阿里巴巴集团 旗下公司

天猫 TMALL.COM

基于流程/事件，构建灵活业务代码

--使用轻量级的技术构建系统

2012.7

曹学锋/震南

Agenda

- 业务变化及架构应对
- 事件驱动基本介绍
- 业务实践

业务挑战变化

- 业务挑战
  - 业务快速变化
  - 业务复杂度不断增加
    - 业务节点复杂度增加
    - 业务流程变化
  - 业务异常控制

架构应对

- 关注业务特征
- 找出业务增长瓶颈
- 服务化
  - 沉淀出不变的部分
- 外化业务流程
  - 用事件粘合服务、规则、服务构建业务流程

流程化目的

- 一个系统好不好主要看它变化时是否适应，升级时是否轻松。
- 变化主要反应在几个方面
- 1.业务模型变化（只需重构某一节点）
- 2.流程变化（只影响一个流程）
- 3.节点压力变化（平行扩展能力）
- 4.流程压力变化（垂直扩展能力）
- 进一步促进业务沉淀如业务流程的沉淀，它是另一个层面的沉淀，模型变流程不变，流程往往是更稳定
- 异常恢复处理能力
- 流程可视化管理
- 统一监控管理
- 让重构不会变成重做
- 浮出业务组件
  - 让不稳定的多变的业务浮到上层
  - 让最懂业务的人维护独立的业务

服务化目的

- 统一技术体系
  - ESB、远程通信框架等
- 统一服务间通信的业务语言
  - 统一业务领域模型

- 让稳定的服务沉淀下来
  - 让专业的人做专业的事
  - 带来业务上的独立性

技术选择

- 选择合适的、能轻则轻
- 活用技术的思想，思想也是相通的，技术的实现是开放的
- 只有最合适的架构

事件 VS 流程

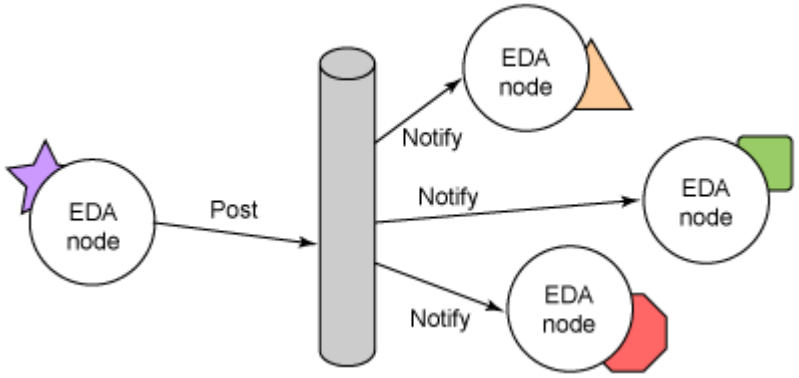
- 流程可以由事件驱动来实现
- 事件可以为流程提供业务、性能及控制上的支持

Agenda

- 业务变化及架构应对
- 事件驱动基本介绍
- 业务实践

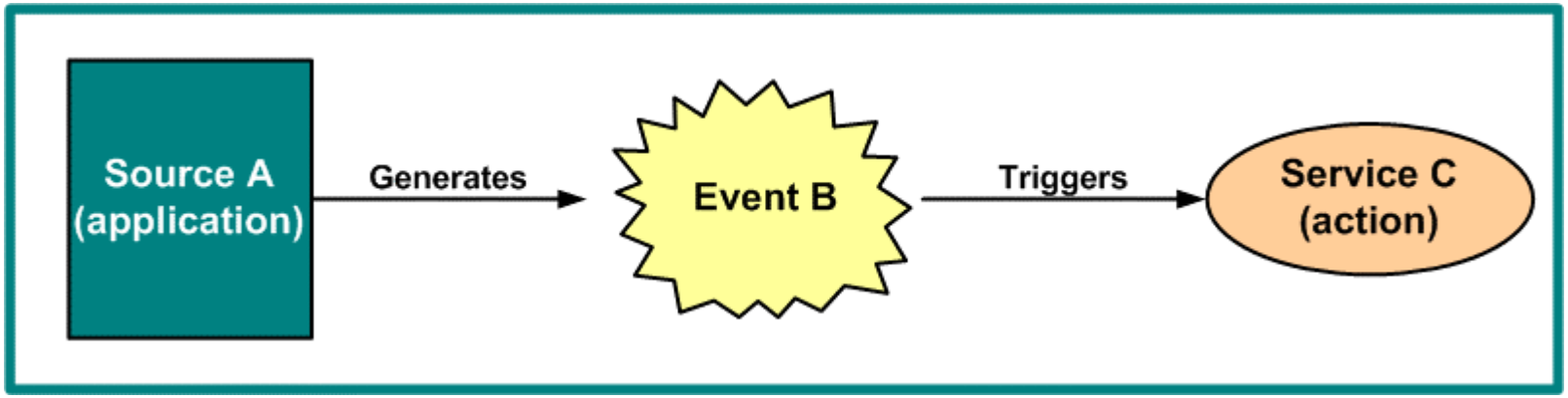
EDA基本特征

- 交互解耦
  - 订阅者对发布者是未知的
- 多对多的通信方式
  - 一个事件可以影响多个订阅者，
  - 一个订阅都可以处理多个事件
- 事件触发
  - 事件触发业务流的执行
- 异步



