CODE A BETTER LIFE
一 行 代 码　亿 万 生 活

美团

旅游电商架构攻略指南

美团技术沙龙-第38期

金孟哲

美团 境内度假研发组

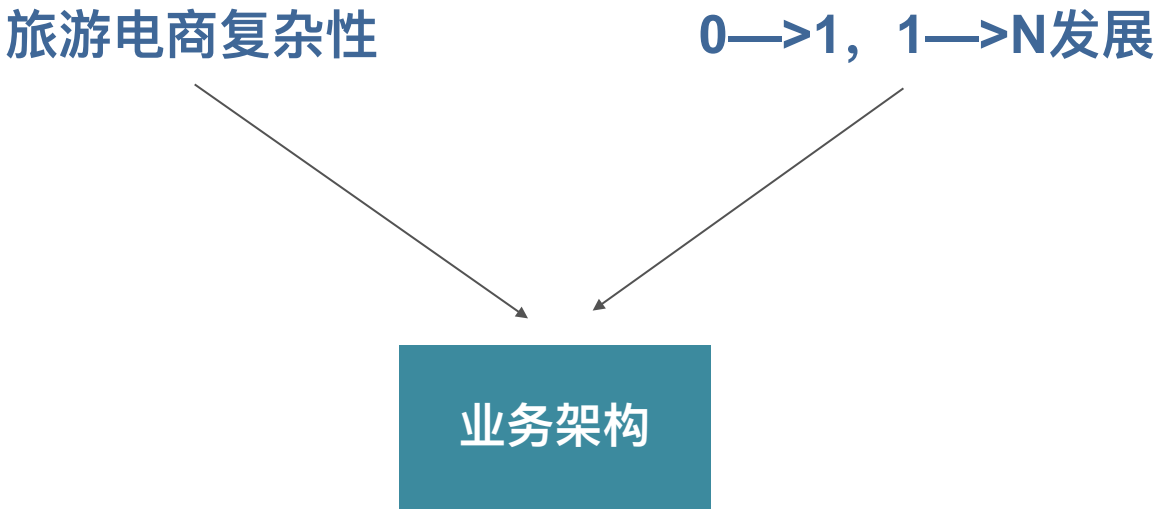
2015.4 加入美团

2016.3 开始深耕度假业务，从0—>1—>N历经系统多轮演进

目前负责度假预订对接平台

为什么选择业务架构演进的主题

美团旅行


旅游电商复杂性
0—>1，1—>N发展
业务架构

遇到很多问题、踩过很多坑，

也有一些实践和经验、一直在努力。

技术专家—金孟哲

旅游度假预订系统架构演进

架构师—王超

多旅游要素动态打包实践

技术专家—徐泼

旅游度假商品中心架构演进

直连低可信系统
实现高可用服务

架构师—郑旭

CODE A BETTER LIFE

一 行 代 码　　亿 万 生 活

更多技术干货
欢迎关注“美团技术团队”

招聘：各种岗位

邮箱：jinmengzhe@meituan.com

美团

CODE A BETTER LIFE
一 行 代 码　亿 万 生 活

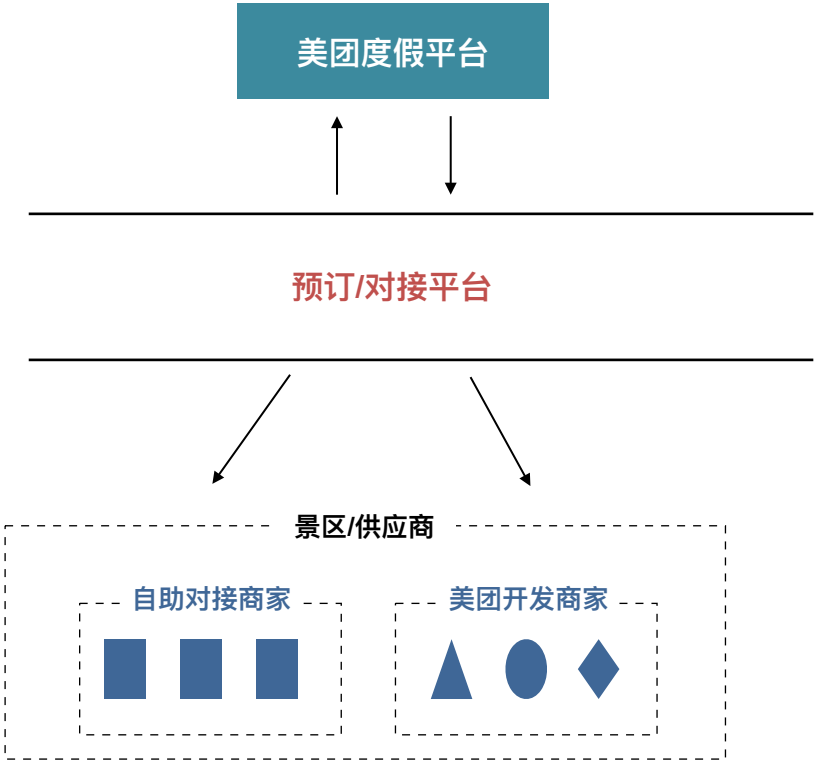
旅游度假预订系统架构演进实践

金孟哲 @美团

2018.7

目录

背景介绍



业务背景

- **业务品类**: 门票为主 + 剧场票、景酒…
- **合作模式**: 商家自助对接 + 美团主动开发
- **核心体验**: 对接规模 + 预订体验

系统背景

- **美团度假平台**: 用户系统、交易系统、供应链/商品系统、运营系统
- **预订/对接平台**
  - 屏蔽差异、高效对接
  - 策略控制、提升体验
- **景区/供应商**: 外部系统接口/稳定性较差

