

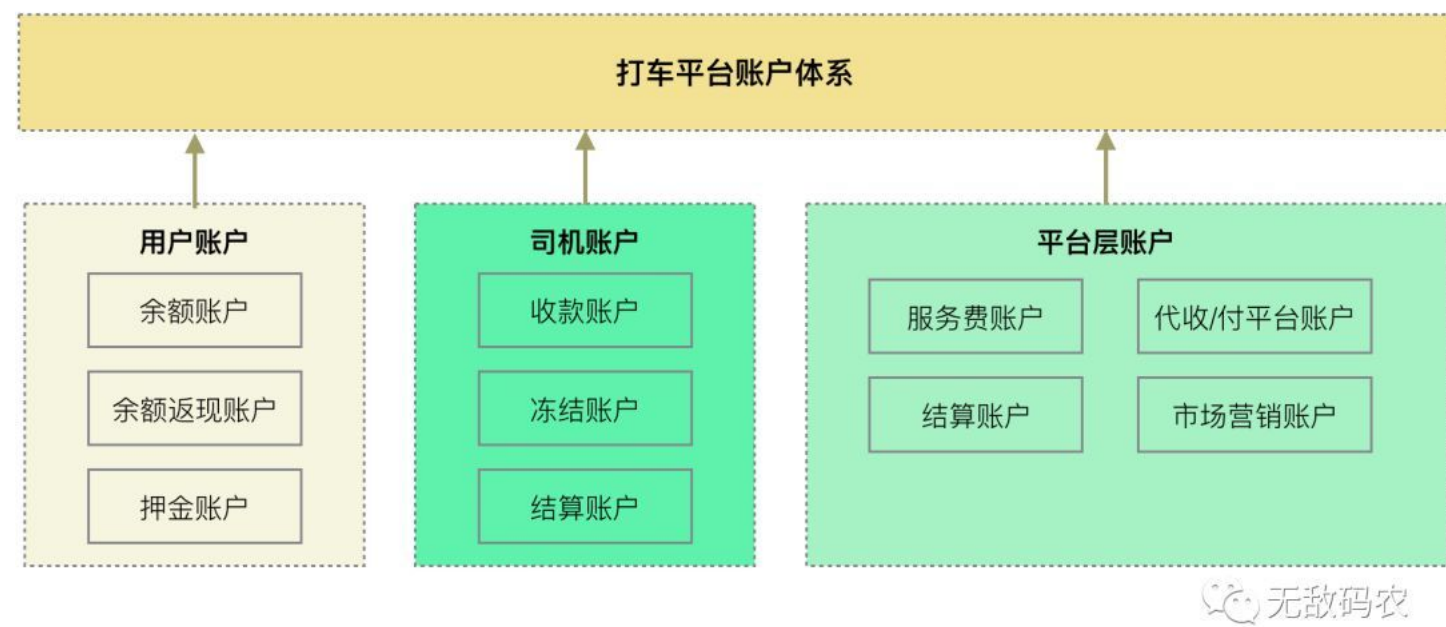
[https://www.sohu.com/a/251075013\\_684445](https://www.sohu.com/a/251075013_684445)  
技术设计

<https://www.cnblogs.com/zy12016/p/9928567.html>

支付清结算系统设计  
<https://zhuanlan.zhihu.com/p/50349784>

押金账户  
余额账户

早期设计比较混乱  
账户和客户信息表设计混乱  
没有账户流水明细、直接修改账户余额，没有完整记录账户流水或账户流水记录业务信息缺乏，特别是会造成严重的财务数据核算困难  
比如业务发展  
充值+返现业务：  
要区分真实余额和返现余额，方便余额退款和资金财务核算，责权发生制，避免公司资金损失的风险  
一次消费业务，需要调用多次账户记账，保证数据的一致性



在国内运营的租车业务司机车费代收代发这类业务是要求运营公司具备支付牌照的，没有则会牵扯到非法二清这样的法律风险，而一般来说在没有支付牌照又想继续运营的话，替代方案目前主要是采用银行资金托管的方式，即用户资金通过银行三类账户进行托管。平安见证宝、华夏的PDS、中顺易

抽象成统一的账户模型  
业务模型  
普通消费端用户、普通服务端用户、平台用户

客户、用户、账户  
客户：是用户身份信息的承载实体，例如张三这个人是一个客户；而客户也不仅仅只是针对个人，针对业务线所关联的法律主体也划分在客户的范畴，如某某打车公司  
用户：而有了基本的客户信息后，企业客户具体可以开展什么业务，普通个人用户是否使用了该企业客户提供的服务，在模型中是采用用户这个概念来承载的  
账户：企业客户开展该业务时根据业务的设计需要开通什么平台账户，普通个人用户使用该业务服务需要开通什么样的个人账户都可以根据业务的设计通过用户下设立相应地账户进行隔离区分。

记账处理：  
一是记录记账凭证，  
二是更新账户的余额

## 方案设计要点说明Q&A

- 1 延迟入账,金额扣减和充值都以流水的形式记录,由定时任务来延迟入账,可能会出现已入账数据,但是定时job还没有入账而引发金额不足的情况,但是会在下次定时job启动之后入账,业务可接受
  - 2 缓存账户余额来判定当前余额是否充足,只有缓存余额充足才会记录【流水表】,可以确保金额不会为扣成负,反过来也就是说只要是【流水表】中存在扣减流水,那么此时金额一定充足,redis是单进程单线程的,所以redis的zincrby扣减操作是线程安全的。
  - 3 精确入账,所有的入账出账操作都要先经过缓存【hotspot\_account\_currentbalance】,所以当定时任务启动的时候只会入账缓存中记录过的数据,避免循环所有的热点数据,增加定时JOB执行效率
- 【为什么定时JOB入账后要判断当前缓存中余额是否大于0】,当定时任务同步【流水表】和业务数据表之后需要把入账的数据同步到缓存中,并且当【流水表】中有充值流水的话需要把充值金额原子加入到缓存余额,来同步db和缓存这余额
- (1) 当缓存金额大于0的时候,说明目前缓存中金额充足无需同步,此时缓存中余额可能比数据库中的小,不过对业务无影响
  - (2) 当缓存金额小于0的时候,说明目前金额不足了需要将数据中的余额同步到缓存余额中,但是这个入账过程中可能会有其他成功插入到【流水表】的流水,但是到目前为止,肯定不会有扣减的数据再插入到【流水表】中了,因为金额已经为0,所以这个时候大胆放心的再冲【流水表】中看看是否有并发的数据,如果存在再更新一下业务表数据,然后把当前account中的可用余额直接set到缓存中的可用余额即可
- 4 设计方案中依靠zincrby的原子扣减操作可以保证缓存中可用余额小于等于DB中实际可用余额
- 5 【定时JOB从流水表中入账业务表数据成功,并且同步缓存中余额后为什么最后一步要删除10秒钟之前的缓存操作日志数据】,这是因为如果10秒钟之内都不存在当前账户操作的记录那么就可以认为在定时任务对当前账户进行操作的过程中没有并发对这个账户的充值和扣款操作,所以可以删除掉缓存中对这个账户ID的操作记录,【可以确定的是我们的入账出账不会超过10秒】,反过来,如果误删除掉缓存中对这个账户的操作记录,也不会对之后的定时JOB有任何影响,因为所有的入账数据都保留在【流水表中】,这个删除10秒钟的操作意在减轻定时JOB的工作量而已。

DB缓存一致性

<https://www.cnblogs.com/zy/2016/p/11076639.html>