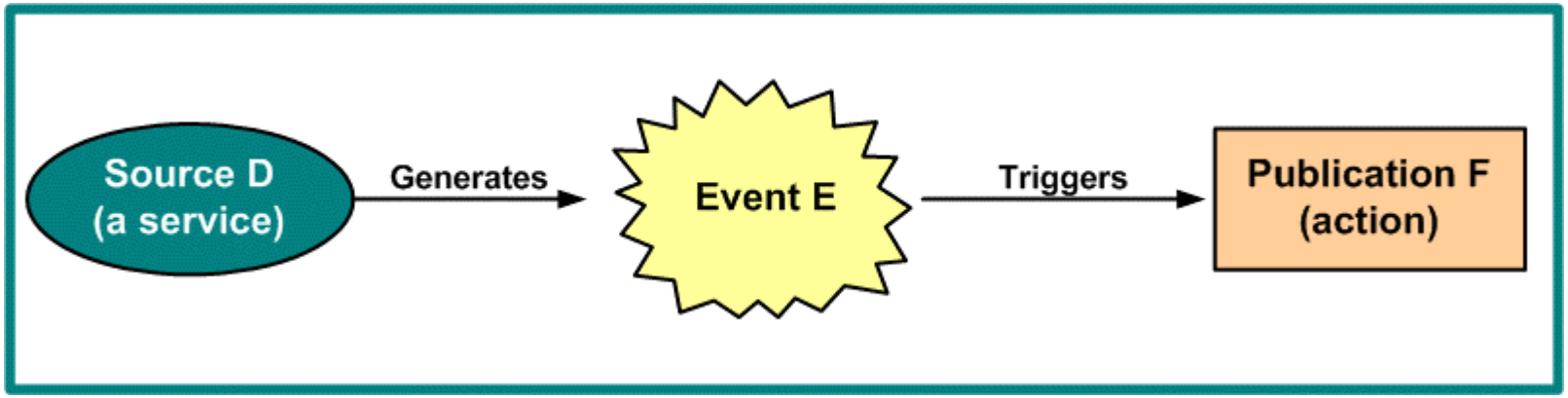
事件和服务 交互场景一

- 一个事件的产生可能会触发一个或多个服务
- 这些服务可能会执行一些基本的功能或是一个完整的业务流程



事件和服务 交互场景二

- 一个服务生成事件
- 事件可能会被立即传播或是执行下游的动作



ED-SOA架构的基本架构

Agenda

- 业务变化及架构应对
- 事件驱动基本介绍
- 技术实践

技术实践-准时送达服务产品

- 次日送达服务规则

- 当日16：00 前付款，次日18：00前送达
- 只有打上这个服务标签并下单并付款成功的商品才承诺时效

服务细则

- 当前需求
  - 库存
  - 成本
  - 时效
- 可能的需求
  - 优惠（不同级别卖家优惠不同）
  - 销售策略（不同渠道、销售范围等）
  - 服务商服务质量（如配送公司的服务评级等）
  - ……

次日达路由规则伪表达

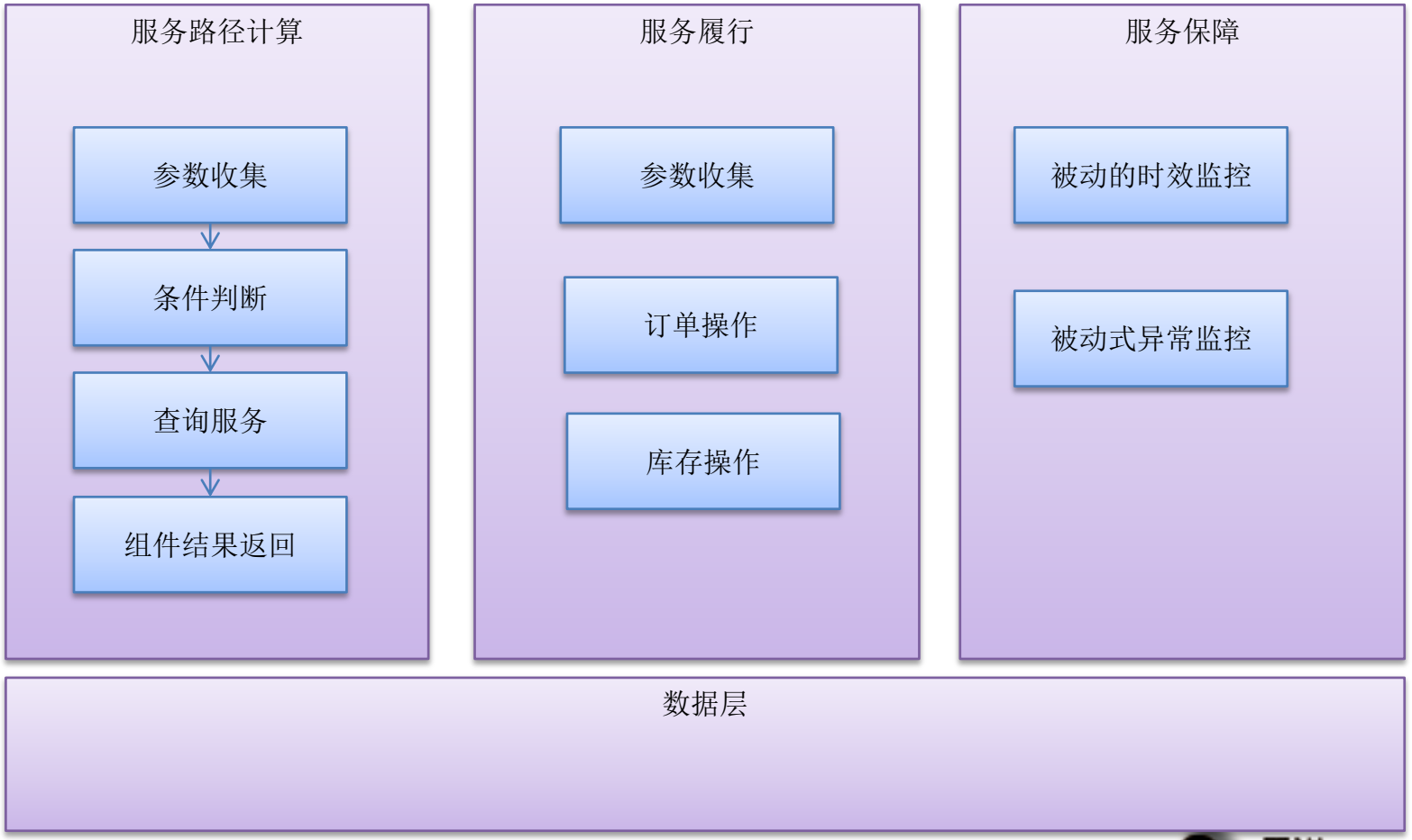
- 请求库存信息（inventory_service）
- 请求配送信息 (tms_service)
- 覆盖范围 (area_service)
- 时效 (time_service)
- 成本计算 (charge_service)
- 服务商KPI（sp_service）
- 规则匹配
  - 时效规则
  - 成本规则
  - 库存规则
- 规则匹配
  - (quantity > 10 && fullfill(area) && &&nodes<=2 && && chage< 20 && sp_kpi >= 3.5)

