10 | 可复用架构案例（三）：中台是如何炼成的?

2020-03-13 王庆友

架构实战案例解析

进入课程 >

讲述：王庆友

时长 14:43　大小 13.48M

你好，我是王庆友。

在第 8 讲中，我通过一个实际的订单服务案例，和你介绍了如何设计一个基础服务。今天，我就继续带你了解，如何在实际的业务场景中，通过一步步的架构升级，最后落地一个中台，实现企业级能力的复用。

通过前面的介绍，我们已经很清楚了共享服务和中台的价值，但在实践中，要不要对系统做这样的升级，我们还需要结合业务来判断，比如说：

1. **业务上有什么重大变化，导致当前系统的弊端已经很明显，不能适应业务发展了呢?**

2. **架构改造时，如何在业务、系统、资源三者之间做好平衡，对系统进行分步式的改造呢?**

我们知道，架构没有最好，只有最合适的。随着业务的发展，系统需要不断地升级，这是一个螺旋式上升的过程，如何结合当前的业务发展阶段，适时地推进架构改造，并能比较接地气地落地，是我们要追求的目标。

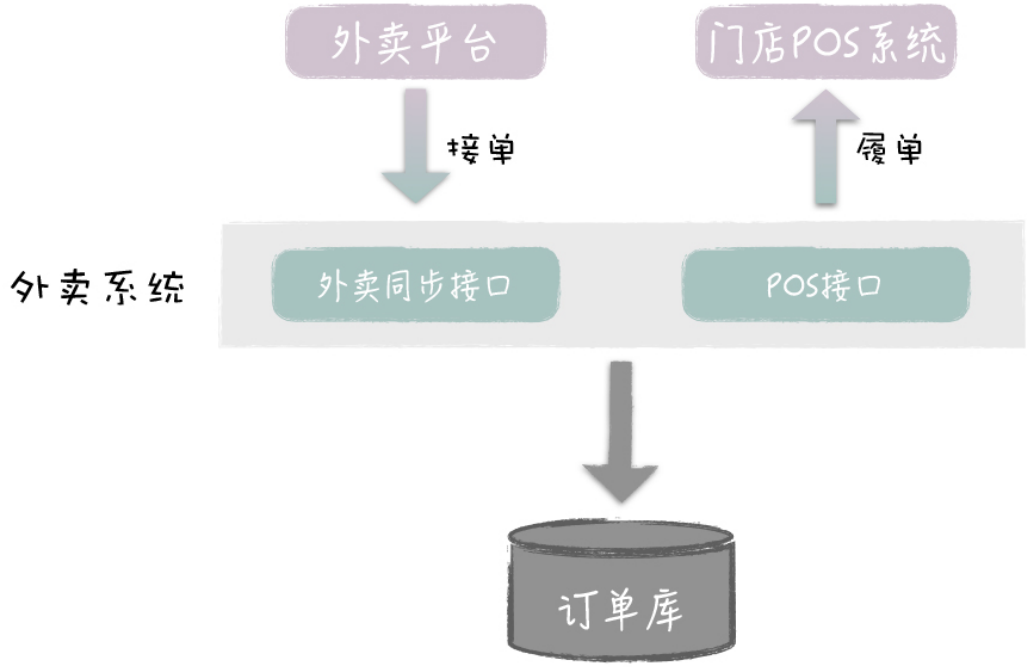
接下来，我以实际的订单系统改造为例，结合订单业务的发展和系统的痛点，为你介绍，如何推进架构从单体到共享服务、再到中台的改造过程，保证系统能够不断适配业务的升级。

先说下项目背景。公司作为供应商，为大型餐饮连锁企业打造 O2O 交易平台，包括三方聚合外卖、自有小程序、App 点餐，这些线上用户的订单最终会落到门店的收银系统，由门店进行履单。

公司的业务发展有一个变化过程，一开始只提供聚合外卖服务，后来进一步提供小程序/App 下单服务。你可以发现，整个订单处理的架构也是随着业务的变化而不断演变的，下面我就为你一一介绍。

聚合外卖订单架构

一开始，我们提供的是聚合外卖服务，相应地，系统整体架构如下图所示：



这里一共有三个系统，分别是三方外卖平台、门店收银系统以及外卖系统。其中，外卖系统是我们开发的，其他两个都是我们要对接的外部系统，接下来，我说下系统具体的交互过程。

