如今，我们都非常习惯于在手机上进行支付，需要买什么，扫一下码就行了。那么，你有没有想过，整个支付的体系是什么样的呢？本文作者从方法论的角度，分析整个支付体系的运转规律和设计方法，一起来学习一下吧。

图1 支付体系分类

这两个体系我们都可以进行三层划分。

1）事

如果说人生是一场旅行，那么这场旅行本身就是一件事，旅途中每一次住酒店，每一次餐馆吃饭，每一次手机充值，每一次晚上刷剧，这些都是一件一件的事情；在酒店前台你用微信扫一扫支付了200元押金，在大厅的购物柜你用支付宝扫一扫打开柜门拿出一瓶可乐，这也是一件一件的事；微信收到了你的扫一扫请求，给了你一个弹窗，你输入了密码点了确认，这也是一件一件的事。

什么是“事”呢，我想就是基于你的意愿形成了决策，并且付出了行动，得到了结果，这一个组合就是一个“大事”，这件事里你的每一个眼神或者动作，每一次点击或者选择，每一次开心或者悲伤，我想也都是“事”。那么总有一件事，会引导你走向支付。

每一次“事”件的发生，都应该被记录，你对女朋友的好与不好，会被女朋友记录；你对别人的坏，会被上天记录；同样你那些触发了支付的“事”也会被记录；女

朋友记得是“爱情的账”，上天记得是“尘世的账”，支付记得是“系统的账”；女朋友会记到脑子里，上天会记到云彩里，支付会记到“账户”里！

所以说支付就是基于“事”记“账”。

2）账

支付的基础是“账户”，银行有银行的账户，企业有企业的账户，人行有人行的账户；账户是资金的存储，也是信用货币的基础，也是现代支付的基础；这些账户就是要记录一次次的支付的“事”，比如记录你在酒店前台用微信付了200元押金这件事。

“事”是怎么记录到账户里呢，首先我们要学会去定义“事”，就想我们去用“男女”定义“性别”，用“好坏”去定义“品行”；而对“支付事件”的定义，我们用“费用”，我们用“费用”去定义一件件发生的事情，或者说为一件件发生的事情去进行分类，只要知道了“费用”我们就知道了这是一件什么“事”，也就是事件会产生费用。

所以说支付就是将支付的“事”产生的费用记录到“账”中。

3）钱

钱是个好东西，国家离不开钱，你我离不开钱，上班为了钱，创业为了钱……这个世界的运转离不开钱，同样，支付的存在也是为了钱，为了更好地印钱，更好地存钱，更好地流通钱……钱的形式有很多种，有现金，有银行存款，有公积金，有手机话费，这些都是钱。

而现代支付社会，钱大部分存在“账”上，站在“钱”的角度去看“账”，我们发现“账”就有了不同的属性，等于钱的账就是“资金实户”，不等于“钱”的账就是“虚户”，就像微信钱包账户记着你有1万元，但是这1万元不能代表钱，而真正能代表钱的是微信在人民银行备付金账户的中那1万；所以说微信的这个“账”是虚拟的“账”，我们一般认为银行的“账”是钱，企业的“账”只是记录。

这时候，你从微信提现1万到银行卡，这是一件提现的“事”，微信扣减了你的微信钱包1万余额，这是微信基于提现的“事”产生的“提现”的费用项，记录到了你的微信“账”上；而这个时候，微信又请求人民银行将备付金中的1万转给了你的银行卡所属行，这是微信真正给了“钱”，当然这个钱也是“账”，只不过是银行的“账”所以说支付就是将支付的“事”产生的费用记录到“账”中，基于记的“账”交付真正的“钱”。

这样我们会发现，所有的支付都可以囊括到这三个字里：事、账、钱，如图2所示：

图3 事账钱的业务流转

既然支付可以用“事·账·钱”去抽象，那么怎么做事，怎么记账，怎么给钱呢？这是个好问题，也是我们产品经理要解决的问题，那就是做“系统”，建“流程”，定“逻辑”；这样我们通过建设一系列的“系统”通过一系列的“流程”在一定的“逻辑”下，实现了“做事”，“记账”，“给钱”的完美协同。