背景介绍

粗放式增长

业务阶段: 粗放增长

- 自助对接高速增长
- 我方开发高速增长

系统挑战

- 功能缺失、BUG
- 基础设施/不完善、效率低

高速发展

业务阶段: 高速发展

- 规模高速增长
- 体验大幅提升

系统挑战

- 架构耦合，外部风险
- 稳定性

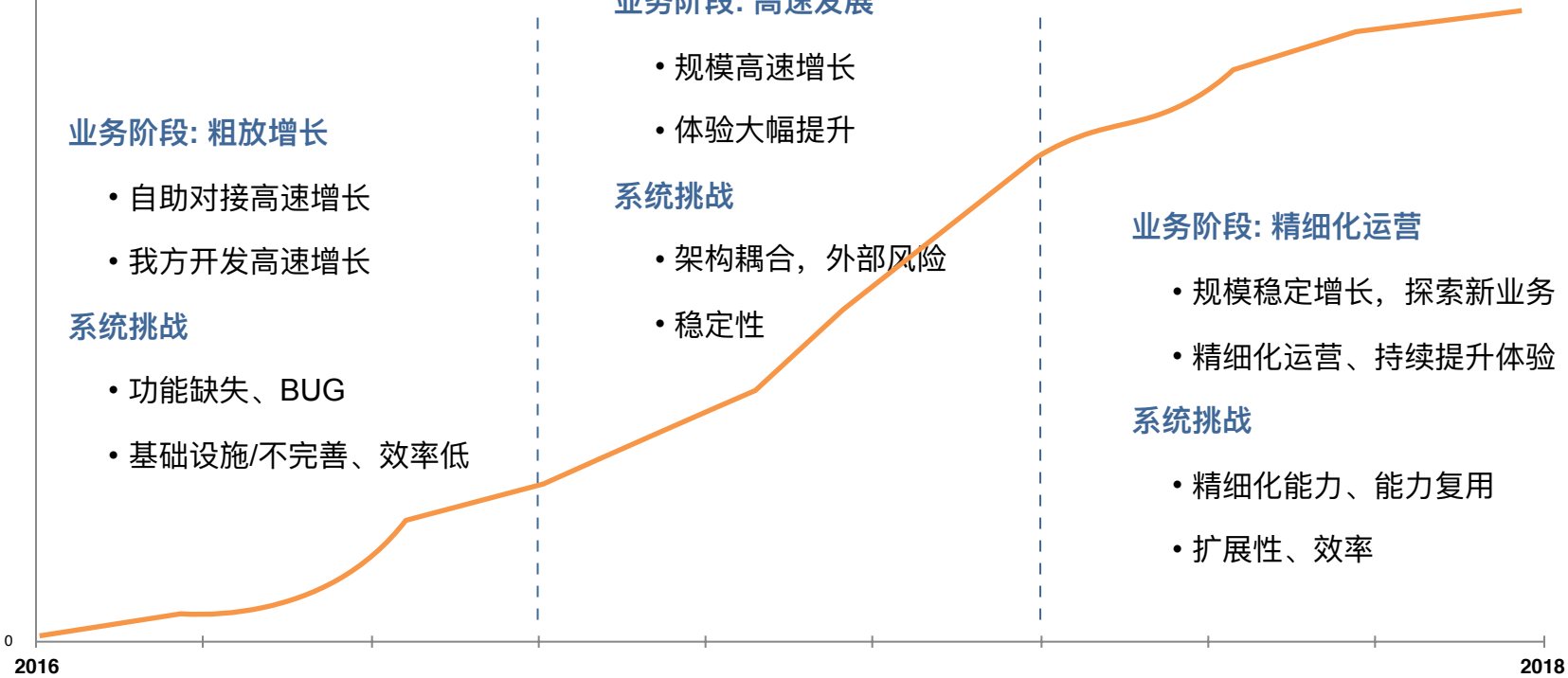
精细化运营

业务阶段: 精细化运营

- 规模稳定增长，探索新业务
- 精细化运营、持续提升体验

系统挑战

- 精细化能力、能力复用
- 扩展性、效率



目录

问题和挑战—对接复杂性

差异性

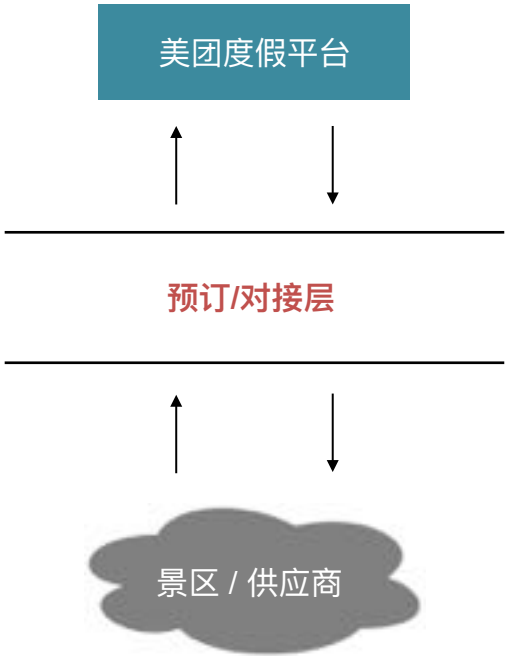
- **行业非标**: 线上化率**~10%** << 机票/酒店
- **对接差异**: 商家对接 + 美团对接
- **商家差异**: 商家数量，接口差异

效率

- **对接规模**: **数千、万**
- **重复性**: **O(N)**

质量

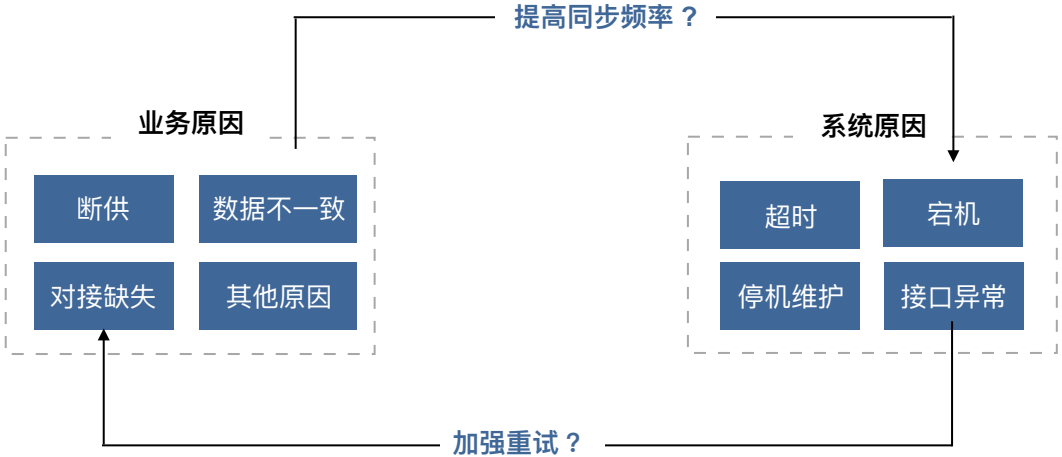
- **领域边界**: 对接 VS 预订
- **迭代质量**: 新对接 VS 已对接



问题和挑战—预订体验

商家业务
- 供应商数：数千~万
- 失败原因：数千~万量级

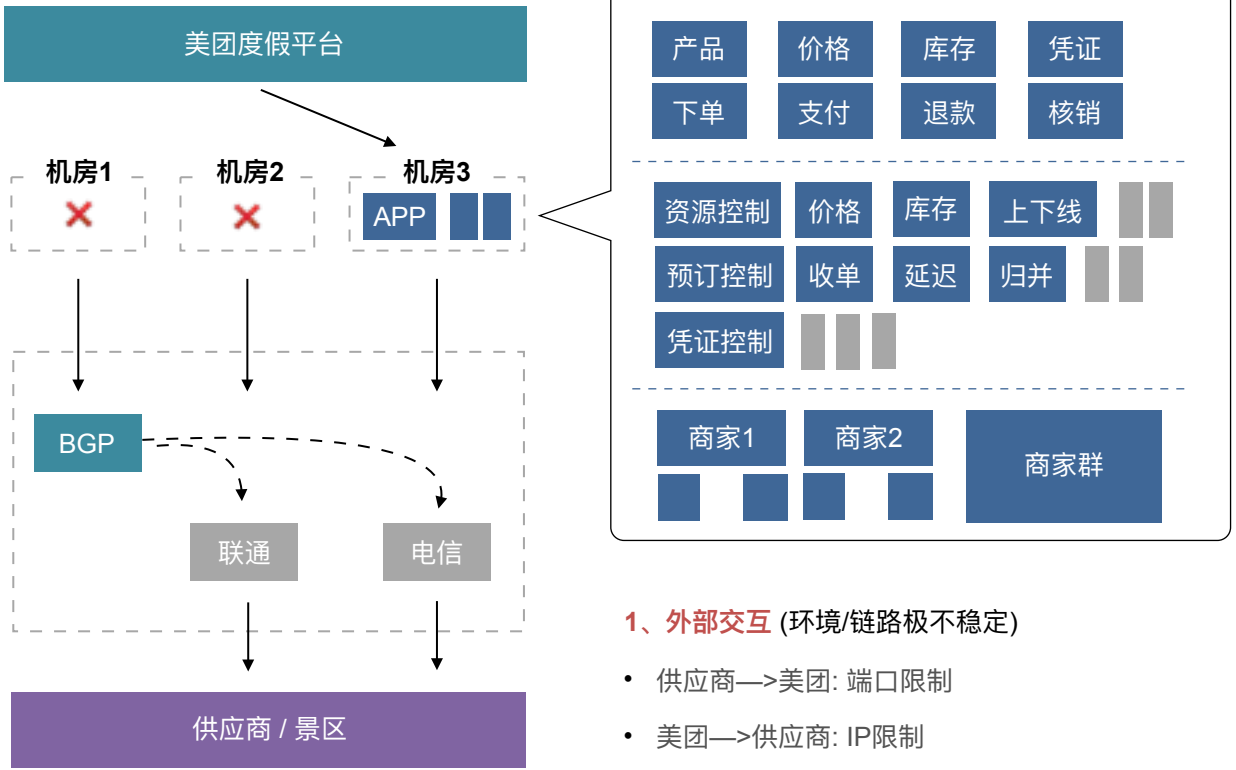
商家技术
- TP90：2~4s
- TP99：> 1min
- QPS：< 5
- 停服宕机：10+次/周
- 超时：24h不间断