首先，用户在三方外卖平台（如美团、饿了么）下单；然后，我们的外卖系统通过外卖平台的 API 拉取用户的订单，把订单落到本地数据库；最后，门店的收银系统访问外卖系统提供的接口获取订单，在门店内部完成履单。当然，门店履单后，收银系统会反过来同步订单状态给外卖系统，外卖系统再同步订单状态到第三方外卖平台。

你可以看到，这里的外卖系统是一个单体应用，内部包含外卖同步接口和 POS 接口两个模块。其中，**外卖同步接口**负责和第三方外卖平台对接，它主要是针对不同的外卖平台做接口适配；而 **POS 接口**负责和门店的收银系统对接。这两个模块都是使用同一个外卖订单数据库。

从**数据模型**上看，系统的订单模型也是完全按照外卖订单的需求设计的，订单状态管理也相对比较简单，因为这些订单都是用户在第三方外卖平台已经完成支付的。所以，我们的外卖系统，主要是负责管理门店履单过程中带来的订单状态变化。

从**系统架构**上看，外卖系统从外卖平台接单，然后把订单推送给后面的收银系统，只需要一个应用、一个数据库、两套接口就可以支持，使用单体架构就能很好地满足外卖的接单需求。

小程序下单架构

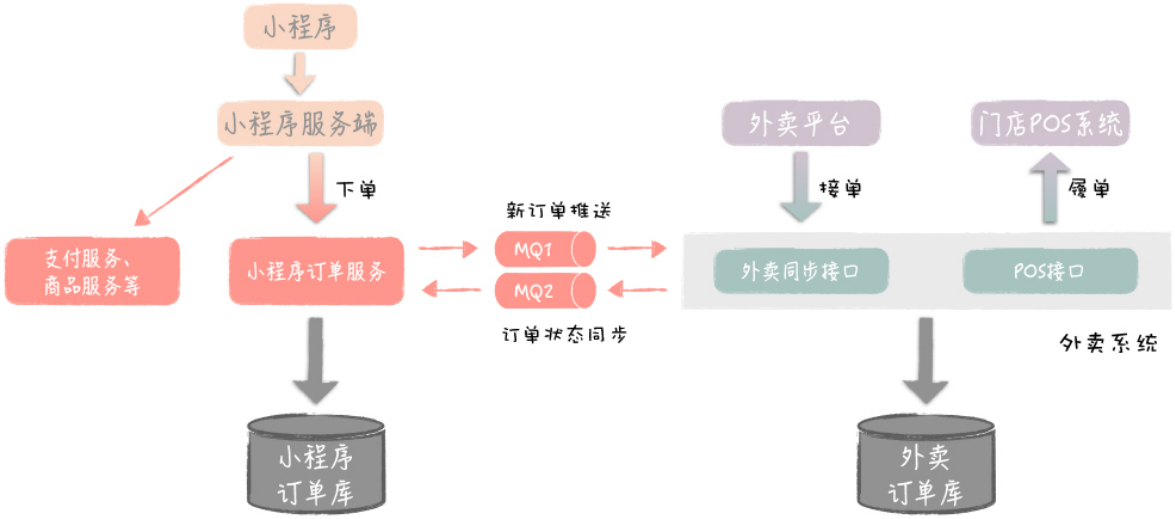
接下来，随着公司业务的升级，除了提供聚合外卖服务之外，公司还提供自有小程序的下单服务。这样，消费者既可以在三方外卖平台下单，也可以在品牌自有的小程序里下单。

不同于三方外卖订单，小程序下单平台是一个完整的业务，它包括小程序用户注册、商品和菜单浏览、商品加购物车、在线支付等等。相应地，这里会有多个基础服务对应具体业务的处理。比如，商品服务提供前台的商品浏览功能，支付服务提供用户的支付功能，这些基础服务都是由独立的小程序服务端负责整合，然后提供接口供小程序前端访问。

当用户在小程序提交订单后，小程序前端会调用服务端的下单接口，然后服务端调用订单服务，在小程序的订单库里落地订单。现在我们已经完成了前台用户的下单，但后台的订单履行怎么处理呢？这里有两种选择：

1. 小程序订单和外卖订单的处理类似，收银系统除了对接外卖系统，同时也对接小程序的订单服务。但这样一来，收银系统需要同时对接两套订单接口，它需要做大的改造。由于这是第三方的系统，我们在实践中很难落地。
2. 我们把小程序订单当作一个特殊的外卖渠道，把小程序订单推送到外卖订单库里，最终还是由外卖系统来对接收银系统，也就是相当于小程序订单直接借用了外卖订单的履单通道。