分析

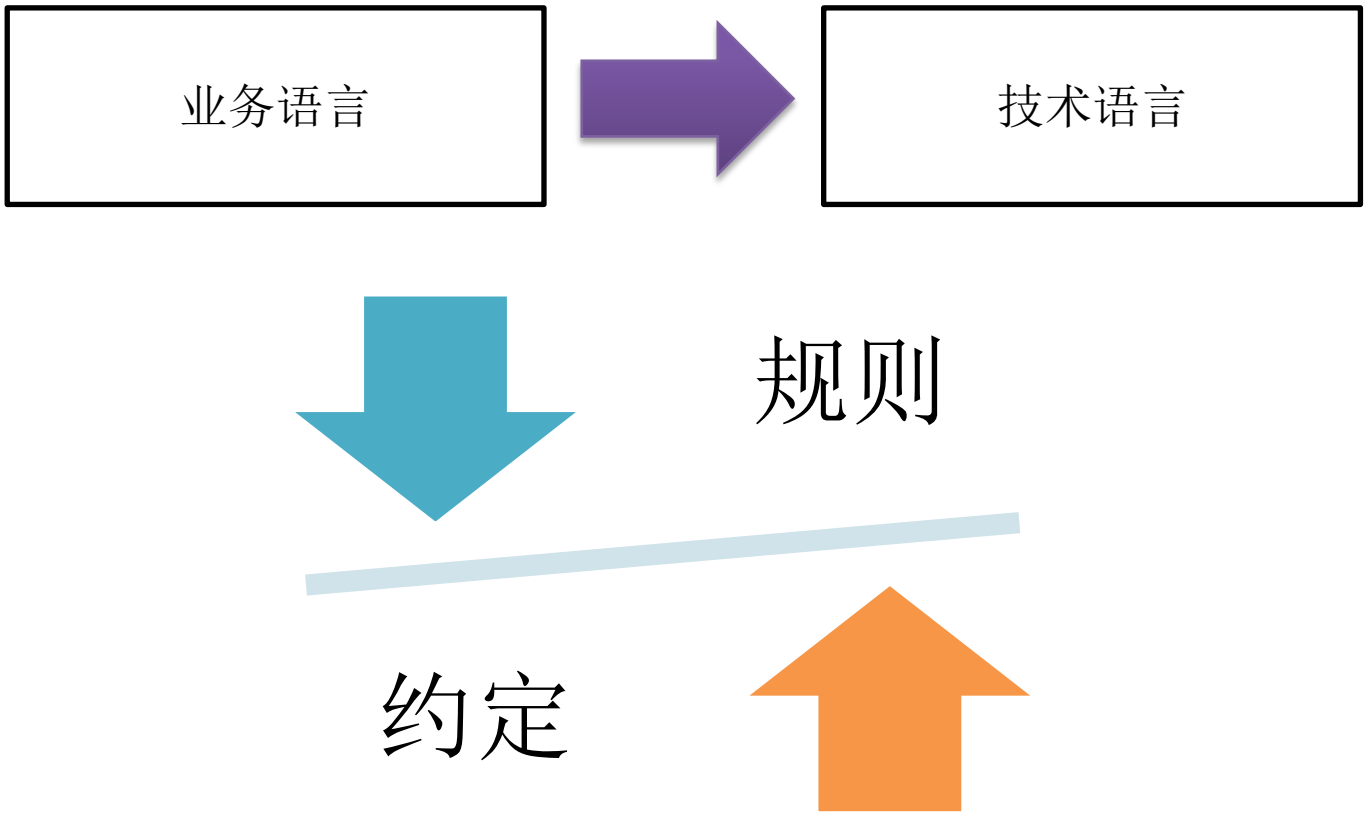
- 服务规则计算（计算是否应该可以使用服务 面向规则 事件 流程）
- 怎样履行服务（面向流程）
- 服务保障（面向事件）
- 业务表达转换为技术语言