原始预订体验

- 预订成功率：X正常用户无法容忍
- 退款成功率：X正常用户无法容忍
- 履约闭环率：X正常用户无法容忍

问题和挑战—稳定性/扩展性



1、外部交互 (环境/链路极不稳定)

- 供应商—>美团: 端口限制
- 美团—>供应商: IP限制
- 供应商: 超时/故障/宕机/攻击

2、领域耦合

- 产品 + 交易 + 预订 + 对接
- 预订功能 + 预订控制
- 标准化 + VIP

3、VIP差异化

- VIP各不相同，GMV占比高
- 稳定性 / 扩展性

目录

架构演进实践—整体策略

粗放式增长	高速发展	精细化运营
• **快速实现**: 可能all in one	• **构建能力**: 业务&系统核心能力	• **能力复用**: 扩展性 / 精细化
• **系统完善**: 基础设施、工具	• **体验/效率并重**: 规模&体验高速提升	• **稳定性/扩展性**: 持续发展的保障
• **关注效率**: 对接、预订规模	• **合理架构**: 业务&系统复杂度急剧增加	• **架构标准化**: 能力标准化，提效

架构演进实践—对接能力

分而治之

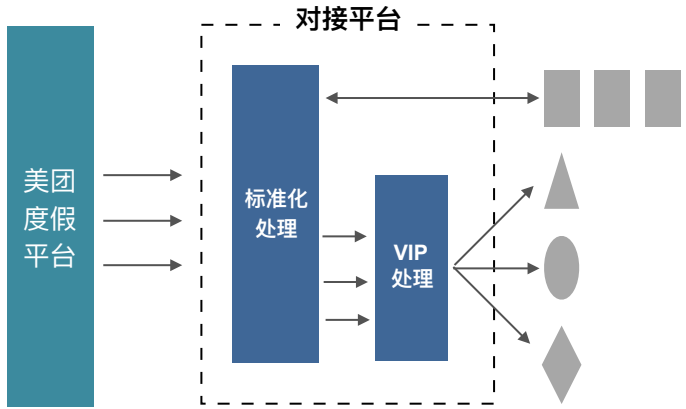
- **标准化能力**：中小供应商，自助对接 (**开放平台**)
- **差异化能力**：VIP供应商，美团开发 (**VIP平台**)

标准化建设

- **业务标准化**：工具/流程—>自助对接
- **技术标准化**：技术基础设施沉淀

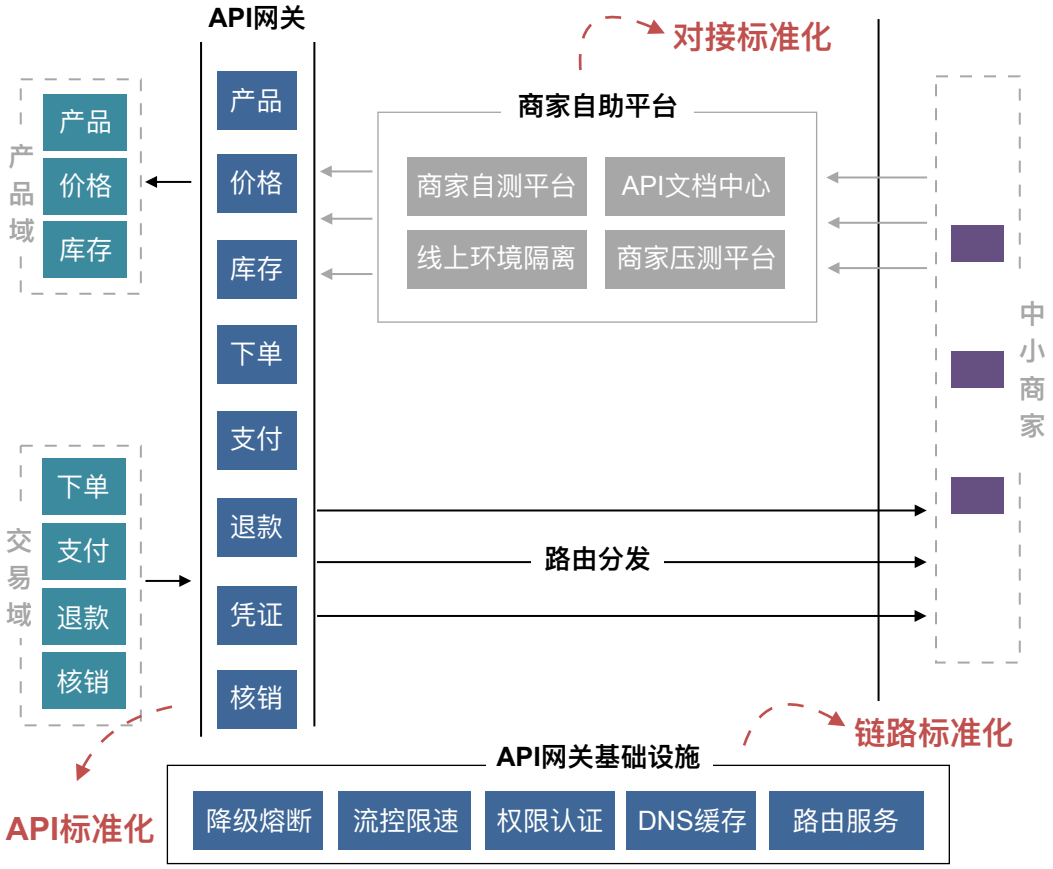
差异化建设

- **统一对接框架**：统一流程，全局时间复杂度降低
- **共性配置组件**：抽象共性，局部时间复杂度降低



核心策略:分而治之

架构演进实践—开放平台



API标准化

- 商家对接闭环
- 用户服务闭环

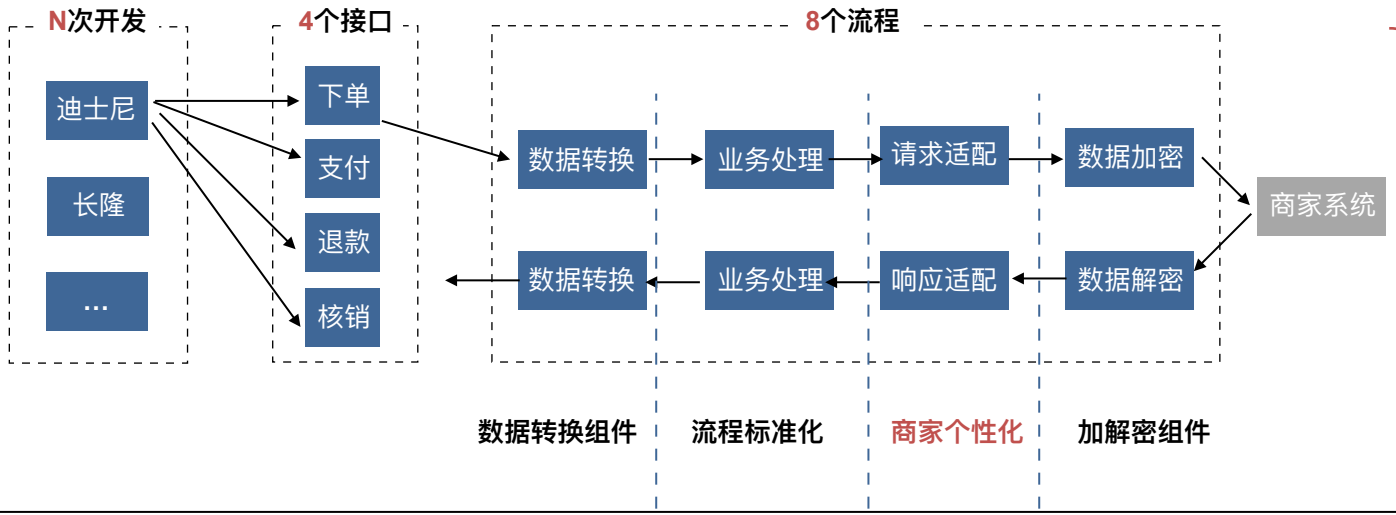
对接标准化

- 商家对接流程
- 自助文档、工具

链路标准化

- 技术基础设施
- 应对外部风险

架构演进实践—VIP平台(统一框架)