当时由于项目上线的时间比较紧急，同时从系统稳定性的角度出发，避免对收银系统做大的改造，我们采用了**第二种方式**，小程序的订单处理就嫁接在已有的外卖系统上，整个系统架构如下图所示：



你可以看到，小程序下单平台和外卖系统相对独立，同时为了更好地解耦，小程序订单服务和外卖系统之间是通过**消息系统**同步订单数据的。

这个方案是一个比较务实的选择，通过复用外卖订单的履单通路，我们也实现了小程序订单的闭环处理。表面上看，我们节省了重新搭建系统的成本，也快速落地了小程序交易这条新业务线。

但这样的架构**实际上是一种妥协**，在后续的系统运行过程中，给我们带来了很多问题：

1. 这里有两套订单系统，一套针对小程序订单，一套针对外卖订单。我们知道，两者的字段属性和订单状态定义都有不同的地方，我们把小程序的订单硬生生地套在了外卖订单的模型里，这样限制了小程序订单能力的扩展。

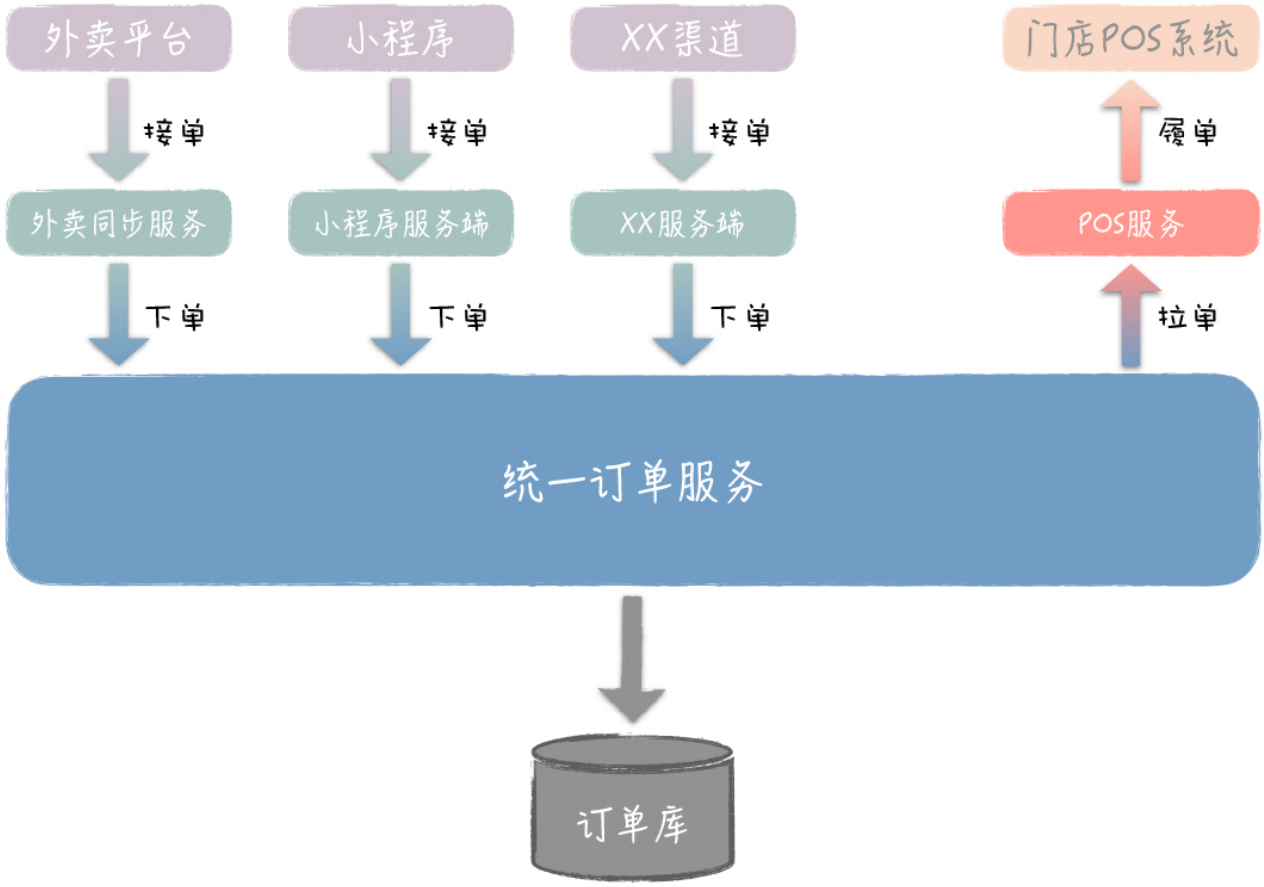
2. 小程序订单处理链路过长，从小程序服务端 -> 订单服务 -> 小程序订单数据库 -> 消息系统 -> 外卖同步接口 -> 外卖订单数据库 -> POS 接口 -> 收银系统，一共包含了 8 个处理环节，系统整体的性能和可用性都存在很大问题。比如，取餐码已经从收银系统同步给了外卖系统，但由于消息队列堵塞，外卖系统不能及时同步给小程序的订单服务，这样导致了小程序用户不能及时地看到取餐码。
3. 为了使两套订单系统解耦，我们使用了消息队列在两个库之间同步订单数据，这降低了系统整体的稳定性。实践中，也发生过多起消息队列故障导致的线上事故。