这个协同，就是我们的“支付三字真经模型”，这个模型是由“系统/流程/逻辑”组成，去实现支付的“做事，记账，给钱”，如图4所示：

图5 做事系统集合

哪些系统是记账呢？清算系统，账务系统，结算系统，会计系统，对账系统；我们划分到“记账”层，如图6所示：

图7 给钱系统集合

4. 流程

大的流程我们上面讲了，就是先做事再记账，然后给钱；那么上面也讲了有很多系统去做事，有很多系统去记账，也有很多系统去给钱；那这些系统之间谁先谁后，谁听谁的；做事这帮系统怎么告诉记账这帮系统去记账呢，记账这帮系统又是怎么去告诉给钱这帮系统去给钱呢？这就是我们支付的流程体系，如图8所示：

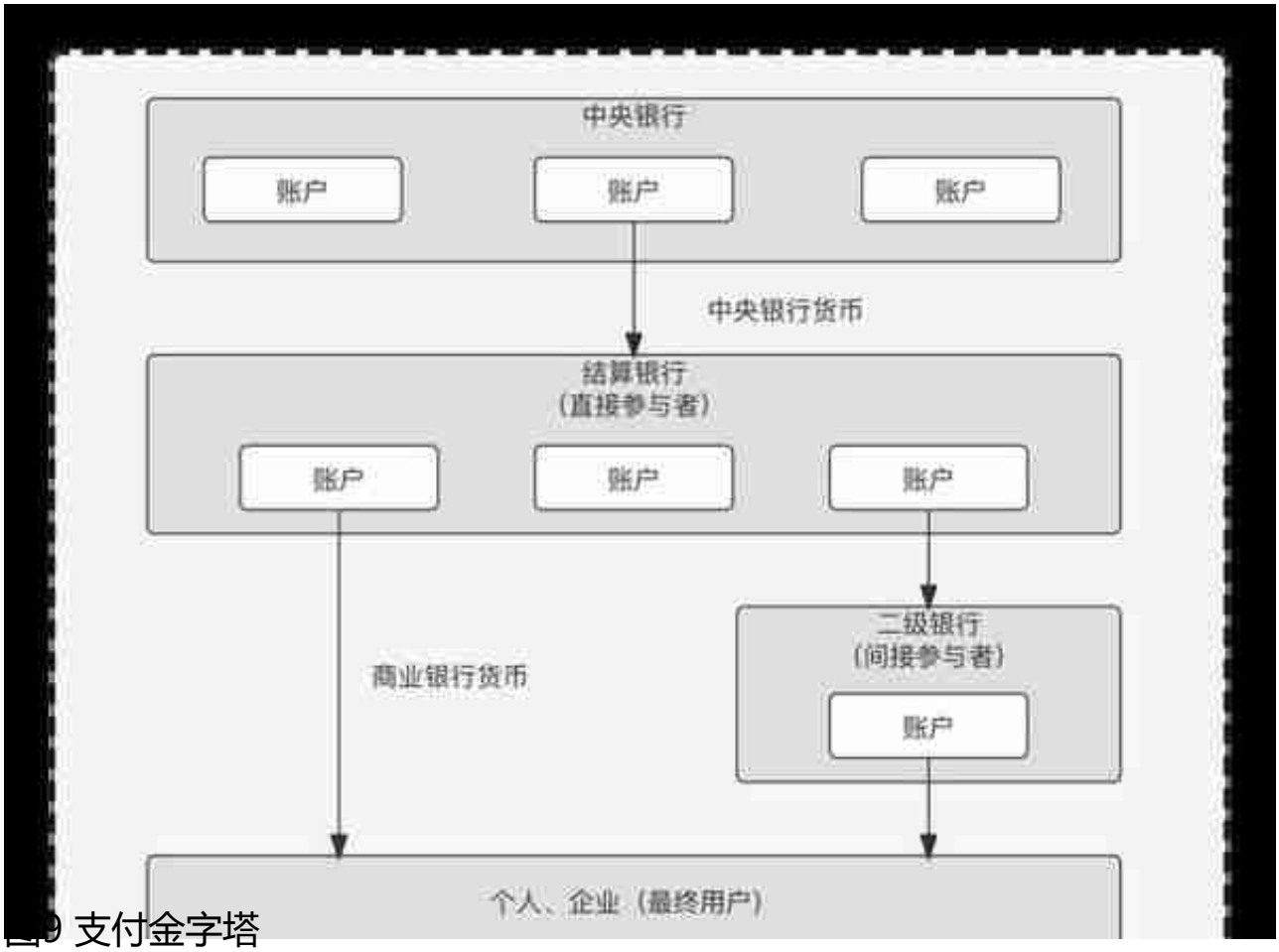

图9 支付金字塔

4. 几个关键认识

结算资产，就是在一个市场里大家需要共识一个有价值的资产用于交易的结算，比如在一个村里大家可以指定用盖有印章的砖头，而当下其实大家公认的结算资产是银行的账户存款，因为大家都认。那么大家认的前提是什么，是大家都相信这个存款可以取出“现金”，如果这个前提不成立，大家就不会相信银行，就不会把现金存到银行，更不会接受银行账户那串数字。