整体复杂度降低

工作量: **1 / 8**

1、数据转换组件

- 标准请求—>VIP请求格式转换
- 1次开发

2、流程标准化

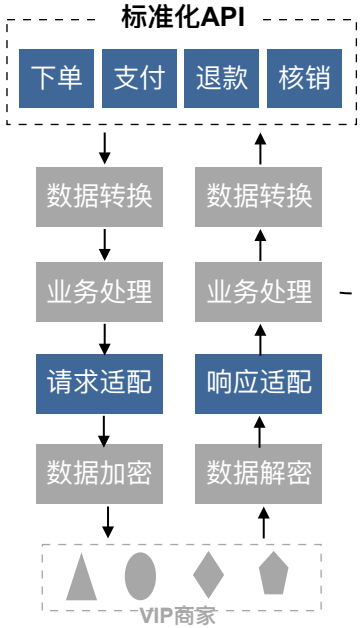
- 下单/支付/退款/核销4个接口标准逻辑
- 1次开发

3、加解密组件: 可以枚举

4、请求/响应适配: 商家个性化部分

架构演进实践—演进思路

全局复杂度降低

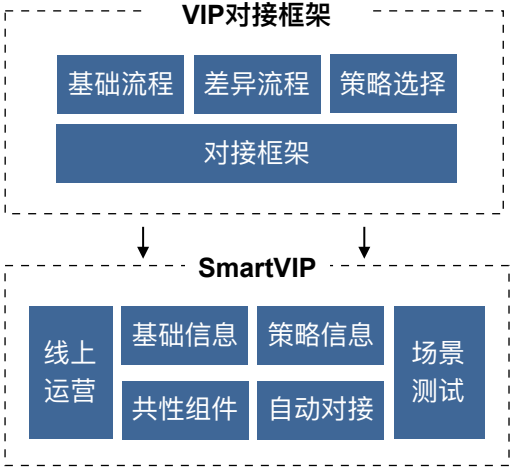


- 标准化和差异化分离
- 统一对接框架

局部复杂度降低

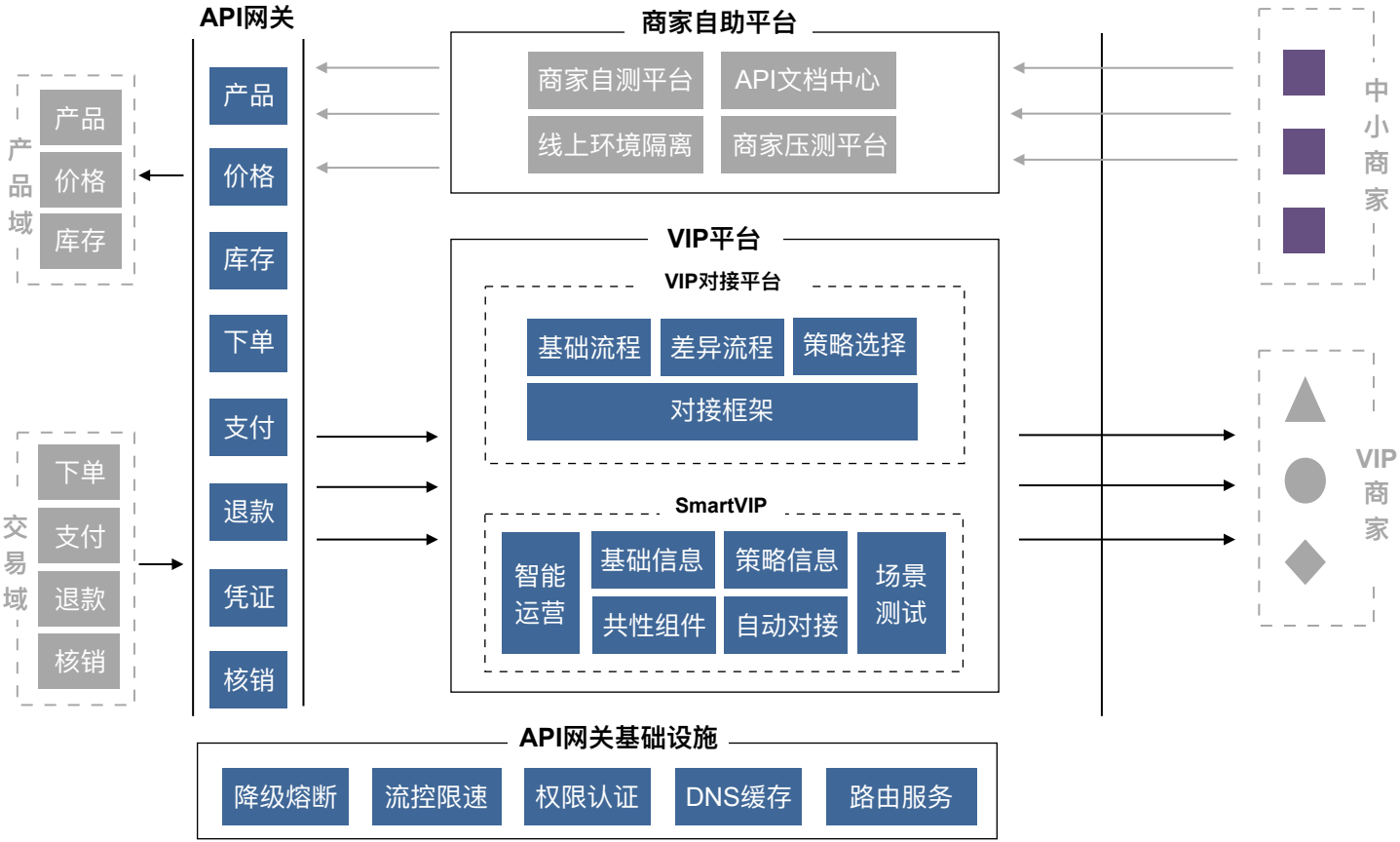
- 局部复杂度: O(N)—>O(1)
- 具备横向能力

全面提效/稳定性



- 消除/减少开发—>配置实现
- 稳定性建设: 应对VIP差异性

架构演进实践—对接平台架构



架构演进实践—总结

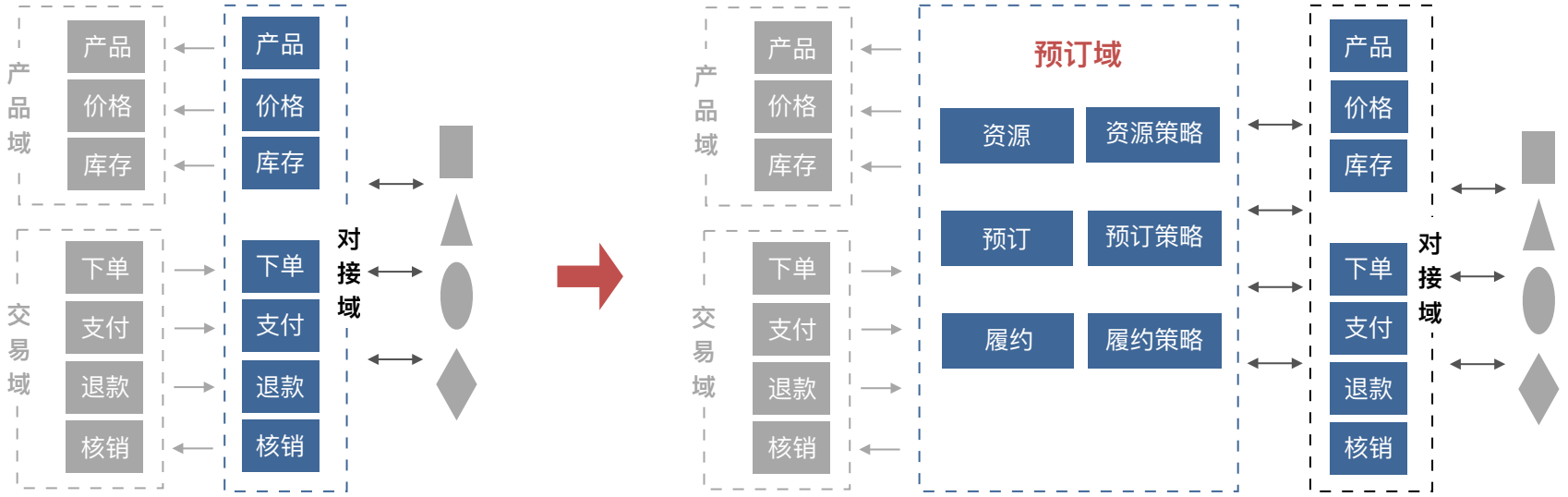
1、**标准化与差异化分离**

2、**整体复杂度降低** (标准化 + 统一框架)