你可以发现，出现这些问题的根源是我们把小程序订单硬塞给外卖系统，一方面订单数据模型不匹配，另一方面由于这是两个系统的简单拼接，导致系统调用链路很长，影响了业务的扩展和系统的稳定性。

那有没有更好的办法，能够把这两个系统有机地结合起来呢？接下来，我们就来看下，如何通过一个统一的订单服务对两个系统进行深度的融合，从而灵活地支持多种订单业务。

统一订单服务架构

这里，我们把小程序订单服务提升为统一共享的订单服务，由它来落地所有类型的订单。对于这个统一的订单服务来说，外卖订单、小程序订单，或者是其他的新订单，都是它的下单来源，所有订单汇总在订单服务里，然后统一提供给收银系统进行履单。具体架构如下图所示：



你可以看到，系统架构经过调整，有两个大的变化：

1. 原来外卖和小程序各自有一个订单库，现在合并为了一个订单库，由这个订单服务统一对外提供订单数据的访问和状态管理。
2. 原来外卖系统的两个模块“外卖同步接口”和“POS 接口”，升级为了两个独立的应用。外卖同步接口变成外卖同步服务，对接外卖平台；POS 接口变成 POS 服务，对接门店的收银系统。它们都是通过统一订单服务存取订单数据。

经过升级，新的架构具备了明显的层次结构，自上而下分为三层：首先是各个渠道端，包括三方外卖平台、小程序前端和 POS 收银系统；然后，每个端都有相应的服务端来对接，比如外卖同步服务对接外卖平台、小程序服务端对接小程序、POS 服务对接收银系统；最后，这些服务端都统一调用底层的订单服务。

在这个架构里，如果我们要增加新的下单渠道，就非常方便，比如要支持 App 下单，我们提供 App 服务端即可；要新增加后台履单方式也非常方便，比如对于新的电子卡券类订单，它不需要经过收银系统，可以直接由企业的 OMS 系统（Order Management System，订单管理系统）处理，要实现这样的业务，我们只需新增加一个和 OMS 系统的

适配应用就可以了。所以，**这里就不仅仅是一个外卖订单和小程序订单的处理平台，而是升级成了一个完整的全渠道交易平台。**

同时，订单处理的链路大大缩短，从小程序服务端 -> 订单服务 -> 订单数据库 -> POS 服务 -> 收银系统，只有 5 个节点，相比之前减少了 3 个，系统的可用性和端到端的性能得到了大幅度的提升。