而这个信任的基础就是国家信用，这个信用体现在银行上就是“监管”，大家相信国家会监督银行执行；而银行之间公认的结算资产是人行的账户存款，也就是备付金，这样的信用基础才使得当下的支付体系得以运行。

所以说，我国主要的结算资产一个是银行账户存款，另一个是人行的各银行的存款；当然社会支付体系里有更多其他的结算资产，比如我们微信和支付宝内的账户资产也可以用于结算，我们日常生活的支付其实用的就是微信支付宝的账户资产进行的支付和结算。

结算协议，大家的结算要基于结算协议，就像我们用微信支付宝进行付款，那是因为我们跟微信支付宝以及我们之间存在协议，承诺接受这种资产进行支付和结算，只不过大家不关注而已

了解了支付的这些方面，是不是很多零散的知识点连接起来了！

三、“故事分析法”与大型项目实施

表达对于产品经理来说很重要，但是这个表达绝对不是像辩论选手一样秒杀一切对手，而是能够清晰地表达自己的理念和产品设计。

我们在做产品设计的时候往往也需要设定一个核心的定位，成为一个生活方式，记录美好生活……所以说我们需要刻意的训练自己去尝试设定产品的顶层设计，然后将这个理念表达出去，在这个理念之下很多问题都会有了决策的依据。

1. 学会讲故事

在表达产品设计方案的方法中，讲故事是一个很有效的方法。训练自己可以用100个字表达任何一个你做的项目、设计的系统、设计的功能，同样这种表达可以是在你设计结束以后对外的宣讲，同时也是在设计之前的思路复盘整理，而这个故事不是基于功能而是基于场景和角色的陈述。

例如，我们都知道账户中心作为对资金的记录，往往会因为一些交易场景造成账户的透支，形成了用户对平台的欠款。那么欠款就有坏账的风险，为了资金安全考虑，我们就需要一定的监控机制以及欠款的追缴策略，让用户偿还欠款，而基于实际情况这种偿还的途径可以有多种。

2. 故事里的秘密

上面的一段描述其实就是一个故事，这个故事很平淡，也描述得很清楚，任何一个

场合，无论是面对研发、测试、老板还是打扫卫生的大妈，我想大家都知道你要因为什么而做什么，这样的一段话你可能不知道，它足以引领你完成整个项目的设计，所以，我称这段描述就是“产品的故事”。我们很容易表达出自己的产品故事，就像我们可以很轻松地表达出我们饿了想吃饭。

为什么说这个故事里有很多秘密呢，我们把关键词标注出来，你发现了什么？我们都知道账户中心作为对资产的记录，往往会因为一些交易场景造成账户的透支，形成了用户对平台的欠款。那么欠款就有坏账的风险，为了资金安全考虑，我们就需要发现欠款并且能够追回来，而基于实际情况这种偿还的途径可以有多种。

故事里的关键词就像引擎的入口一样，可以帮助我们不断地打开思维的大门，慢慢地一座系统的缜密的架构就出来了，不信我们试试。

1）资产

什么是资产，不同的情况或者不同公司的用户会有不同的定义，结算款是资产，保证金是资产，在途结算款也是资产，也许信用也可以成为资产。

所以说，我们要问自己如何“定义资产”，这就是我们下一步围绕资产的分析工作，资产就有他的形式、他的度量、他的优劣等级；假如我们得出来的用户的资产有这样几种形式——结算款、保证金、在途待结算。那么这些资产怎么表达呢，账户里的好说，就是余额，那么在途结算款怎么获得呢？这不新的分析任务又来了……所以围绕资产的分析在逐步展开，而且相互自洽和补充。

