，简单来说它是HTTP的安全版。它是由Netscape开发并内置于其浏览器中，用于对数据进行压缩和解压操作，并返回网络上传送回的结果。HTTPS实际上应用了Netscape的安全全套接字层（SSL）作为HTTP应用层的子层。（HTTPS使用端口443，而不是象HTTP那样使用端口80来和TCP/IP进行通信。）SSL使用40 位关键字作为RC4流加密算法，这对于商业信息的加密是合适的。HTTPS和SSL支持使用X.509数字认证，如果需要的话用户可以确认发送者是谁。总的来说，HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议要比http协议安全。

1、有利于SEO优化

百度搜索引擎目前会优先收录HTTPS网站。HTTPS协议分为两种，一种是建立一个信息安全通道，来保证数据传输的安全;另一种就是确认网站的真实性。这也是很多站长要求使用https最大原因。

2、提升用户的信任度

使用HTTPS加密访问的网站一般为大型网站，例如：淘宝、京东等。如果是博客类建议使用SSL证书，对用户体验上会比较安全，避免用户信息泄露。

3、加密传输

HTTPS具有加密传输通道，有效保障隐私数据密文传输，截取也无法解密。HTTP明文传输，没有SSL加密通道，任何人都可以轻松截取或篡改机密数据。

4、防止山寨、镜像网站

防范钓鱼电子邮件一般含有恶意链接，引诱没有用户访问克隆网站。

如何添加HTTPS

　1、需要从可信的证书办法机构CA获取服务器证书。

　2、必须在WEB服务器上安装服务器证书。

　3、必须在WEB服务器上启用SSL功能。

　4、客户端(浏览器端)必须同WEB服务器信任同一个证书认证机构，即需要安装CA证书。