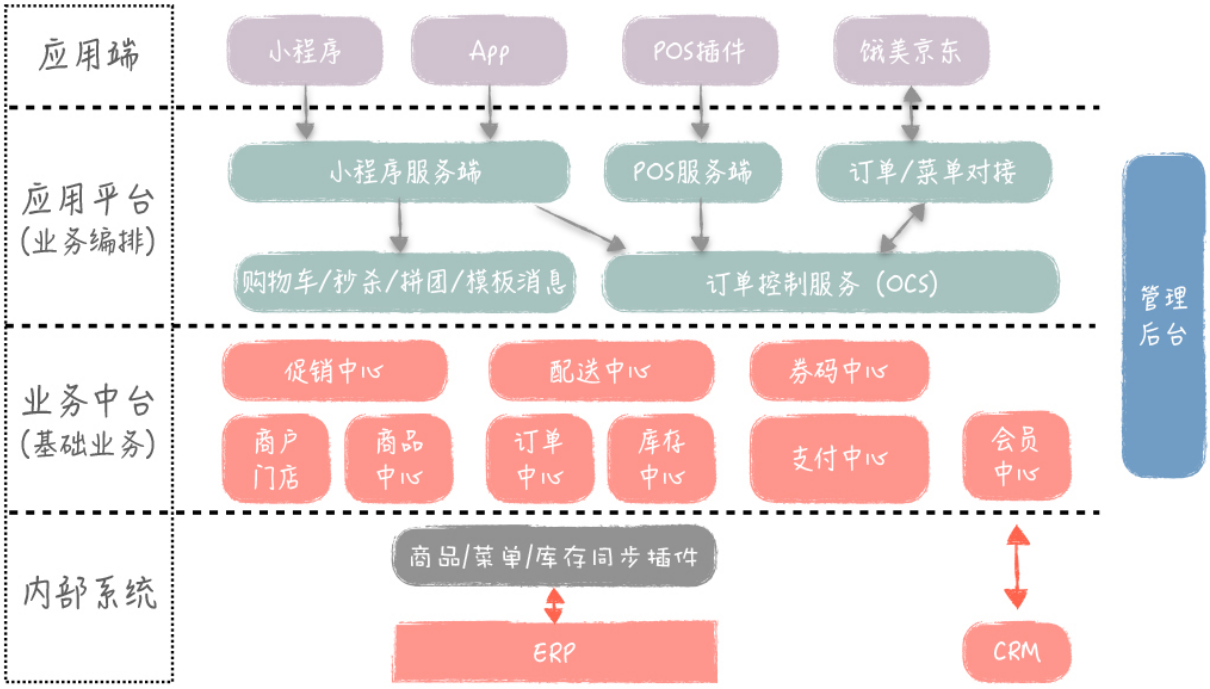
最后，统一订单服务实现了统一的订单属性定义、统一的订单状态管理，以及订单数据的集中存储，这对后续的 BI 分析和数据中台建设非常有帮助。它们处理数据时，只需要从一个订单库拉取数据，解析一个订单数据模型就可以了。

中台架构

上面的统一订单服务整合了外卖和小程序的订单，并且为新的下单渠道预留扩展。按照同样的思路，我们可以构建统一的商品服务，同时满足外卖和小程序上商品的管理；可以构建统一的促销服务，同时支持线上和线下的促销活动；也可以构建统一的库存服务，实现线上和线下库存的同步和共享等等。

通过构建这样一系列的共享服务，我们就实现了各个渠道业务规则和业务数据的统一管理，最终我们落地了一个强大的业务中台，可以很方便地扩展各个业务，实现企业整体业务能力的复用。

最后，实际项目的中台架构如下图所示：



在这个架构中，**前端**有 3 个业务场景，分别是小程序点单、App 商城下单、外卖平台下单，每个业务场景都有相应的**服务端**负责对接。在各个服务端下面，还有一些**辅助的应用**，如购物车、秒杀、拼团等等。同时这里还有一个**订单控制服务**（Order Control Service，OCS），负责订单逻辑的编排以及前后台之间的状态同步，你可以把它看作是基础服务之上的聚合服务。

再底下就是核心的**业务中台**，它由 9 大服务中心组成，这些中心和商户内部系统进行对接。其中，商品中心和库存中心对接 ERP 系统，会员中心对接 CRM 系统，订单中心对接 POS 收银系统，这里的对接分别由对应的适配插件负责。

通过这个订单业务改造落地后的中台架构，你可以看到，中台由各个通用的基础服务构成，它是相对标准的；而插件是定制的，具体和每个企业的后台系统有关。这样，通过共享服务和中台，我们就把企业内部基础设施和线上业务场景有效地打通了，从系统架构的层面，为企业的全面数字化转型打下了良好的基础。