2）一些交易场景

账户资产其实都是基于交易产生的，那么有些交易场景就会有不同的资产方向变化，比如什么交易场景会增加资产？什么交易场景会流失资产？这里也只有流失资产的交易场景才会造成资产的透支。那么有哪些交易场景会造成账户透支呢？这又是一个分析任务，同样不同的交易场景造成的透支是不是可以通过制度解决掉呢？这里我们又发现了一个解决问题的可能性。

3）透支

透支是一个现象，不一定所有的账户中心都允许透支，那么就算不允许透支，一些交易场景发生以后也会出现欠款，只不过这种欠款更隐蔽，比如订单退款因为余额不足而无法退款成功，那么这个待退款订单其实就是欠款的另一种表达。

那么让账户透支的好处是什么呢，其实就是让欠款显现出来，而且更容易进行核算

，比如后续的收入可以直接抵扣欠款，不然的话后续收入无法与待退订单进行抵消。所以说我们对透支有了更深的分析，当然这个分析可以继续下去。

4）欠款

欠款是资产风险的暴露，是一个结果，也是我们这次要解决的主人公。欠款一旦发生，就意味着未来可能有损失。这里像定义资产一样，我们怎么定义欠款，以什么样的形式呈现出来？假如我们以账户余额为负作为欠款的表达。

5）风险

前面说了欠款是主人公，那么风险就是我们要消灭的对象，也是我们做产品的目的，这个风险有多大，有没有达到要马上解决的地步，如果堵住了，意味着多大的价值；对风险的定义和量化，就成了我们定义产品价值的关键。

假如当下总欠款体量1000万，那么这个体量明显是需要堵住的，所以说这个产品的意义就是为了堵住1000万的风险资金缺口。

6）资金安全

堵住风险和确保资金安全是一个正面一个反面的表达，这也是作为一名资金产品经理最核心的职责，确保资金的安全。

7）发现欠款

现在欠多少钱呢，谁欠钱呢，所以说我们需要一个机制可以将风险报出来，提供给负责的人员进行评估。

8）追回来

啥是追回来，其实就是要钱，怎么要，什么时候要，所以说我们需要建立完善的追缴制度。

9）途径

既然要追钱，是不是就需要追缴的方法，用户想还钱，是不是就需要还钱的途径。另外用什么还，用保证金？直接支付欠款？每个还款方式怎么样？这里慢慢我们就开始看到了解决方案的影子……而且随着发展，这个途径会越来越多，那么这个地方也会成为一个拓展性的口子，我成为“拓展基因”以故事为起点，一篇产品大戏

慢慢拉开。

3. 学会抽象业务流程

我们做产品设计不要一上来就开始画页面，做脑图，想讲故事，然后确定主业务流程。也就是场景模拟上面的故事里，虽然我们发现了几个分析入口，其实整个故事中蕴藏一个主线，这个主线就是业务流程，而业务流程的梳理将决定了设计的好坏。

其实流程很容易，我们在表达一件事情或者思考一件事情的时候，可以从最确定的最宏观的视角入手，然后不断地去修正和补充，直到这个链条足够顺畅就像交易的主业务流程一样：用户下单，发货收货，完成，这个就很容易理解，这个理解还不会出错，那么我可以继续细化它。