3、**局部复杂度降低** (差异中抽象共性)

对接复杂性

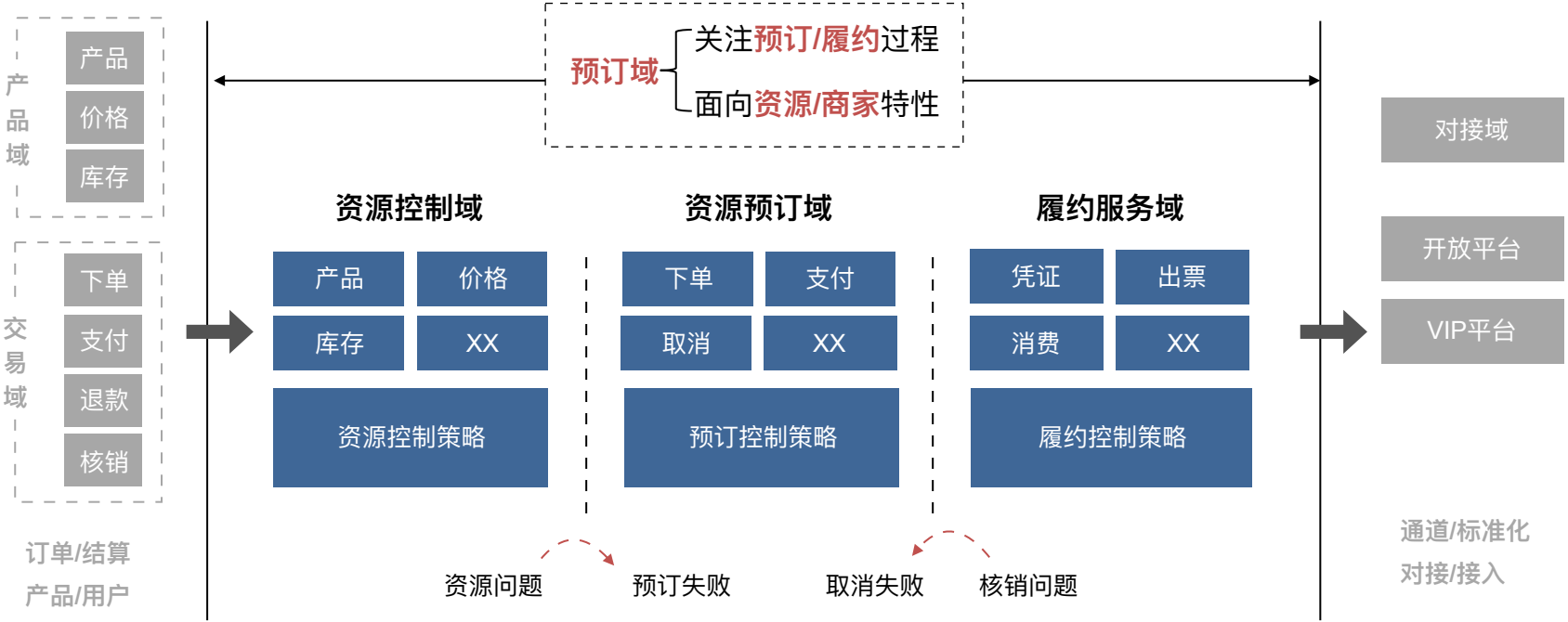
架构演进实践—预订能力(问题&思路)



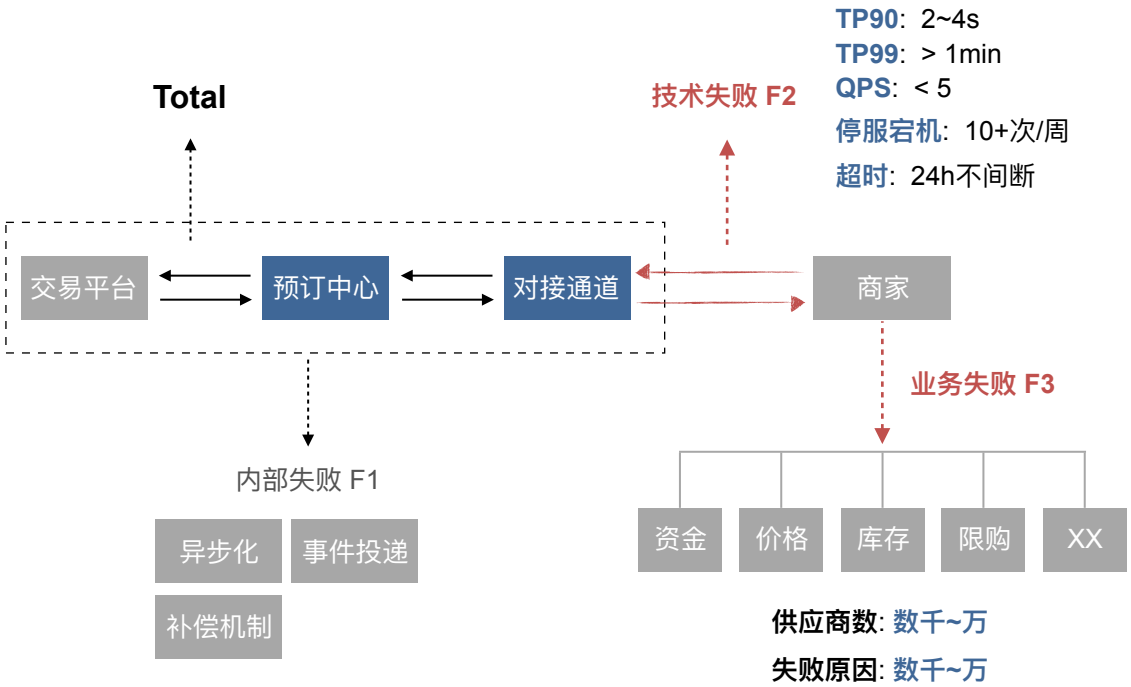
- **通道限制**: 通道能力—>无法对业务和技术问题**精细控制**
- **领域限制**
  - 外部: **产品 + 交易 + 预订 + 对接**边界不清
  - 内部: 预订内部业务领域复杂
- **策略限制**: 系统架构无法支持**灵活 + 统一**的策略控制

- **领域分离**
  - 产品 + 交易 + 预订 + 对接
  - 预订内部: 资源 + 预订 + 履约
- **策略控制**
  - 流程与策略分离，策略平台化

架构演进实践—预订领域


产品域
产品
价格
库存
交易域
下单
支付
退款
核销
订单/结算
产品/用户
预订域
关注预订/履约过程
面向资源/商家特性
资源控制域
产品
价格
库存
XX
资源控制策略
资源预订域
下单
支付
取消
XX
预订控制策略
履约服务域
凭证
出票
消费
XX
履约控制策略
资源问题
预订失败
取消失败
核销问题
对接域
开放平台
VIP平台
通道/标准化
对接/接入