总结

今天，我从一个企业的订单业务变化出发，为你介绍了为什么要落地一个统一的订单服务，以及如何落地，并通过打造一系列类似的共享服务，逐步升级系统到中台架构。

相信通过这个实际案例，你进一步理解了如何通过共享服务和中台，实现业务能力的复用，并能根据公司的业务发展阶段，选择合适的时机、合适的架构，以接地气的方式对系统进行

逐步改造。

最后，给你留一道思考题：目前你的公司有没有落地共享服务，它是怎么逐步演变过来的呢？

欢迎你在留言区与大家分享你的答案，如果你在学习和实践的过程中，有什么问题或者思考，也欢迎给我留言，我们一起讨论。感谢阅读，我们下期再见。

上一篇　09 | 可复用架构案例（二）：如何对现有系统做微服务改造？

下一篇　11 | 技术架构：作为开发，你真的了解系统吗？

精选留言 (7)

写留言

Jeff.Smile

2020-03-13

老师，从功能上看，第二层的应用平台层是否可以省掉，直接由前端比如小程序或者app等渠道直接发起request调用中台呢？第二层主要作用是什么？

展开

作者回复: 前端直接调中台的服务不合适，主要有两个原因:
1. 对于外部过来的请求，我们需要提供一些非业务性的功能，比如签名验证，协议和参数适配（外部的rest和内部的rpc）
2. 中台只是提供基础业务功能，前端过来的请求是代表一个业务场景，需要同时用到多个服务的功能，比如前台下单，需要用到用户服务，商品服务，库存服务，订单服务等，这不合适直接在前端做功能整合。

4 1

Alex
2020-03-16

用户中心估计是大家第一个要动手的共享服务吧

展开

嬰
2020-03-15

老师，请教一下在文中这套系统中，用户的数据结构应该是什么样子，因为这里有小程序用户，有外卖第三方平台用户，那么用户系统的数据表是不是一张大表，包含了所有小程序和外卖用户需要的信息？如果是这样，后面在对接其他新业务用户，那么就要修改数据表扩展新字段，这好像不是很好，那么是不是可以认为内部有多个表存储这这些用户数据，然后订单里面在关联不同的用户id?

展开

作者回复: 用户基本信息表就一张，同一个用户就一条记录。
然后有一个用户渠道表，一个用户一个渠道一条记录，比如小程序渠道，公众号渠道，支付宝渠道就有三条记录。用户基本信息表和用户渠道表1对多关联。

孙同学
2020-03-14

https://www.processon.com/view/link/5e51378ce4b0c037b5f9d1e3 学习整理更新

粗线条Jackie
2020-03-13

仔细研读了老师的每一期课程，受益匪浅。我们团队现有的Ecommerce平台，由最初的电

话下单一体式应用，逐步演化成支持了PC Web、小程序、企业App和第三方外卖平台的SOA服务化架构。比对了老师的文中介绍，感觉目前我们的服务模块更像是一个"应用平台"+"业务中台"的结合体，供前端应用直接调用，向下与连锁门店和POS打通，为了适配业务的快速变化，现有服务模块的更新频次过高和模块间的紧耦合是比较突出的问题。...
展开

作者回复: 赞一个，能够结合实际做很深入的思考。试着回答你的问题：
1. OMS是一个很重要的概念，说下我理解的OMS职责。当订单生成后，OMS就开始工作，比如对订单进行审核，决定订单流转到哪个仓库履行，交给哪个配送商进行配送，OMS是一个决策和调度中心，自己不具体干活。它和订单中心还不是一回事，一般说它在订单中心后面，会通过订单中心读取数据，并修改订单的状态。
#2，这个OMS看起来更像一个订单中心，不知它有没有提供后续订单如何处理的决策。
#3 业务中台，共享服务划分和业务紧密相关，DDD是一种方法和工具，可以使用它来划分，DDD概念拆的很细，给我们提供很清晰的分析思路，但也是有点繁琐，如果能够参考而不是照搬它那就最好。

tt
2020-03-13

老师，有一个问题，如果中台中的模块是由若干的不同的厂商开发的，要如何保证部署的独立性呢。一个基础服务版本升级，现有接口没有变化，增加若干新的接口，还如何组织测试呢?

作者回复: 一般来说，中台总体上是由一个供应商开发，其他供应商可以基于中台做具体业务场景开发。

阿男
2020-03-13

老师您好，如果是传统It公司，所做的系统没有这么大的体量，领导又想上个中台，有没有好的下手思路?

作者回复: 可以根据当前c端最急需的，慢慢落地一个个能力，比如会员中台，营销中台，把前后台给打通。