选商品，下单，支付，发货，确认收货，算账，给商家结算……同样上面的故事里我们可以抽象出这样一个流程，如图10所示：

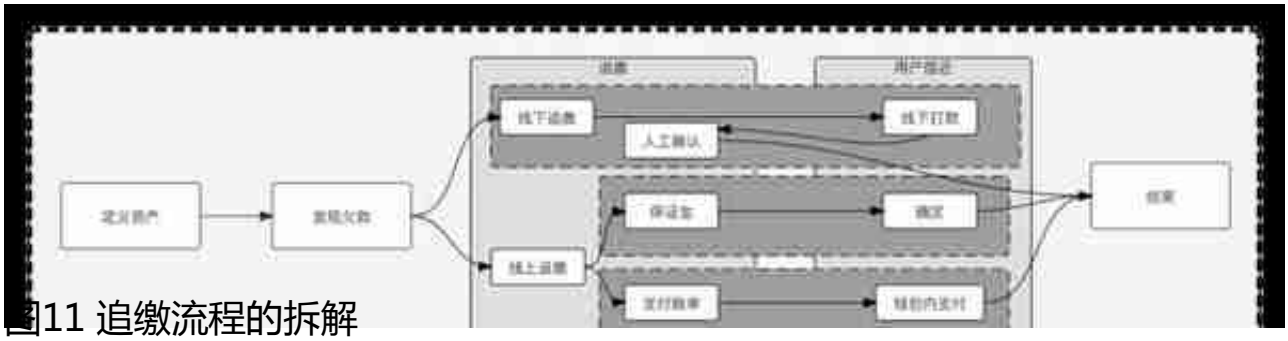

图11 追缴流程的拆解

而且从流程图中我们发现，在追缴方式上具备拓展性，随着发展，追缴方式会越来越多，而触达追缴方式的入口就是在选择要追缴某笔欠款的时候。

2）业务子流程的设计

在上面的已经拆解过的流程里，我们发现了三个子流程，那就是“不同追缴方式”的流程，而且要针对不同的追缴方式单独设计流程，如图12所示：

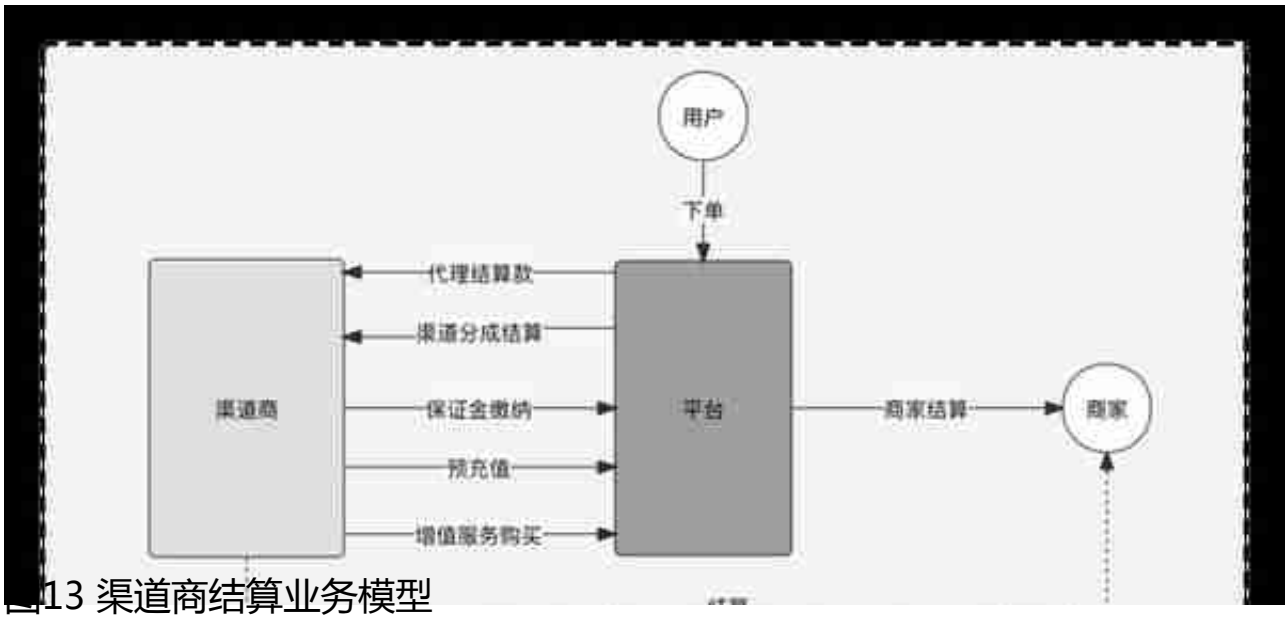

图13 渠道商结算业务模型

目前的清结算现状是除了用户下单业务以及平台跟商家之间的结算以外，其他全部环节全部由线下完成。所以说业务方的痛点是比较明显的，人力成本高、结算周期长、结算准确性差、清结算数据线上不可追溯。为了降低结算成本，提高结算效率，希望将全部环节实现线上化。

上面就是业务场景和业务痛点。

2. 落地分析

怎么解决结算线上化问题，其实我们首先要评估这个项目值不值得做，那就是看下整个业务体量以及人力成本究竟有多高，线上化以后的综合收益如何。整体评估下来我们觉得这个项目是值得做的，当然不值得做我们也不会有下面的设计了。