要完成哪些业务功能

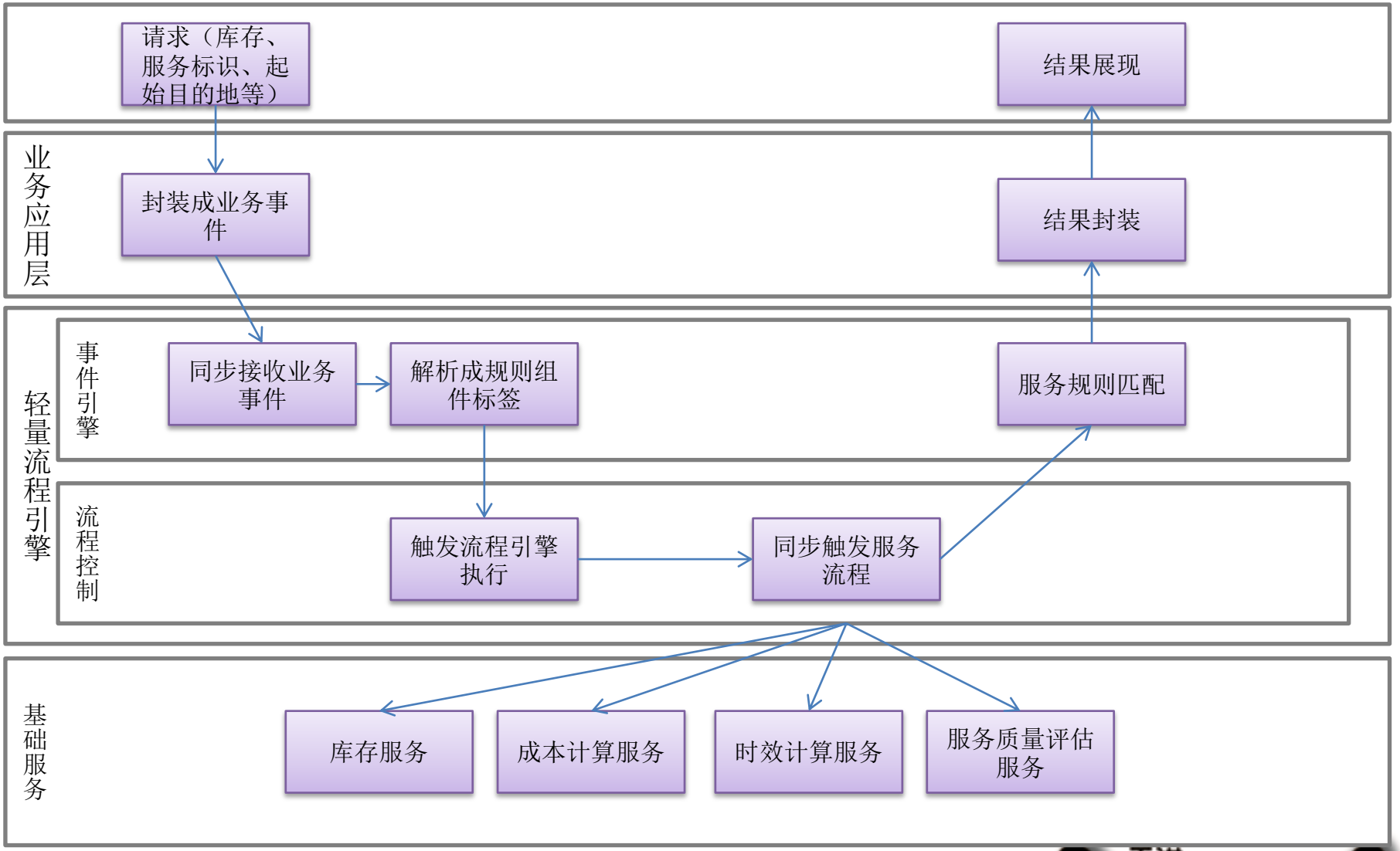
原始业务状态


服务路径计算
参数收集
条件判断
查询服务
组件结果返回
服务履行
参数收集
订单操作
库存操作
服务保障
被动的时效监控