架构演进实践—资源控制



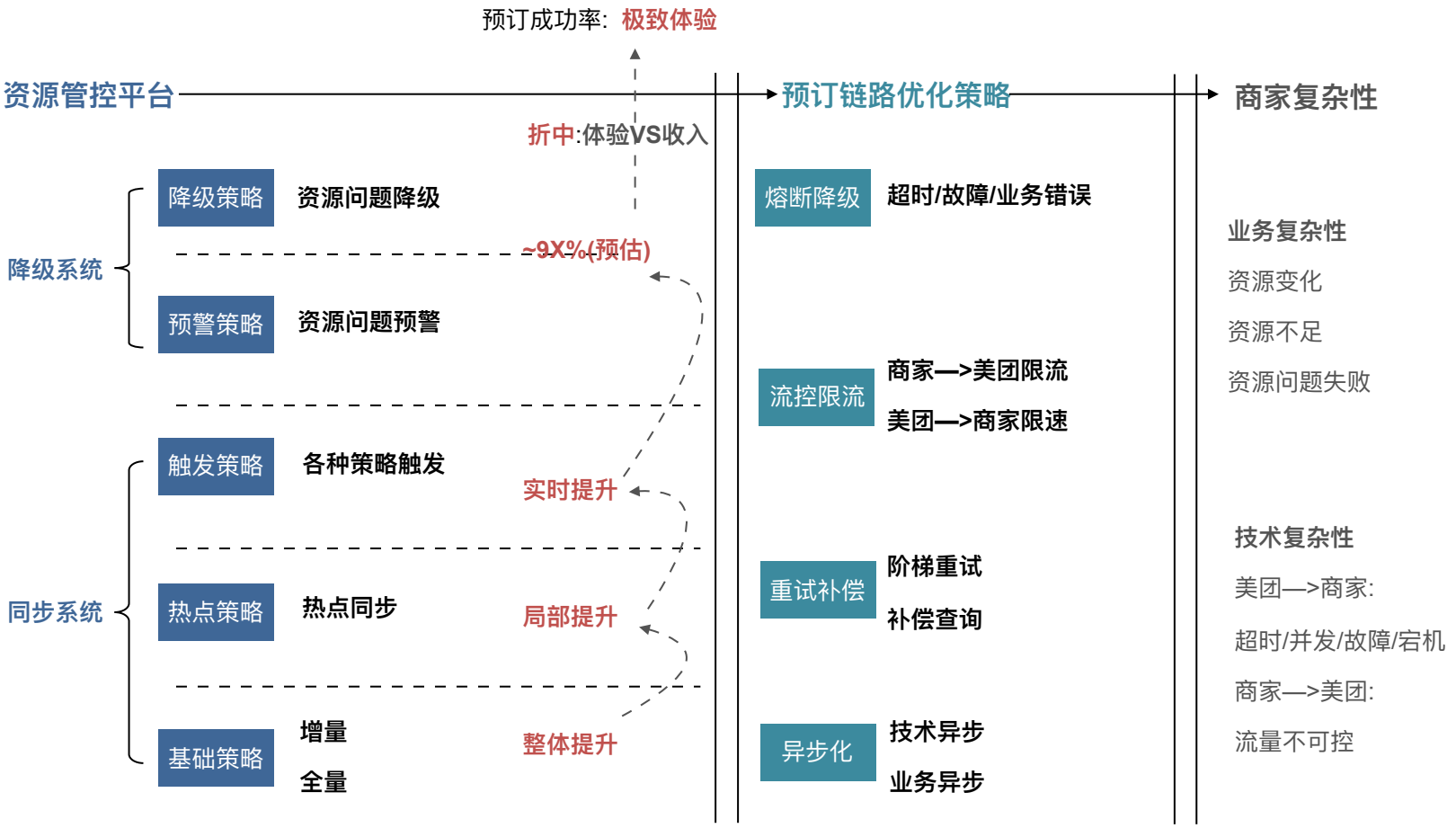
预订链路分析

(Total-F1-F2-F3) / Total ≈ **(Total-F2-F3) / Total** (内部失败可忽略)

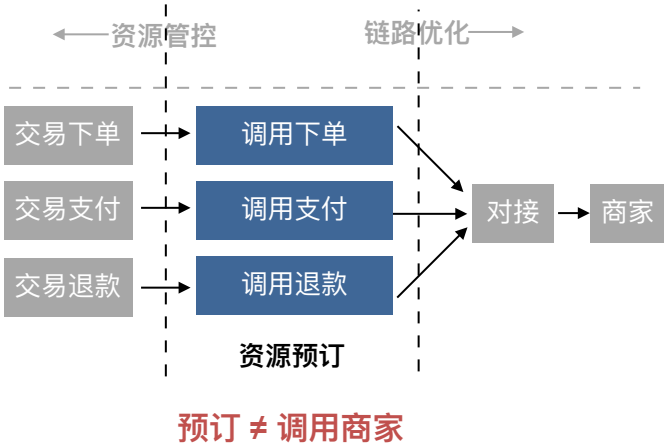
核心思路

- **技术复杂性 (F2)**
  - 链路优化—>资源获取能力
  - 提升预订成功
- **业务复杂性 (F3)**
  - 资源控制—>同步+预警+降级
  - 减少预订失败

架构演进实践—资源控制


预订成功率: 极致体验
资源管控平台
预订链路优化策略
商家复杂性
折中:体验VS收入
降级系统
降级策略
资源问题降级
~9X%(预估)
预警策略
资源问题预警
同步系统
触发策略
各种策略触发
实时提升
热点策略
热点同步
局部提升
基础策略
增量
全量
整体提升
熔断降级
超时/故障/业务错误
流控限流
商家—>美团限流
美团—>商家限速
重试补偿
阶梯重试
补偿查询
异步化
技术异步
业务异步
业务复杂性
资源变化
资源不足
资源问题失败
技术复杂性
美团—>商家:
超时/并发/故障/宕机
商家—>美团:
流量不可控