既然决定接这个需求了，那么下面就要思考了，这个业务模型怎么实现线上化。

这时候就要看实现路径了，一个是看看能不能复用现有的系统能力，另一个就是做一套渠道结算体系；前者相对成本较低，后者相对成本较高。如何做抉择呢？这个要看当下公司的系统模式，如果是大中台，且已经成熟，那么可以复用中台能力；如果是各业务线自治，那么可以为渠道做一套计费业务，可以服务的部分复用。

最后我们选择计费结算相关能力复用中台能力，而渠道商的保证金以及预充值和增值服务的购买等需要渠道业务自己完成渠道平台的建设。所以说这个项目就涉及到了两个大的团队，一个是中台团队，另一个就是渠道业务团队。

除了产研团队以外，还需要另外一些团队参与进来，比如运营团队参与进来负责制度制定，财务团队参与进来梳理财务诉求，资金团队参与进来负责资金账户相关事项。整个项目班子搭好以后，进行立项，开启动会，指定项目经理，拆解子项目，分派责任人……开干！！！

很不巧，陈天宇宙成了该项目的首席架构师兼项目经理；那么我第一件事要做什么呢？那就是抽象业务模型，产出清结算模型，设计渠道商计费结算线上化整体业务和产品架构。

3. 规划产品架构

这个过程首先我们要梳理所有的参与角色、计费项目、支付业务、计费对象以及关系、计费模型、结算流程。

- 参与角色涉及到了渠道商、商家、平台
- 计费项目涉及到了渠道商分成、平台佣金、商家结算款、保证金、预充值款
- 支付业务涉及到了保证金充值、预充值、服务购买、结算付款等
- 结算需要进行账期的管理，结算路径需要有直接结算和代理结算等

整个过程需要涉及到这样几个关键的流程，保证金充值流程、预付款充值流程、增值服务购买流程、计费流程、结算流程等。

通过以上的分析，我们可以整理出下面的产品架构，以及业务流程架构，如图14所示：

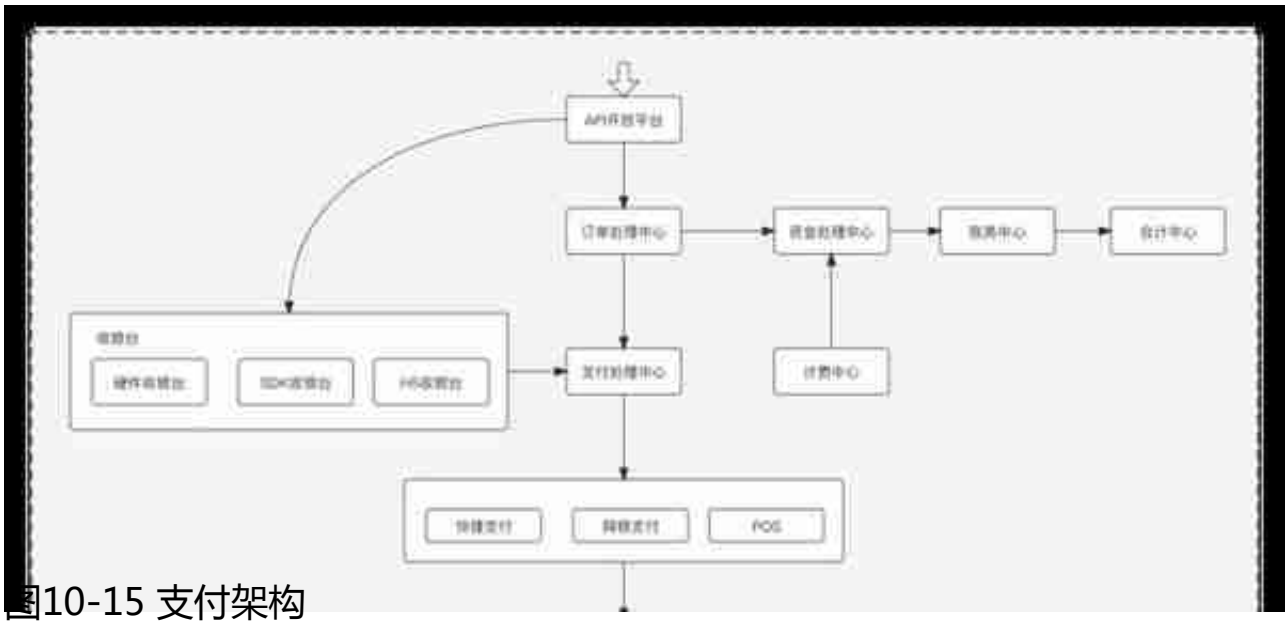

图10-15 支付架构

4. 架构就是远方

把握好“一点”的设计和“三条线”的设计，就可以搭建起一个完整的支付体系。该设计方法不仅适用于一家三方支付机构，同样适用于一家普通的交易平台，四方聚合支付。只不过支付通道的不同，三方接入的是银行通道，普通商家和四方聚合公司接入的是三方通道。