被动式异常监控
数据层

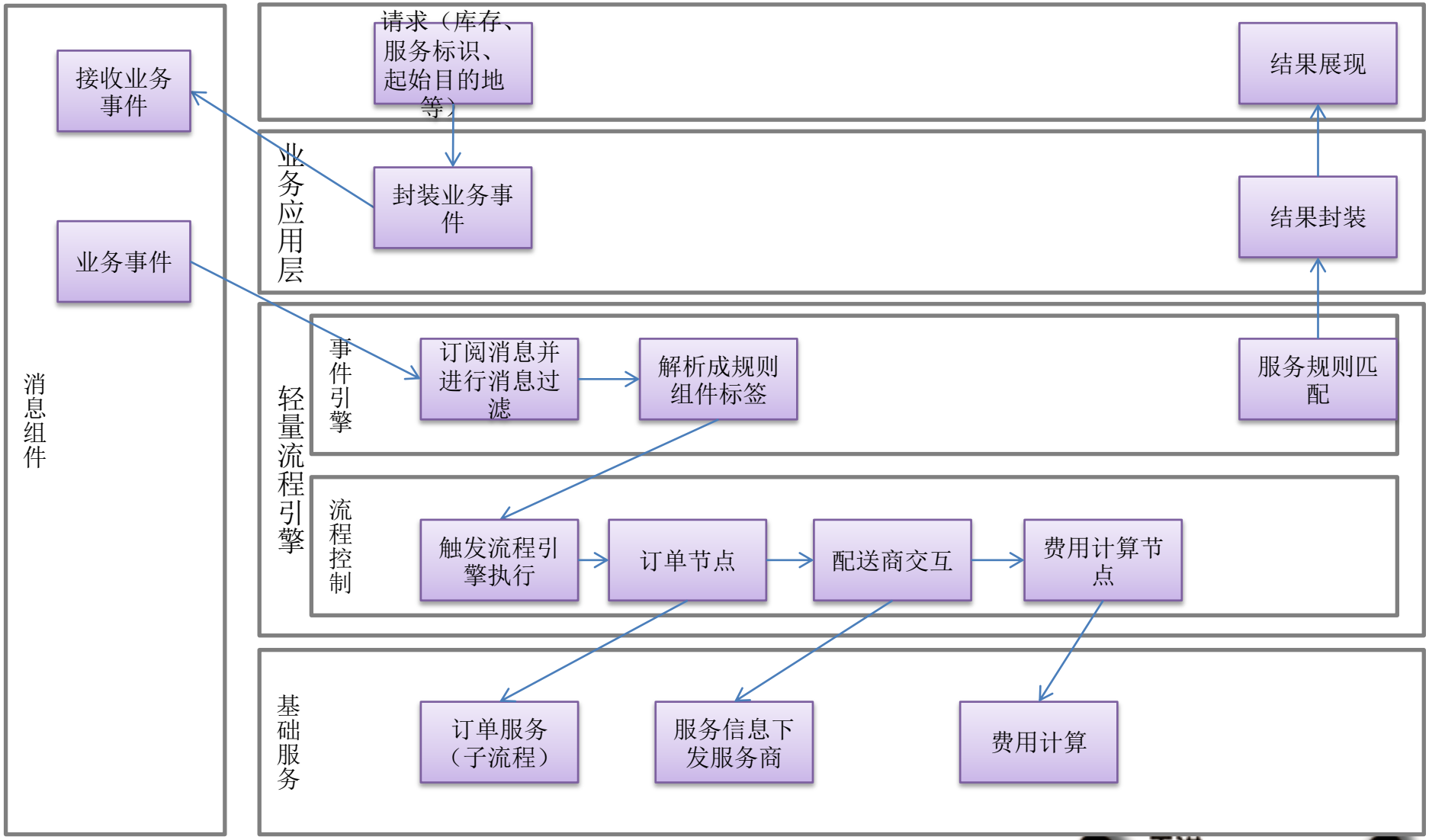
基于规则 OR 约定



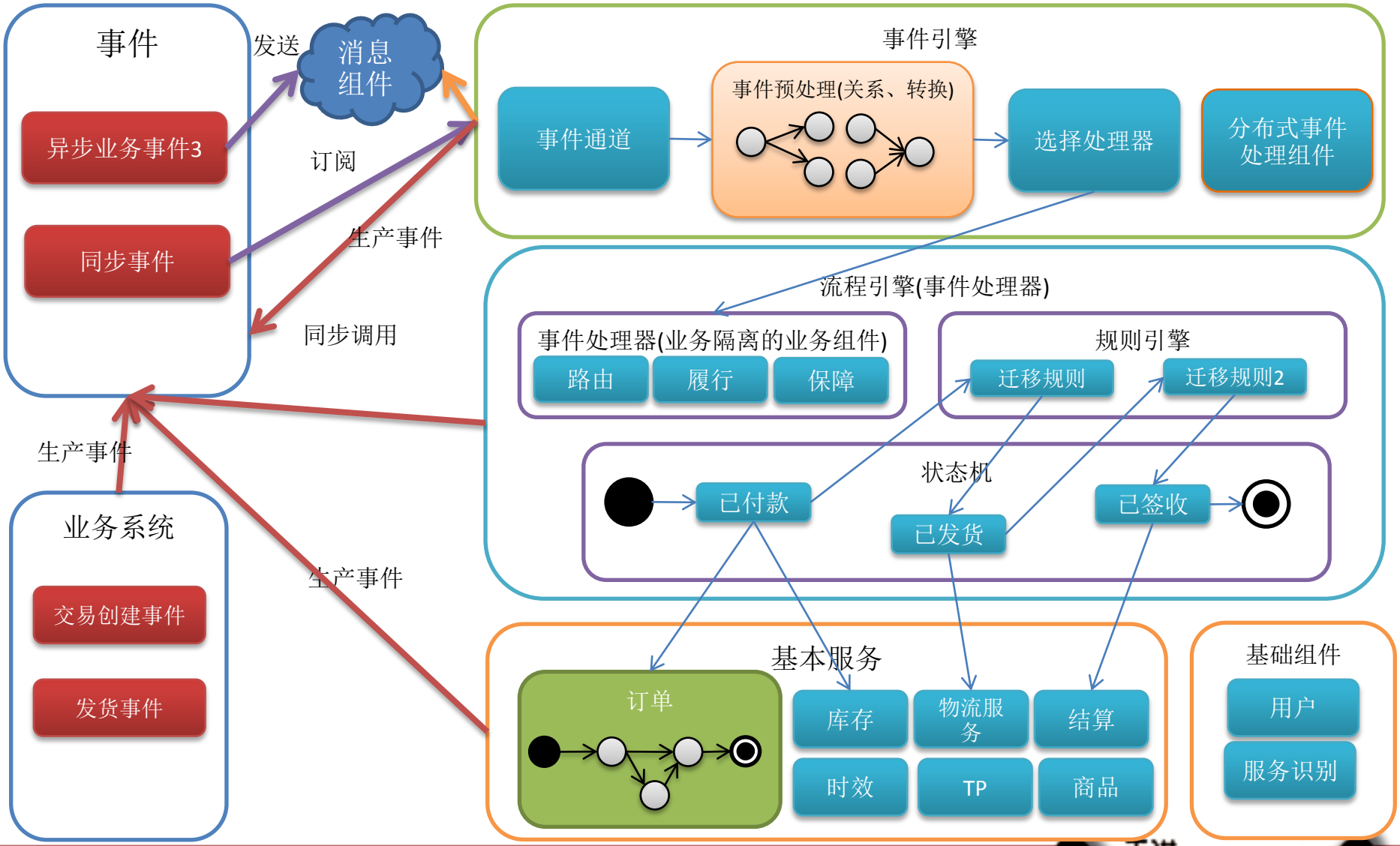
服务路由(同步)