架构演进实践—资源预订

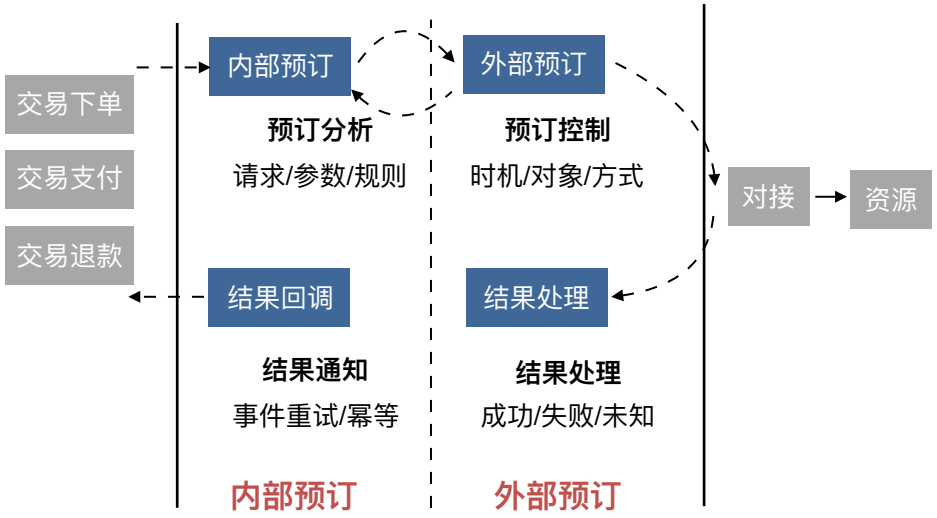


困难挑战

- 技术因素
- 业务因素
- 行业因素

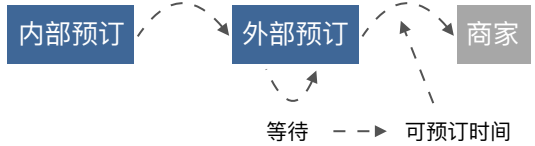
矛盾取舍

体验
供给
收入

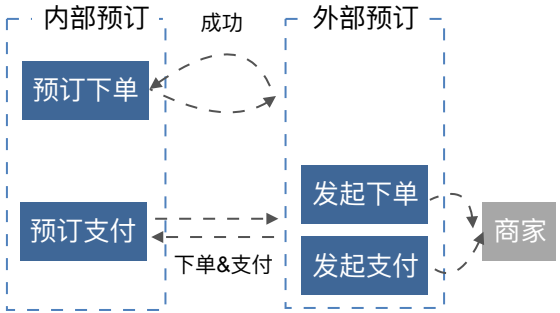


架构演进实践—资源预订模型

延迟模型



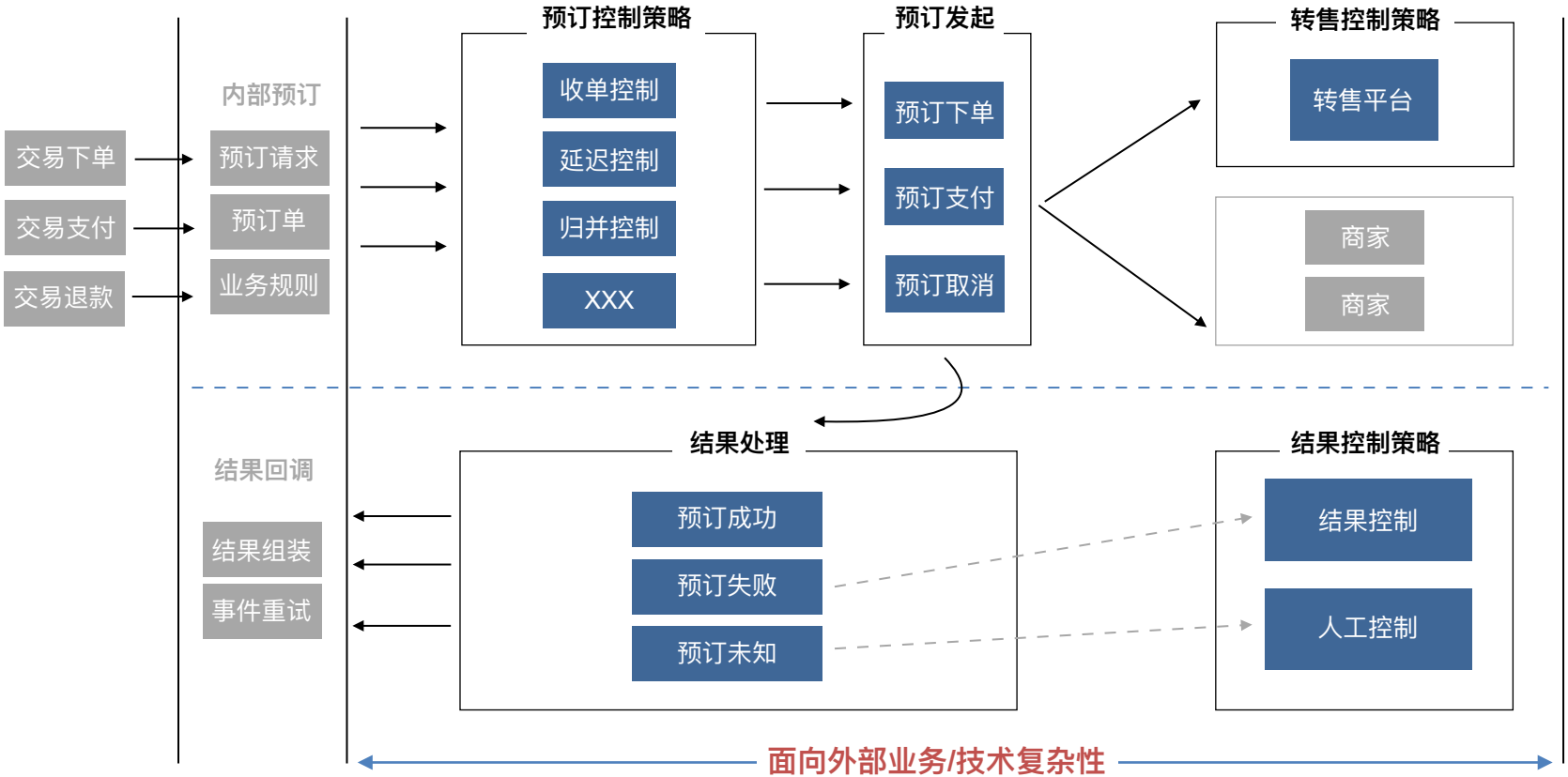
归并模型



核心思路

- 应对**商家业务/技术复杂性**
  - 业务复杂性: 线下处理不及时、断供
  - 技术复杂性: 超时、故障、停服
- 在**外部预订**和**结果处理**环节添加控制
  - 外部预订: 时机、对象、方式
  - 结果处理: 对非标结果的标准化处理