架构的意义是什么，那就是可以确保未来不会频繁重构，确保在每一个维度都具备足够的可拓展性，确保每一个模块和细节都为整体服务，同样确保产品不被业务牵着鼻子走，这是产品的顶层设计，也是走向远方的那盏灯塔。

5. 产品的意义在于解决问题

产品的意义是什么，是做出一个“厉害”的系统么，是做出一个“正确”的功能么，是遵守一个通用的规范么？我想都不是，何为正确，何为规范，谁又是标准？

产品的意义应该是用可用的手段解决当下甚至是未来的问题，所以，产品应该聚焦问题本身或者需求本身，不是系统或者功能本身。

就像很多朋友会问，账户的流水用体现“-负号”？我们总是习惯站在功能角度去设计产品，负号是一个功能；而常常忽略了，我们所面对的问题，而这才是根本。产品应具备通过功能灵活解决问题的能力，而不是仅仅是设计功能本身。

我们再看要体现流水的“负号”么？要不要呢？其实我们不应该问要不要体现负号，而是要问自己，负号要解决什么问题，进而问自己“我在面对一个什么问题”。

而这个问题就是“我需要用一种方法反映出流水的方向”，面对这个问题时，你就不应该用“用不用负号这个功能去回应”，而是“如何反应流水的方向”，当触达最本质的问题时，我们才能到达最关键的十字路口“选择”，一个产品面对问题时应该具备选择的能力。为问题选择答案，而不是仅是论证一个答案的正确与否。

反应流水的方向就不仅仅可以用负号了，可以用收支字段、借贷标识，这个时候你还会问我，流水里用不用反应负号么！当你再思考这个功能怎么设计的时候，不妨回到起点，问问自己我要解决什么样的问题，期望达到什么样的效果，我有多少可选的方案，当下的这个让我百思不解的功能是唯一答案么！放弃对模板和对权威的痴迷，而是探索自我的设计风格，培养追寻未知新事物的好奇，产品设计可以很自由且很潇洒……

六、账户迁移方法论

账户部分在之前我们讲得比较详细了，而且在线课堂也有账户的专栏，所以在此基础上我们来观摩一个实际的账户相关的案例。

这是一个大型且高危的项目，危险程度要看涉及到的资金属性、账户数量、资金体量。比如你要是几十万，那闭着眼就做了；但是如果是千万用户，百亿资金情况下，小心脏就要时刻蹦蹦跳了。

我们知道企业在发展过程中到了一定阶段，难免需要进行一些系统的重构或者数据的迁移；比如要做中台，那么业务的统一势必要将一些系统下掉，业务以及数据迁移到新的中台服务中去；我们要讲的就是“旧账户服务迁移至新账户服务”如何做，如何实现用户无感知的服务迁移

我相信，这个案例可以让你把账户掰开了揉碎了再认识一遍；而且对账户的把控力上升3个台阶；这也是高端面试过程中容易问到的问题，也是一个很容易脑子一片空白哑口无言的题目；当然这个案例也可以作为简历中一个经典的颇具竞争力的实战项目

这次我计划采用半讲解半方案文档的模式书写这篇文章；既可以让大家理解，又可以作为半成品文档提供给各位拿来即用；好了，价值讲清楚了，你准备好开始了么……

1. 项目概述

为了构建统一中台账户服务，围绕中台统一账户管理支持各业务线客户账户以及账务处理能力，需要将各业务线分散的账户业务全部切入中台账户服务中心，并且稳定后下线旧账户服务。

2. 整体方案框架

分期分批，为了确保资金安全，业务正常无停产运转，对于每个旧账户集群采取“兼平切”的方案理念，分阶段、分批次完成整个迁移工作，整个项目分三期完成。

第一期：服务兼容

在旧账户服务层兼容新服务层，或者在新服务层兼容旧服务层，或者在新旧服务之上构建新的兼容服务层。这次基于“业务无感知，用户无感知”原则，我们采用第一种方式，在旧服务层内兼容新账户服务，如图16所示：