服务履行（面向任务和事件）



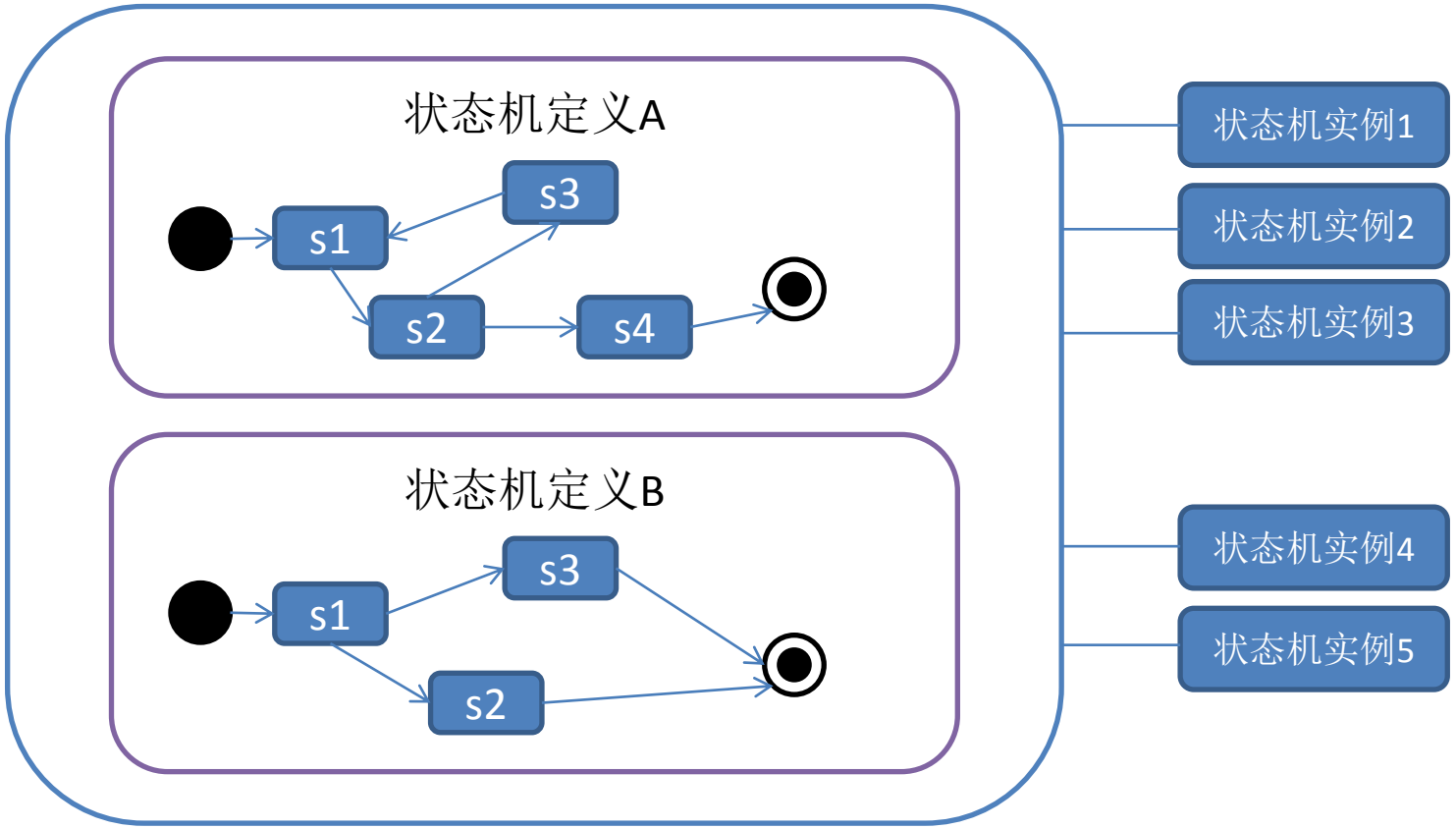
基本技术架构


事件
发送
消息组件
异步业务事件3
同步事件
订阅
生产事件
同步调用
事件引擎
事件通道
事件预处理(关系、转换)
选择处理器
分布式事件处理组件
流程引擎(事件处理器)
事件处理器(业务隔离的业务组件)
路由
履行
保障
规则引擎
迁移规则
迁移规则2
状态机
已付款
已发货
已签收
生产事件
业务系统
交易创建事件
发货事件
生产事件
基本服务
订单
库存
物流服务
结算
时效
TP
商品
基础组件
用户
服务识别

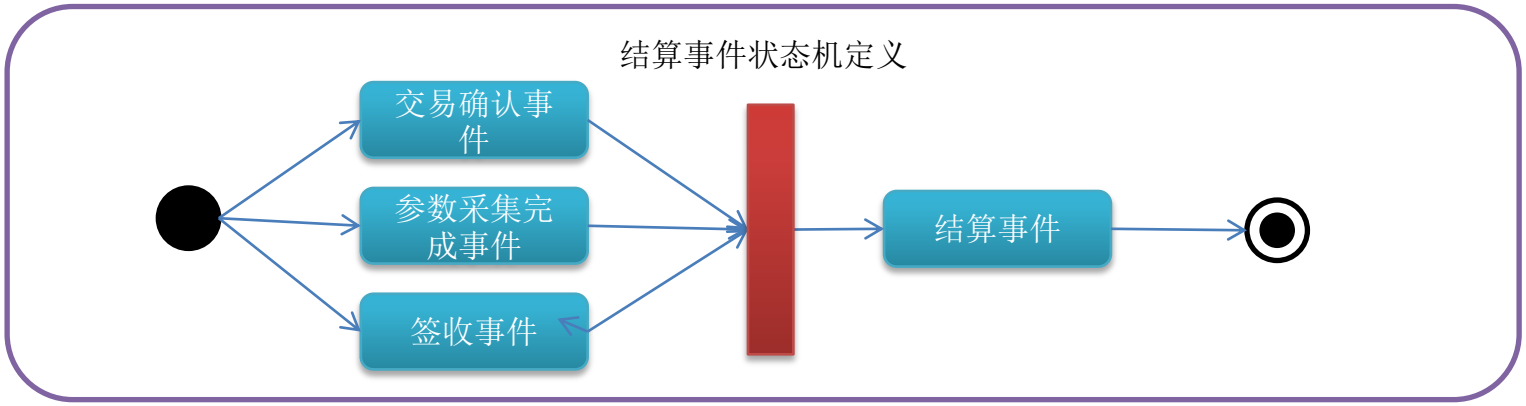
轻量事件&流程引擎结构