架构演进实践—资源预订架构



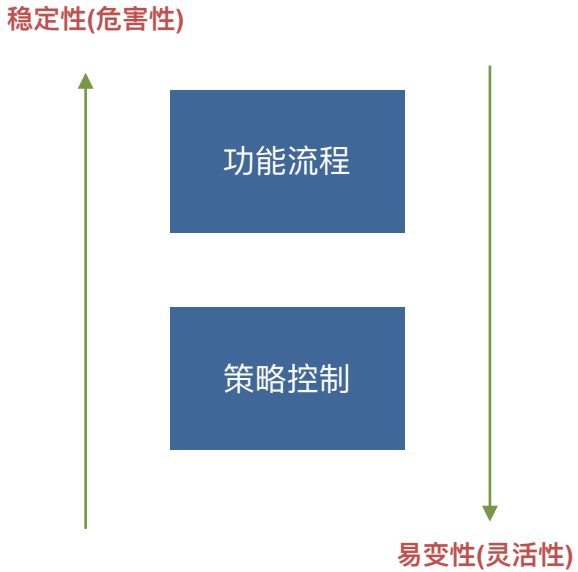
架构演进实践—策略控制带来的问题

大量策略控制带来的问题

- **灵活性/扩展性**: 策略迭代慢、无法适应精细化运营
- **稳定性**: 策略频繁变更的稳定性挑战
- **效率**: 策略的复用和统一性

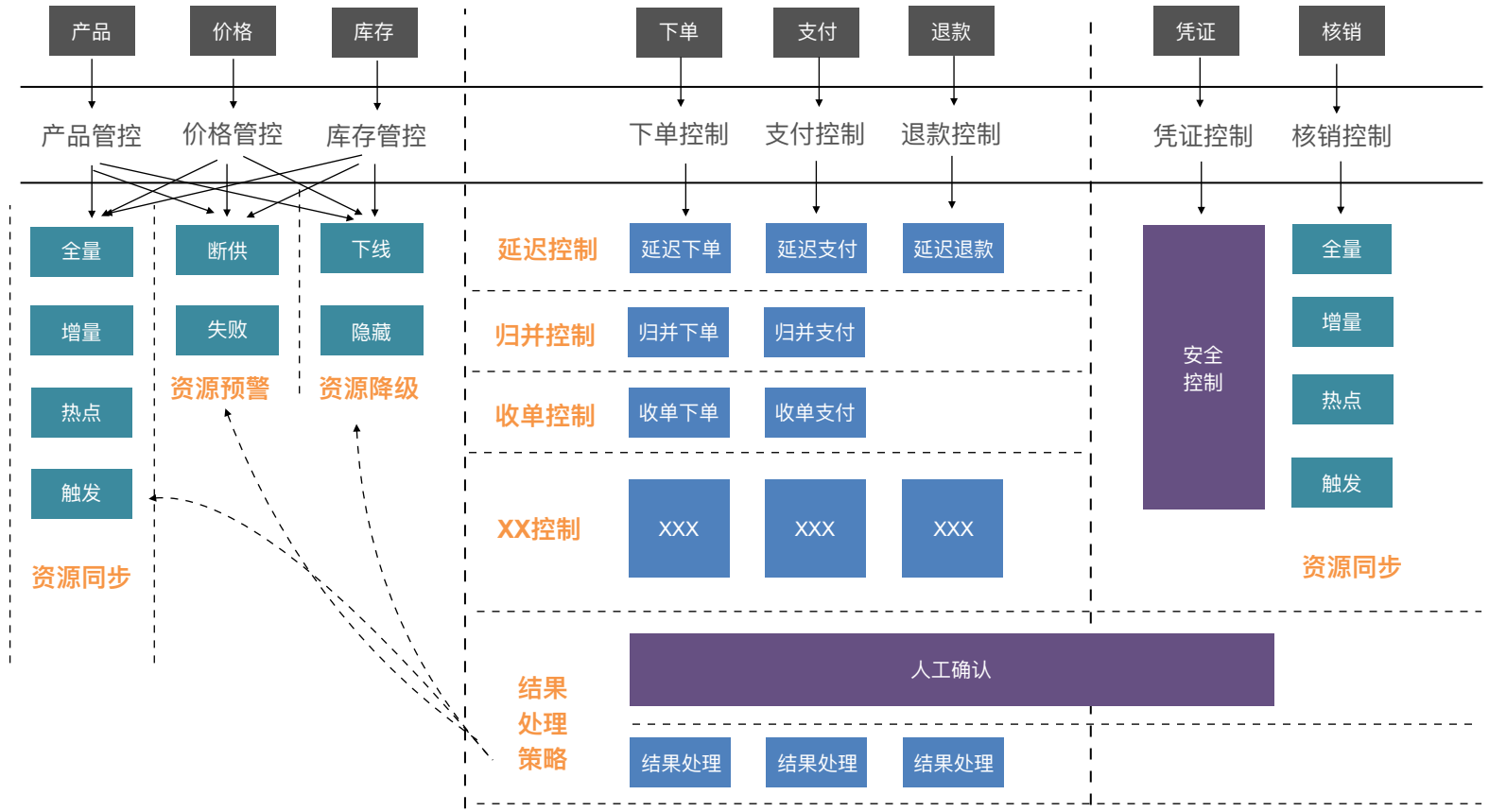
目标/解决思路

- **功能/策略解耦**
  - 策略平台独立建设
  - 策略组件复用，统一性收敛
- **易变/不变分离**
  - 稳定性分级: 重点保障功能稳定性
  - 策略失效：体验不是最优，功能OK

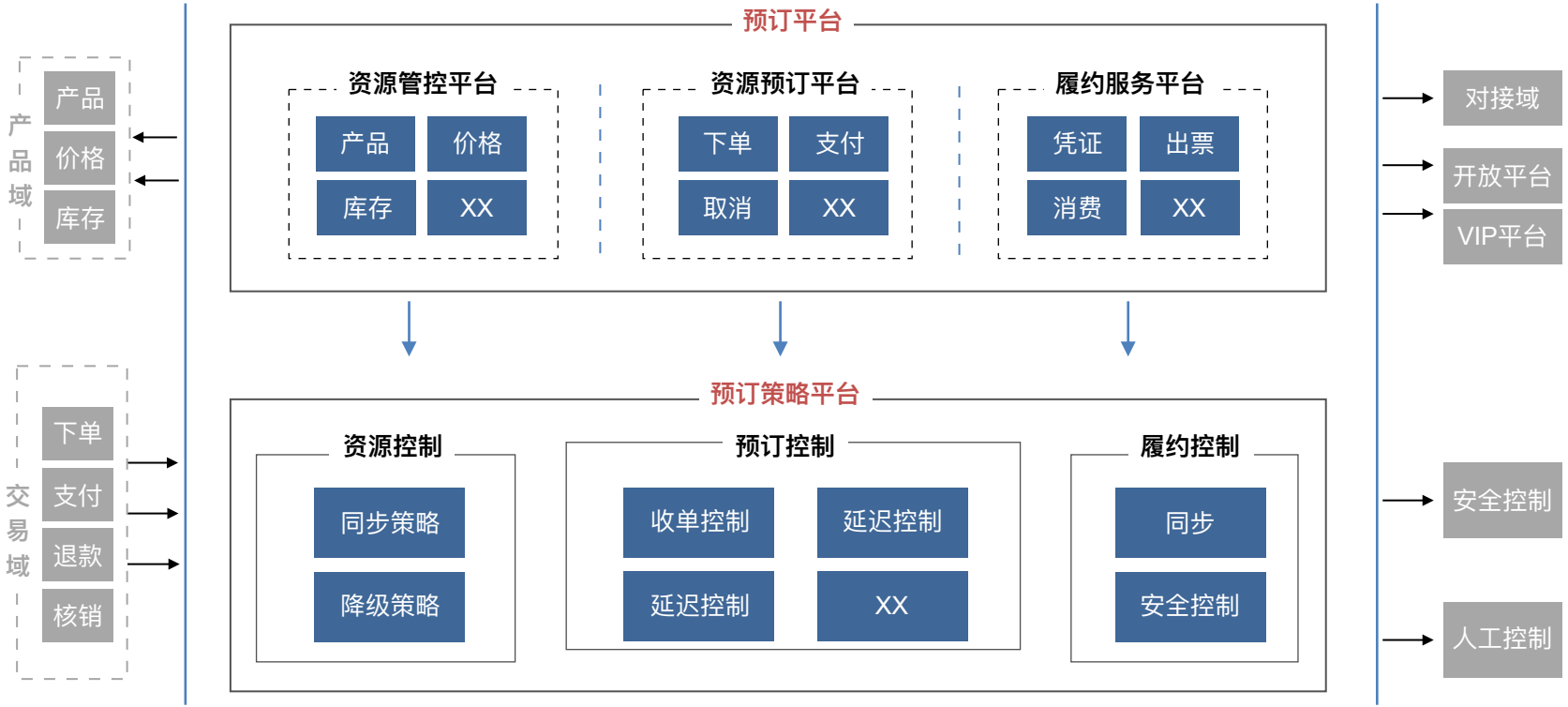


功能/策略 对系统的不同诉求

架构演进实践—策略控制抽取



架构演进实践—功能与策略分离



架构演进实践—预订总结

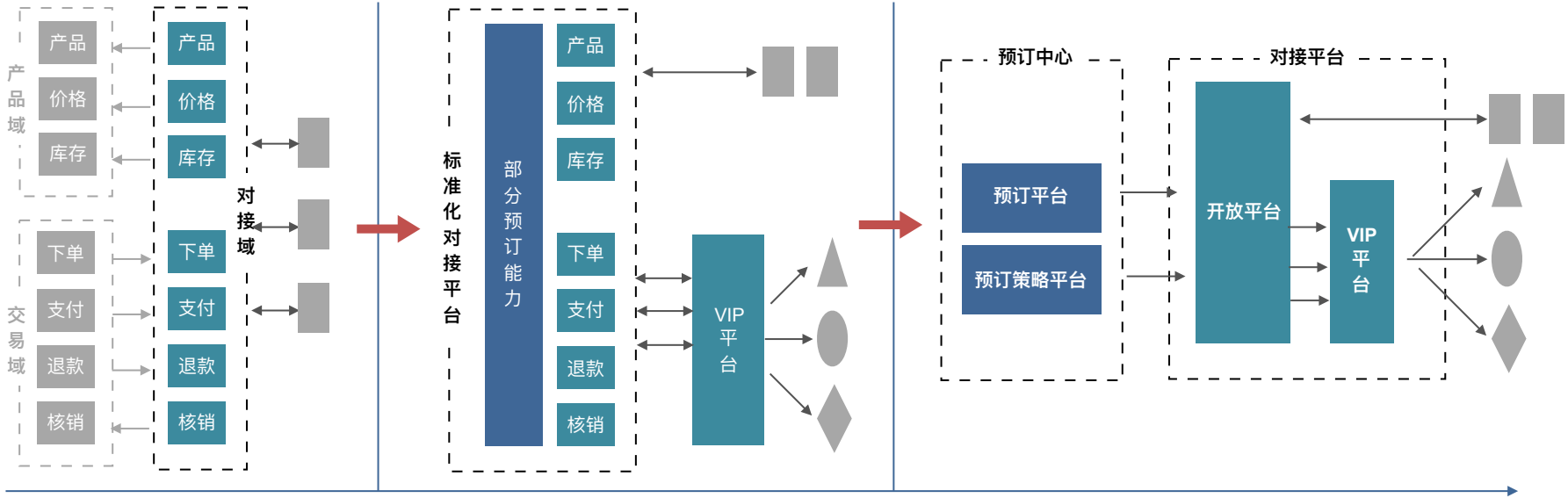
1、**领域分离: 预订 + 对接**

2、**架构分层** (网关、分离技术/业务复杂性)

3、**易变与不变分离**(功能 + 策略)

预订建设

架构演进实践—架构演进总结



粗放增长

- 对接API
- 自测平台

高速发展

- 标准化对接平台
- VIP对接平台

精细运营

- 预订中心
- 对接平台

目录

思考和总结

1、愿景驱动: 理想架构、稳定性/扩展性

2、问题驱动: 发现和解决实际问题

3、架构设计理念与业务特点结合

4、架构演进与业务发展阶段结合

Q & A