阶段	时间	旧账户		核算	新账户	
		流水	余额		流水	余额
新账户	12	--	0		流出 100	1000
	11	--	0		流入 100	1100
关停旧	10	关闭清零	0	关闭核算		
双写期	9	流入 1000	1000	√	流入 1000	1000
	8	流出 100	0	√	流出 100	0
	7	流出 50	100	√	流出 50	100
	6	流入 100	150	√	流入 100	150
账务结转	5	转出 50	50	√	转入 50	50
	4	流出 100	50			
	3	流出 50	150			

图17 账户迁移账务处理思路

4. 账户结构设计

我们先看旧账户结构，分多类型账户，下设二级类型子账户，如图18所示：

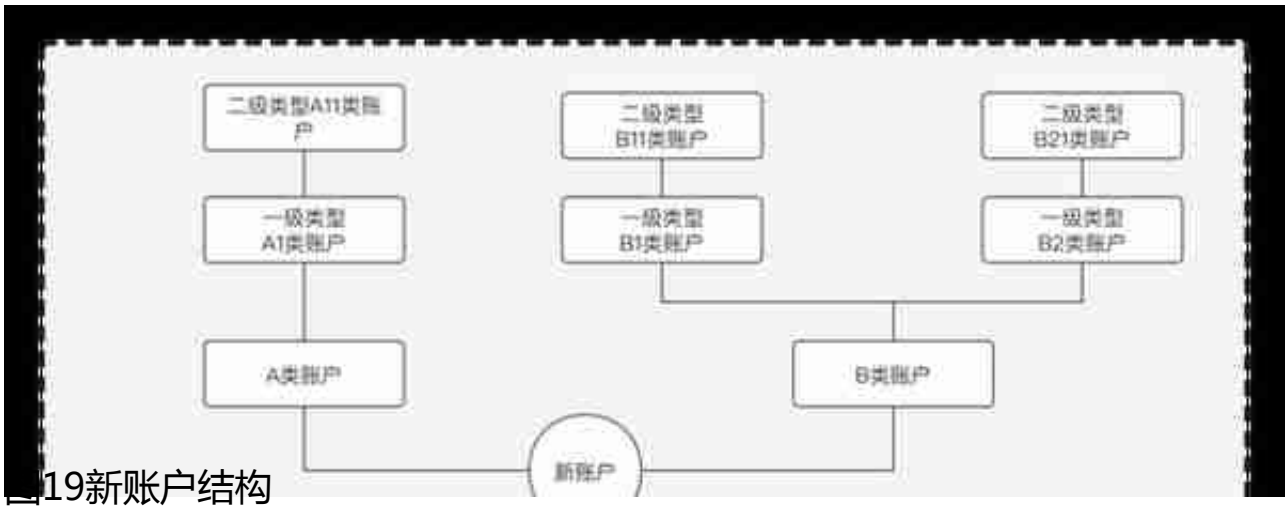

图19新账户结构

两套账户的影子关系，在末级账户上进行对应，如图20所示：

序号	账户ID	灰度	灰度状态	账户开通	旧余额结转	旧账户状态	操作
5	24356	是	灰度结束	成功	成功	停用	—
4	45466	是	成功	成功	成功	正常	—
		是	灰度中	成功	进行中	正常	—

图21 灰度管理

3）回滚方案

原则上业务层不进行回滚，技术层回滚方案由技术决定；针对出现问题的账户进行人为干预，业务上无逆向执行迁移。

4）执行

以上便是整个方案的框架了，后面就按照该方案框架进行详细的技术设计以及执行了；先执行一批灰度账户；3个月后没有问题对全部账户进行灰度；再3个月后没有问题；开始逐步关停旧账户服务。

5）复盘

整个项目完成以后，业务，运营，产品，技术等进行大复盘，了解各方业务情况，是否存在其他问题，比如财务记账准确性没有变化，正常结账是否有影响等。

专栏作家

陈天宇宙，微信公众号：陈天宇宙，人人都是产品经理专栏作家。多平台支付领域专栏作者，十年资深产品；专注为10万支付产品经理和支付机构以及企业提供深度

支付内容和服务！

题图来自 Unsplash，基于 CC0 协议

该文观点仅代表作者本人，人人都是产品经理平台仅提供信息存储空间服务。