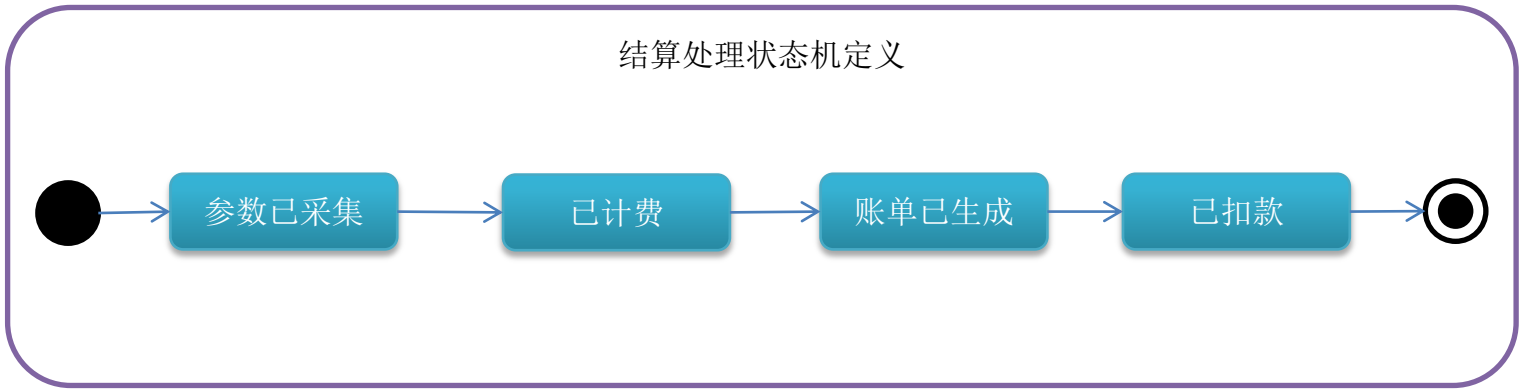
- 事件引擎基本设计
  - 事件对象结构设计(事件类型、流程实例ID、业务对象)
  - 事件引擎调用接口（同步&异步）
  - 事件存储(根据业务特征选择)
  - 状态机实现（包括事件关系管理）
- 基础产品
  - 消息总线产品（发布&订阅 activeMQ等）
  - 缓存
  - 分布式事件处理组件
  - 规则引擎(http://code.taobao.org/u/qhlhl/)
  - 远程调用框架(http://code.taobao.org/p/tbschedule)
- 业务要求
  - 基本服务抽取(统一领域模型)
  - 抽象出业务隔离的业务组件
  - 基本组件抽取（用户识别、服务识别等）

状态机引擎

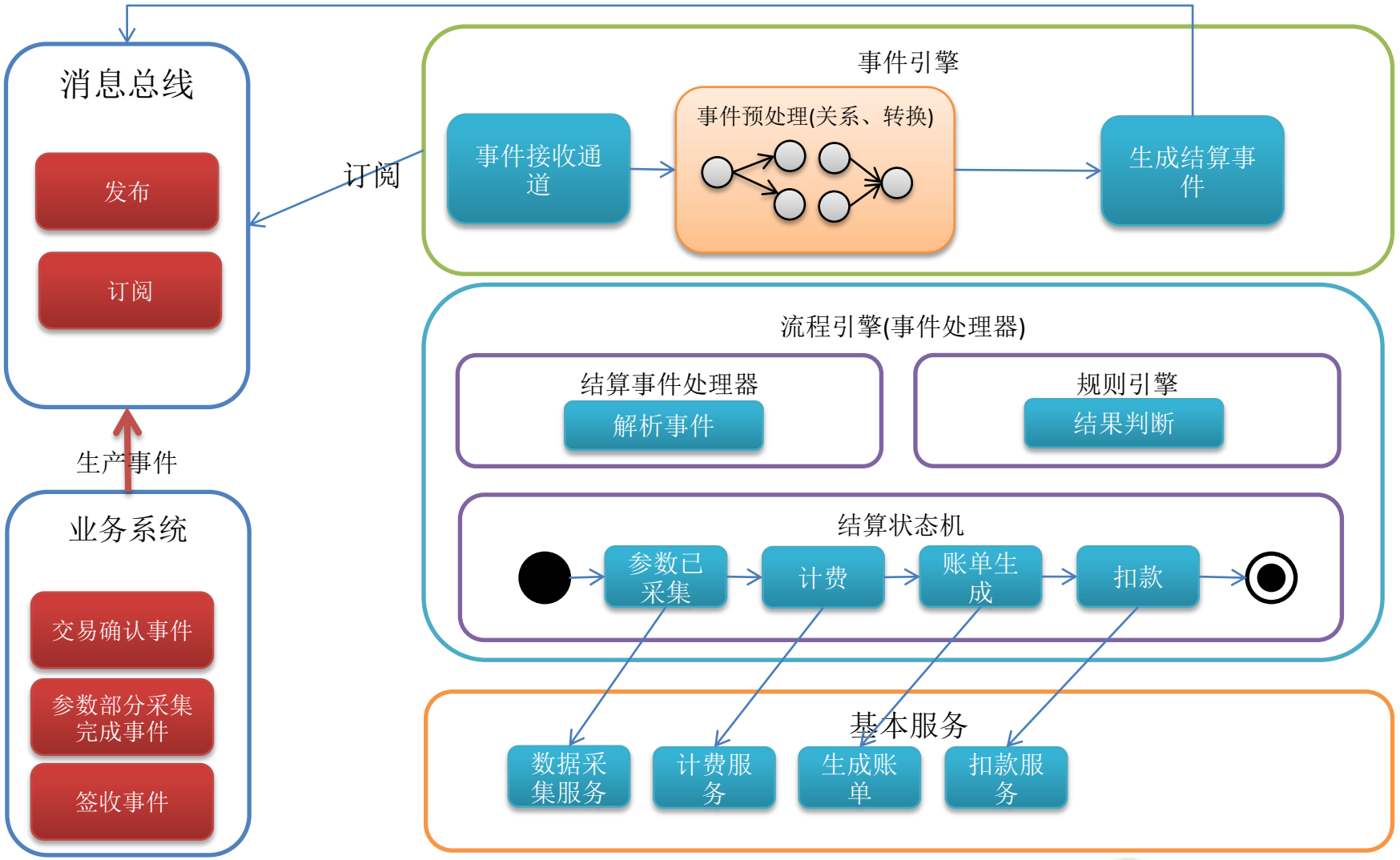

状态机定义A
s1
s2
s3
s4
状态机定义B
s1
s2
s3
状态机实例1
状态机实例2
状态机实例3
状态机实例4
状态机实例5

结算流程-状态机定义





结算事件处理



结算流程关键点描述

- 组合事件处理
  - 无序事件处理
  - 缓存使用
  - 缓存状态机实例的信息
- 状态机实例ID生成
- 状态机实例信息修改

总结与思考

- 抓住系统用户
  - 业务用户
  - 开发者
- 选择合适的技术及架构
  - 尽可能选择轻量技术，不被技术绑架
- 切中业务场景
  - 根据应用场景选择合适的技术
  - 核心业务决定核心架构

Q&A

THANKS

-THE END-